COMMISSION INTERNATIONALE DE L'ENSEIGNEMENT MATHEMATIQUE

(The International Commission on Mathematical Instruction)

THE INFLUENCE OF COMPUTERS AND INFORMATICS

ON MATHEMATICS AND ITS TEACHING

* * *

* *

*

STRASBOURG 25 - 30 MARCH 1985

SUPPORTING PAPERS

ulp

Institut
de Recherches
sur l'Enseignement
des Mathématiques
10, rue du Général Zimmer
67084 STRASBOURG CEDEX

ICMI Symposium
March 1985
STRASBOURG

SUPPORTING PAPERS

INDEX

<u>Some experiences</u>

<u>Symbolic systems and algebra</u>

VISUALIZATIONS

Ce symposium a été organisé grâce au support du Fond mathématique international, alimenté par les contributions de :

- l'UNESCO,
- IBM Europe
- Hermannéditions,
- IBM France

ainsi que de :

- la Royal Society
- l'I.C.S.U. (International Council of Scientific Unions)
- la D.C.R.I. (Direction de la Coopération et de la Recherche Internationale),
- la S.M.F. (Société Mathématique de France),
- l'Université Louis Pasteur de Strasbourg.

COMMISSION INTERNATIONALE DE L'ENSEIGNEMENT MATHÉMATIQUE
(INTERNATIONAL COMMISSION ON MATHEMATICAL INSTRUCTION)

THE INFLUENCE OF COMPUTERS AND INFORMATICS ON MATHEMATICS AND ITS TEACHING

Extrait de *L'Enseignement Mathématique*, tome 30, 1984

L'Enseignement Mathématique, t. 30 (1984), p. 159-172

COMMISSION INTERNATIONALE DE L'ENSEIGNEMENT MATHÉMATIQUE (INTERNATIONAL COMMISSION ON MATHEMATICAL INSTRUCTION)

AN INTRODUCTION TO THE ICMI STUDY ON THE COMPUTER AND INFORMATICS

The International Commission on Mathematical Instruction is planning a number of studies on topics of international interest. Each study will be built around an international seminar and will be directed towards the preparation of a published volume intended to promote discussion and action at national, regional, or institutional level. Each study will attempt to identify key problems within a specific area and to provide up-to-date accounts of relevant thought, research and practice.

The effect of computers and informatics on mathematics and on its teaching at the university and pre-university level is the theme of the first such study to begin. The Planning Committee has already met and has prepared the discussion document which follows. We now invite reactions to that document.

It is essential here, however, to emphasise that the ICMI studies do not aim to find an "ICMI approved" solution to any particular problem. Rather we wish to encourage the discussion in depth of key issues and the sharing of knowledge and experience. For that reason we ask those who respond to our invitation to bear in mind the international nature of the exercise and the consequent need to focus attention on aspects of the topic field which have general and not merely national interest. Thus, for example, it would be in keeping with the study's aims if descriptions of newly devised curricula concentrated not merely on the end effect of such changes in terms of revised syllabus content, but also emphasised the principles which governed curriculum design and the constraints which had to be satisfied.

Case-studies of change form one type of response to the discussion document which would be welcomed. We hope, however, that the document will also encourage individuals, national sub-commissions and other national committees to respond in a variety of ways. Many points relating to the effect of the computer and informatics on mathematics and on how one might respond to these through curriculum change are made in the discussion paper. There will, however, be other issues which readers wish to raise, and also a need to develop, elaborate and exemplify in greater detail ideas which are only hinted at there.

We hope that many responses will be generated and that papers will be sent to the Secretary for the study, Dr. F. Pluvinage, IREM, 10, rue du Général-Zimmer, 67084 Strasbourg Cedex, France.

An international seminar at which invited papers and those received in response to the discussion document will be considered is to be held in France between 25 and 30 March, 1985. Attendance at this meeting will be by invitation only, but a number of places will be reserved for those who submit responses to the discussion document. Further details of the seminar can be obtained from Dr. Pluvinage.

Following the seminar it is intended to publish *Proceedings* which will include a survey of the key issues raised as well as a selection of papers submitted.

THE INFLUENCE OF COMPUTERS AND INFORMATICS ON MATHEMATICS AND ITS TEACHING

AN ICMI DISCUSSION DOCUMENT

prepared by

R. F. CHURCHHOUSE, B. CORNU, A. P. ERSHOV, A. G. HOWSON, J. P. KAHANE, J. H. VAN LINT, F. PLUVINAGE, A. RALSTON, M. YAMAGUTI

Computers and informatics are changing all societies of our time. As the steam engine introduced the first industrial revolution, so the computer is introducing what is often called the second industrial revolution. That first revolution was accompanied by the development of the physical sciences; one must expect that new sciences connected with informatics will accompany the second. The prospects, then, are immense: new needs, new sciences, new technologies, new qualifications, the elimination of repetitive or laborious work, and, of course, new social challenges to be met.

Mathematics does not escape this movement, and this is why ICMI has taken the initiative of organising an international study on the theme: the influence of computers and informatics on mathematics and its teaching. As a first stage, the present document is being circulated for discussion. It is organised around three important questions:

1. How do computers and informatics influence mathematical ideas, values and the advancement of mathematical science?

2. How can new curricula be designed to meet the needs and possibilities?

3. How can the use of computers help the teaching of mathematics?

So far as questions 2 and 3 are concerned, we are limiting our study to the curriculum and teaching at university and pre-university level (from the age of 16 years). School mathematics will be the subject of another ICMI international study. Naturally one will find the same topics and ideas appearing at every level, in answer to the questions posed. Two aspects, in particular, are essential: the influence of technology which allows better things to be done more quickly, and in different ways, and the influence of fundamental concepts of informatics, in the forefront of which is found algorithmics.

1. The effect on mathematics

Mathematical concepts have always depended on methods of calculation and methods of writing. Decimal numeration, the writing of symbols, the construction of tables of numerical values all preceded modern ideas of real number and of function. Mathematicians calculated integrals, and made use of the integration sign, long before the emergence of Riemann's or Lebesgue's concepts of the integral. In a similar manner, one can expect the new methods of calculation and of writing which computers and informatics offer to permit the emergence of new mathematical concepts. But, already today, they are pointing to the value of ideas and methods, old or new, which do not command a place in contemporary "traditional" mathematics. And they permit and invite us to take a new look at the most traditional ideas.

Let us consider different ideas of a real number. There is a point on the line R, and this representation can be effective for promoting the understanding of addition and multiplication. There is also an accumulation point of fractions, for example, continued fractions giving the best approximation of a real by rationals. There is also a non-terminating decimal expansion. There is also a number written in floating-point notation. Experience with even a simple pocket calculator can help validate the last three aspects. The algorithm of continued fractions—which is only that of Euclid—is again becoming a standard tool in many parts of mathematics. Complicated operations (exponentiation, summation of series, iterations) will, with the computer's aid, become easy. Yet even these simplified operations will give rise to new mathematical problems: for example, summing terms in two different orders (starting with the largest or starting from the smallest) will not always produce the same numerical result.

Again, consider the notion of function. Teaching distinguishes between, on the one hand, elementary and special functions—that is, those functions tabulated from the 17th to the 19th century—and, on the other, the general concept of function introduced by Dirichlet in 1830. Even today, to "solve" a differential equation is taken to mean reducing the solution to integrals, and if possible to elementary functions. However, what is involved in functional equations is the effective calculation and the qualitative study of solutions. The functions in which one is interested therefore are calculable

functions and no longer only those which are tabulated. The theories of approximation and of the superposition of functions—developed well before computers—are now validated. The field of elementary functions is extended, and functions of a non-elementary nature are introduced naturally through the discretisation of non-linear problems. Informatics, too, compels us to take a new look at the notion of a variable, and at the link between symbol and value. This link is strongly exploited in mathematics (for example, in the symbolism of the calculus). In informatics, the necessity of working out, of realising the values has presented this problem in a new way. The symbolism of functions is not entirely transferable. This has resulted in computer languages of different types: thus the notion of a variable in LISP does not correspond exactly to that in some other languages in which variables have values.

For the last of our examples let us consider sets of points linked to dynamical systems, iteration of transformations or stochastic processes. The use of computers has brought new life to their study, both by physicists and mathematicians, and has given rise to a new terminology: for example, strange attractors, fractals.

From these examples it can be seen that computers and informatics have *stimulated* new research, *restored* to the mathematician's consideration questions recently neglected but previously studied over a long period of time, and *made possible* the study of new questions. We hope that as a result of the discussions connected with the ICMI study new light will be cast on each of these aspects.

There has always been an experimental side to mathematics. Euler insisted on the rôle of observation in pure mathematics: "the properties of numbers that we know have usually been discovered by observation, and discovered well before their validity has been confirmed by demonstration ... It is by observation that we increasingly discover new properties, which we next do our utmost to prove". Computers have suddenly greatly increased our possibilities for observation and experimentation in mathematics. The solution of the non-linear wave equation, the soliton, was discovered by numerical experimentation before it became a mathematical object, and gave rise to a rigorous theory. In the iteration of rational transformations it is the plots obtained by computers which have guided recent research. An entirely new art of experimentation is developing in all branches of mathematics. Calculations which were formerly impracticable are now easily accomplished; it is now a question of working out an appropriate plan of action. Visualisations are possible and they form a unifying bond between mathematicians in offering them subjects for study on which specialists from different disciplines can unite to work. There has been a considerable increase in the number and variety of stimuli which allow, indeed encourage, one to query and investigate their mathematical nature in order to establish and appreciate their inter-relationships. An awareness of these new possibilities has for some years penetrated research mathematics. Only on rare occasions has it been allowed to influence and infiltrate our teaching. However, these possibilities for experimentation, now practicable on a large scale, are most full of promise for the renewal and improvement of the teaching of mathematics.

Mathematics is also, and will remain, a science of proof. But the status of proof is not immutable. The level of rigour and degree of formalisation depends upon time and place. For some twenty years the fashion was for non-constructive proofs of existence theorems: methods of ideals for g.c.d., pigeon-hole principle for rational approximations, axiom of choice for functional analysis, probabilistic methods without explicit constructions, etc. Today the point of view has changed. Whenever possible, one makes use in a proof of an algorithm which permits one to obtain effectively the object sought.

Computers have had another effect on the status of proof, as has been shown in the celebrated case of the four-colour theorem. Until now, the most long and involved proofs were edited and published and the reader had access to and control over any exterior information required (tables, references). In principle, a mathematician working alone was supposed to be able to follow and verify every step of a proof thanks to this method of presentation. Now new types of proof have appeared: numerical proofs in which there occur numbers of a size and in quantities which preclude their being manipulated by hand, and algorithmic proofs dependent on the effectiveness and correctness of the algorithms. Computer-aided proofs have produced the need, therefore, for a new form of professional practice. This does not yet seem to have been codified. Doubtless it will be in the future.

Algorithms have played an important rôle in mathematics since Euclid, and even more since the birth of algebra. They constitute the most important mathematical constituent of informatics. We have referred to the rôle of algorithms as tools in proof and we are all aware that they are essential tools in calculation. Now, however, they are becoming more and more a study in themselves. Fascinating questions now arise on space and time complexity—on how to formulate algorithms so as to minimize computer storage space and running time—and on the development of algorithms

suitable for processors running in parallel. To take but one example, by mathematical ingenuity the time complexity for the fast Fourier transform algorithm has been reduced from n^2 to $n \log n$, which is of considerable practical importance for large values of n. There are other problems concerned with the effectiveness of algorithms, their correctness and the way in which they can be elaborated. We note, as an example, the rôle of invariants and fixed points when establishing the correctness of algorithms.

One must also stress how algorithms are increasingly being called upon to play a central rôle in society: they arise in business and commerce, in technology and in automation. Mathematical problems arise then in many new domains, and mathematical methods have an increasingly far-reaching applicability.

Finally, from now on symbolic systems will enable the computer user to carry out difficult calculations within algebra and analysis. The possibilities raised are enormous, and one must take the measure of the actual performance of such systems and of their rôle in research mathematics, as well as of the influence they should have on the teaching of mathematics at the university and pre-university levels. Informatics, for example, extends the field of mathematical research on formal calculus.

2. The effect of computers on curricula

Curricula are generally the product of a long tradition, and their evolution is governed by two principal factors: the needs of society and the state of the discipline. The needs of society are very diverse: in each country, studies prepare for different professions, each of which has its own demands; between different countries there will be varying priorities. *A priori*, social needs introduce into curricula an element of diversity and even of divergence. On the other hand, reference to the discipline of mathematics itself is usually a unifying factor, when the specialists agree amongst themselves on what is essential content. And this unity also responds to a social need, to have a common body of knowledge and a shared language.

We have therefore to consider two major series of questions: the first relating to the expressed needs of society, to local experiences, to national policies; the second relating to new possibilities, to the adaptations which will have to be made as a result of new requirements, to choices prompted by the present state of knowledge and technique.

First we present three questions prompted by the social context (the national framework, the teaching of scientists, the industrial environment).

Question 1. In each country are there new mathematical curricula--permanent, provisional, experimental—motivated by the introduction of computers and informatics? The responses which we have so far received point to the existence of such experimental curricula.

Question 2. Mathematics has a duty to serve those in other disciplines —physicists, engineers, biologists, economists, etc. What are the changes prompted by the growing importance of computers and informatics within these disciplines? The partial replies we have received have come from the computer scientists themselves.

Question 3. What mathematics is necessary as a part of basic scientific culture—at a university level—within the new industrial environment? Those responses which we have had—coming from computer scientists—point to a strong theoretical demand; the use of computers and of informatics demand *more* mathematics, *better* understood, and would lead to a new equilibrium between "pure" and "applied" mathematics.

Let us pause at this point before going on to pose a new series of questions.

Doubtless, informatics will have three major effects on the orientation of teaching. First of all, symbolic mathematical systems are going to render simple and rapid, questions which were previously difficult and complicated. Already today there exist programs to evaluate definite integrals, to solve differential equations, even to calculate explicit solutions of certain functional equations. Thus mathematics teaching can lay less emphasis than formerly on the setting out and practising of classical methods of integration. On the other hand, our teaching can permit a student, by calling upon the available systems, to encounter a much greater number of problems and so understand better the underlying mathematics. The more such programs that will be at our disposal, the more necessary it will be for the student to understand the mathematical theory if (s)he is not to lose his/her bearings.

Next, informatics makes many calls upon the help of discrete mathematics: combinatorics, graph theory, coding theory. The applications of informatics to management, communication and information make little use of the differential and integral calculus, but they make use of varied structures on finite sets. It is advisable, then, to ask whether discrete mathematics should replace certain classical parts of analysis in the basic core

of mathematics provided for students and whether certain fundamental concepts of analysis might not with advantage be approached *via* a study of discrete situations. For example, the place of series in analysis courses might need to be modified.

Finally, then, the general effect of computers and informatics on mathematics will have necessary consequences on its teaching, on the importance attached to subjects and to methods, and in the order chosen for the presentation of material.

In all the various branches of mathematics one can envisage computers supplying numerical and visual experiences intended to foster intuition. One can also favour algorithmic presentations of theories and proofs.

These thoughts lead us to pose a second set of questions:

Question 4. What is the mathematics underlying symbolic mathematical systems? How should they be introduced into the curriculum?

Question 5. What discrete mathematics should be introduced?

Question 6. What changes can be envisaged in the order of presentation of topics (series before integrals, statistics before probability, probability before integration, ...)?

Question 7. In particular, what elements of logic, numerical analysis, statistics, probability, geometry can be introduced from the very beginning of university courses?

Question 8. What are the foreseeable changes in the way individual topics are presented, particularly when one takes into account the available algorithms (Newton's method for solving equations, continued fractions for real numbers, polynomial interpolants in integration, triangulation in linear algebra, ...)?

Question 9 (Of major importance). What content might possibly be omitted in the foundation courses (17-18 years)?

The changes brought about on curricula by informatics and computers will obviously have consequences for the training needed by teachers. In addition to supplying the elements of computer science and informatics which they will need, we must also prepare them to teach mathematics in a new way. This problem is going to arise as much at the level of in-service training as at that of the initial (pre-service) training of teachers.

We therefore pose the following question:

Question 10. What elements of computer science and informatics should be introduced into the training of teachers, and how can they be prepared and helped to teach mathematics in a new way, consonant with the new computing context? Some experience in this area already exists.

3. The computer as an aid to the teaching of mathematics

3.1. The general effects of computers

The use of computers compels one not only to recognise in the area of experiments a source of mathematical ideas and a field for the illustration of results, but also a place where confrontation will permanently occur between theory and practice. This last poses a problem, which will occur in the training of teachers as well as of students, of promoting the *experimental attitude* (observation, testing, control of variables, ...) alongside, and on a par with, the *mathematical attitude* (conjecture, proof, verification, ...). Does it suffice, to speak, as some people do, of "experimental mathematics"?

We now have a triangle, student-teacher-computer, where previously only a dual relationship existed. Is there not a danger that, in order to preserve as much as possible the traditional student-teacher relationship, students' work on a computer will be restricted to simplistic activities which are "without risk" for the teacher?

Students are bound to be aware (as a result of their environment and the media) of the widespread use of computers as well as their associated peripherals, even interconnecting systems and data banks. They have also seen spectacular graphics displayed on a screen, or traced on a plotter. As a result of this, students have new expectations with respect to teaching in general and that of mathematics in particular. How can the computer be used by and with the students in order to meet these new expectations?

In addition to the changes of interest to which informatics leads, one must also draw attention to the changes in the difficulty of exercises and problems. Not only will the use of a computer change the order of difficulty of exercises, but it will also change the relative difficulties of the various ways of solving the same exercise. How can one arrive at new hierarchies and take them into account when one constructs exercises?

3.2. Objectives and modes of operation

There are various methods of using a computer in our teaching. The teacher can use the computer like a "blackboard", in the same way as one proceeds when giving demonstrations in the experimental sciences. However, the use of an interactive computer permits a much higher degree of interaction with the audience. This particular mode has been tested in various places, but its wider use depends upon the provision of more, and more varied, software. What are the specific requirements which must be met by such software?

The computer can be used by students, individually or in groups of two or more, in order to accomplish predetermined work (this is really programmed learning adapted to work on the computer: unfortunately, there seems to be little software of this type available having any great mathematical interest). In a similar manner, the computer can provide the student with a permanent and readily accessible form of self-evaluation.

Another use of the computer is for "practical work": the experimental manipulation of mathematical objects in connection with open-ended problems (e.g. statistical treatment of data, geometric explorations, the manipulation of functions, ...).

One sees, then, the need for the development of "software banks" in order both to provide support for teachers and lecturers and also to encourage further improvements. This software, which should be available to all, would be located in "multi-media centres" in the middle of institutions and seen as a means of communication on a par with written texts, films, ...

The preparation of software will call upon the united skills of mathematicians, computer scientists and practising teachers. How should one share out the work in order to produce satisfactory software and within which structural framework?

Finally, another use of the computer, in the school or university environment, is in the context of a "computer club". After an initial period of familiarisation, it is the users/members who are chiefly responsible for determining the paths to follow. This type of work is of relevance not only to students, but also to their teachers. What needs can be identified, therefore, for those who are going to be responsible for the training of teachers?

3.3. The treatment of particular areas

The peripherals used (screen, printer, plotter, ...) determine different ways of using informatics. The adaptation to mathematics poses some general problems, like that of the handling of symbolic writing which is not linear, despite the apparent linear sequence of characters in a normal text. For example, consider the various methods employed to reduce to a linear form mathematical statements often best presented in the form of a tree.

We now consider methods of employing the computer to meet needs to be found in various areas of mathematics which are taught at the levels of education under consideration.

In all of these branches one will note the central rôle of visualisation, of experimentation, of simulation, and of the way in which the computer fosters the generation and refining of conjectures.

First, however, we pose a general question. A certain number of fundamental concepts are used in the teaching of mathematics, often in an implicit manner, for example, intuitive logic, the concepts of a variable, of a function, ... Can informatics help us bring precision to, and increase our understanding of, such concepts?

Statistics and probability; data processing. The computer permits the processing of data on a truly grand scale. Problems of sorting data into classes are no longer relevant. Again, simulation is a tool which can hold a place in probability similar to that of plotting figures in geometry. Thus it is possible, thanks to pseudo-random techniques for selection, to provide "reality" for all types of conceivable situations—betting, decision making, testing ...

Geometry. The production of graphical images (e.g. perspective views of objects in space, orbits) and the concept of computer-aided design (graphics software) are extremely helpful for the development and fostering of intuitions. They make it possible to explore geometric objects and figures and provide access to new figures. What changes does plotting by means of a computer introduce with respect to geometry founded on the use of ruler and compass?

Linear algebra. The algorithmic approach furnishes tools for mathematical demonstrations (e.g. pivotal condensation) and leads us to approach in a different manner the study of such questions as inversion, the solution of systems of equations, and the decomposition of matrices. Moreover, visualisation can give support to intuition, e.g. for the study of eigenvalues and of diagonalisation. Do not such techniques as the simplex method merit a place in our teaching?

Analysis. As a result of the effects of symbolical systems, exercises on differentiation, searching for primitive integrals and finding finite Taylor series, are destined to decrease in importance. On the other hand, the graphical representation of functions and finding approximate solutions of numerical or functional equations will become worthy of additional consideration. Experimentation can provide opportunities for the discovery and formulation of qualitative properties before they are formally proved, for example, for the solution of differential equations. Approximation brings with it problems of convergence, beginning with sequences and series. Moreover, the qualitative aspect of the concept of convergence, the numerical study, leads naturally to the quantitative aspect, speed of convergence. Finally, discretisation provides a further field for experimentation, e.g. for functional equations.

Numbers, numerical analysis. The numbers of a machine are very different from those of a mathematician. This leads one to explore the differences and, *en passant*, to consider the principles of numerical symbolism. In another connection, should we be taking the use of parallel processors for research in numerical analysis into account at the teaching level?

Sets, combinatorics, logic. The methods of working now force one to give operative definitions (the enumeration of surjections $S(n, p)$ is a simple example of recursivity, which also allows one to give a working meaning to a surjection). In this area, too, the rapid production of numerical results permits easy exploration and the devising of conjectures. Does the learning of formulae by experience constitute a particular current interest in this field?

In traditional fields of study there are subjects which demand new and special attention because of the particular characteristics of working on a machine: that it uses discrete methods. It is important, therefore, to pay attention to theoretical approaches to discrete topics (e.g. difference equations); at this time, complete courses of discrete mathematics are being proposed for students. Is it really true that this provides us with a new theme for teaching?

3.4. Assessment and recording

The teacher often understands the assessment of his pupils' learning in the restricted sense of evaluation through examinations, while assessment of teaching is usually ignored. The computer, however, now makes possible a variety of ways of controlling assessment, ranging from the presentation of exercises to students to the management of individual files. The use of the computer to construct and to conduct evaluatory tests has hardly been experimented with up to now except in the teaching of computer science itself. Should we foresee a general development in the growth of examinations "on a computer", and if so how are such tests to be designed?

The notion of control and evaluation can also be extended to what happens when we use a computer. The juxtaposition of the output from a computer with mathematical results is specially relevant to such "experimental control". At the end of this report it is time to mention the usefulness of results which do not correspond to what has been foreseen and to those programs which do not function perfectly. It is obviously helpful to recall that frequently programs will not work at the first attempt. What is the mathematical interest in such mistakes?

3.5. The training of teachers

We have referred above to the problem of the content of teacher-training. It is equally advisable to question the form that this training should take, particularly the provision of in-service education for practising teachers. What can be envisaged if we think of "light" in-service training —day-release or short-term courses—and what if teachers can be given at least one year's complete leave of absence from teaching? But even this last is not sufficient considered in the context of the gradual evolution of materials and software. Here it would seem essential to open local support centres designed to provide a follow-up to such courses, to supply up-to-date software and to encourage teaching experiments. It would be a great pity if interest in computers and informatics resulted in the establishment of "heavy" administrative machinery, distant from most teachers, in which decisions relating to teaching were taken. What networks (local, regional, national, international) is it advisable, therefore, to set up and what type of connections must be established between them?

REMARKS ON THE INTERPLAY OF MATHEMATICS, COMPUTERS AND SOCIETY

Rudolf STRAESSER, IDM Bielefeld

Paper from the
ICMI-Conference
on the

"Influence of computers and informatics on mathematics and its teaching"

(Strasbourg, March 25-29, 1985)

ABSTRACT:
The Paper is restricted to four remarks relating society's needs, computers and mathematics:

(1) It is difficult to get scientific knowledge on society's needs for mathematics and mathematics education. So there is no easy way to curriculum development serving society's needs.
(2) With computers used at work, arithmetic seems to be less important while structural, mathematical knowledge is getting more important.
(3) Technology is not the unique agent which determines education - nor is it the discipline mathematics. An interplay of technology, organisation of work, the discipline concerned (in this paper: mathematics) and education is a more adequate description.
(4) There are differences between developed, industrialised and rural, developing countries. This is true also for mathematics, computers and education.

0. Introduction

"Curricula are generally the product of a long tradition, and their evolution is governed by two principal factors: the needs of society and the state of the discipline". This can be read in ICMI, part 2. Without asking, if all factors of the evolution of curricula are touched upon, the paper will concentrate on the first: the so-called needs of society. In a short paper one cannot possibly cover the whole topic. The paper is restricted to four remarks relating society's needs, computers and mathematics. Three remarks are formulated on the basis of work in the field of technical and vocational education; the fourth remark is a more general one.

1. A Warning

At a first glance, society's needs seem to be easily detected: Just look for the mathematics used in the various parts of society, especially the mathematics used in various sectors of the economy, and condense it into a curriculum! This approach was really tried and led to interesting results. The mathematics used on the job centers around topics such as basic arithmetic, percentages and proportions (rule of three). In some vocations, these techniques already seem to be all the mathematics needed /cf. KNOX, for further details and results, for critical remarks cf. DAWES and JESSON/. At least in those days (the 1970ies) nothing was to be found about computers.

Unfortunately this approach of a direct transcription of the professional use of mathematics into a curriculum did not work for the following reason: the "real" professional use of mathematics is not as easily discernible as claimed. "Considerable differences ... were found to exist even within occupations which might be assumed ... to be similar. It is therefore not possible to produce definitive lists of the mathematical topics of which a knowledge will be needed in order to carry out jobs with a particular title" /cf. COCKCROFT-Report, p. 19/. Another problem seems to be the identification of the uses of mathematics on the job /cf. MATHEMATICS in Employment, p. 8/, which partly explains the difficulties of employers, but does not ease the task of the curriculum developer. In addition to these problems the curriculum developer has to bear in mind that there are "important differences between the ways in which mathematics is used in employment and the ways in which the same mathematics is often encountered in the classroom" /cf. COCKCROFT-Report, p. 19/.

Even if those experiences with the needs of society have been made partly before the widespread use of computers in production, distribution and administration, these findings can be a warning against a too simplistic approach to identifying needs of society. Consequently society's needs cannot easily be taken as a justification of (mathematics or computer science) curricula.

2. The Growing Importance of Structural Knowledge

The second remark can be illustrated by two developments in the use of mathematics at work:

The COCKROFF-Report stated that it was rather difficult to find mathematics used at the workplace. Especially in business and administration the computer takes over a lot of work of the kind of producing tables, doing the accounting and internal calculation of the companies. But: studies on the introduction of commputers to this economic sector show that even more structural background knowledge is needed with the advent of computers /e.g. BAETHGE/. Can this interpreted as a growing need for more abstract, if not mathematical concepts in order to understand and take part in the running of a company? Or more specific: Does the growing use of business-software (like spreadsheets etc.) urge for more abstract and mathematical instruction - or education?

A different example from industry points to the same direction: The use of Computer-Assisted-Design(CAD)-techniques and computerized numerically controlled (CNC) machines seems to ask for at least some understanding of coordinate geometry and linear algebra. In the Federal Republic of Germany this was even formulated as a special research and development problem by the central institution responsible for technical and vocational education /cf. BUSCHHAUS/. Again in this case mathematical knowledge seems to be needed as a structural background for the specific use of computers - and again: the need for singular calculations of special values seems to diminish. To sum up the first point: Vocational use of computers seems to lower the importance of arithmetic and makes more important structural, mathematical knowlegde.

What was stated above can be formulated in more general terms:
With the growing sophistication of technology and organisation of work learning at work is getting more and more difficult. This is true partly for financial reasons, as the break-down of a machine produced by an unexperienced worker or learner tends to be more expensive with every step of technological development. Technological development increases the cost of machines operated by a worker and leads to an increase of financial loss in case of errors. This development leads to increasingly serious problems with the training on the job.

This is also true for organisational reasons, because the growing complexity of interrelated and automatised work makes the purposes of an individual's work opaque to him, thus impeding identification with one's work, and effective and pertinent learning. Alienation from daily work by increasing opaqueness of reasons and purposes of one's work tends to impede learning at work.

Mathematics and mathematical structures are ONE possibility of reintroducing an understanding of the work in progress and understanding of technological and organisational change. Insofar mathematics can be even more important for (technical/vocational) education in the future than it is now.

3. The Interplay of Mathematics, Technology, Organisation and Education

For the third remark we have to go into details of the introduction of NC-machines in industry: At least in the Federal Republic of Germany this introduction shows two quite different patterns:

a) The preparation of the programs needed to operate the new machines (e.g. a turning lathe) is done in laboratories and/or offices by engineers. The operator working at the machine has only executive and supervisory functions. Flexibility and mathematical knowledge is not needed for such an operator's job.

b) Sketches and/or technical drawings of the product to be made are given to qualified workers who do their own programming/optimasation of programs on the shop floor at the CNC-machine. They have to execute and supervise production. These workers have to know e.g. about coordinate geometry in three dimensions in order to do their programming.

Both patterns do occur in industry in the Federal Republic of Germany and obviously lead to controversial demands on mathematics education. Some even interprete the case the other way round: Sufficient mathematics education is a prerequisit of pattern b) and education is the active part in this interrelation. To put it in a more cautious formulation: technology is not the unique agent which determines education - nor is it the discipline mathematics; but an interplay of technology, organisation of work, mathematics as a discipline and education seems to be the correct description of what is happening - even if we look into the so-called needs of society and mathematics education /for a broader look on education and economy cf. LUTZ/.

4. Remark

As a final remark I want to point to a limitation of what is usually written about computers and their influence on mathematics and education: Statements usually apply only for developed, industrialised countries - but forget the problems of mathematics education in rural and developing countries. A controversial discussion in working group 3 of the ICMI-conference in Strasbourg showed that there is a lot of research and development to be done in order to cope with "the influence of computers and informatics on mathematics and its teaching" in developing countries.

5. References

BAETHGE, M.: Auswirkungen des Einsatzes der Informationstechnologien auf die Qualifikation und Ausbildung kaufmännischer Angestellter. in: log in, 2 (1982) 15-18

BUSCHHAUS, D.: Auswirkungen der CAD-Technologie auf die Qualifikationsanforderungen in den technisch-zeichnerischen Ausbildungsberufen. in: MEYER, N. - FRIEDRICH, H.R. (ed.): Neue Technologien in der beruflichen Bildung. Köln 1984, 134-139

COCKCROFT-Report (Committee of Inquiry into the Teaching of Mathematics in Schools): Mathematics Counts - Report of the Committee. London (HMSO) 1982

DAWES, W.G. - JESSON, D.St.: Is there a Basis for Specifying the Mathematical Requirements of the 16 Year Old Entering Employment? In: Int.J.Math.Educ.Sci.Technol., 10 (1979) 391-400

ICMI (ed.): The influence of computers and informatics on mathematics and its teaching. in: L'enseignement mathematique, 3o (1984) 159-172

KNOX, C.: Numeracy and School Leavers - a survey of employers' needs. Sheffield (Sheffield Regional Centre for Science and Technology, paper no. 12) May 1977

LUTZ, B.: Die Interdependenz von Bildung und Beschäftigung und das Problem der Erklärung der Bildungsexpansion. in: Mathes, J. (ed.): Sozialer Wandel in Westeuropa. Frankfurt/M. 1979, 634-670

MATHEMATICS in Employment (16-18). Report. Bath (University of Bath) 1981

THE INFLUENCE OF COMPUTERS AND INFORMATICS ON MATHEMATICS

Masaya Yamaguti

§0. I have tried to respond to the question: Did computers change mathematics? But at every period of our history, mathematics was always changing. Then it is hard to point out what part was really changed by computers and informatics. Of course, several new fields are newly created because of computers. I feel the influence on mathematics is something much more than this. The question should be modified as follows: Did computers change the structure of the evolution of mathematical study? To this question I can reply "yes".

§1. How have computers changed or are changing the evolution of all concepts in mathematics?

That is the content of §1 of our report "The influence of computers and informatics on mathematics and its teaching". But here, I would like to emphasize "The integrating effect of different fields of mathematics". Computers and computer experimentation have had the effect of integrating different specialities in mathematics.

From the middle of our century, different specialities of mathematics have continued to differentiate. But this tendency is now stopping because of the influence of computers.

i) One of the most famous examples is the study of non-linear dispersive wave propagation, particularly the soliton solution of the K-d-V equation was discovered as a result of numerical experimentations carried out by Kruskal, Zabusky and Miura. But after this finding, it became more and more an object of pure mathematical study and now, we have some very rigorous mathematical arguments for the whole structure of exact solutions for there kinds of equations via Kac-Moody algebra.

The origin of this series of researches was the visualizations performed by computers.

The merit of this kind of visualization is enormous, because by the visualization of some possible facts, mathematicians in many different fields have a common avenue of research. This causes a recovery of the unity of mathematics.

ii) Let me explain another example which comes from my own experiences. Almost 3 years ago, I observed a graph of the Weierstrass nowhere differentiable continuous function computed by a computer. At the moment, I was engaged in the study of chaotic dynamical systems. I knew that the general solution of the famous discrete dynamical system:

$$x_{n+1} = 4x_n(1 - x_n)$$

is written explicitly by the elementary function of n and the initial value x_0. I felt that some relation existed between the Weierstrass function and this chaotic dynamical system. In reality, it was true that the Weierstrass function is a superposition of these general solutions of the above dynamical system. And then I began a collaborative work with M. Hata on the Takagi function which is also a nowhere differentiable continuous function found by T. Takagi just after Weierstrass. This function is easier to define and it has a clear explanation using the dynamical system:

$$x_{n+1} = \psi(x_n), \quad \psi(x) = \begin{cases} 2x & (0 \le x \le \frac{1}{2}) \\ 2(1-x) & (\frac{1}{2} \le x \le 1) \end{cases}$$

The Takagi function $T(x)$ is now defined by the following expansion:

$$T(x) = \sum_{n=1}^{\infty} \frac{1}{2^n} \psi^n(x) ,$$

where ψ^n is the n-th iterate of ψ. Now we have several general functional equations which have such functions as their solution.

Little later, we remarked that these functional equations had been studied by Julia, de Rham, and Moser without using dynamical systems. But we also become aware of the related researches done in the early period of this century, by Cesaro, Faber, Lebesgue, etc, through several works by de Rham. Finally, we found finite difference schemes (multigrid) which were satisfied by these singular functions (See [1]). By this method, we succeeded in obtaining nice relation between the Takagi function and Lebesgue's singular functions. The next step is the introduction of the concept "Invariant set of contractions" which was proposed by R. F. Williams and rediscovered by J. E. Hutchinson [7] and Hata. This method to describe a function is very simple and useful. It contains all works done at the end of last century, those of Peano, Hilbert, Polya, Sierpinski, Osgood, ... about singular curves. Moreover, it contains many new figures which are kind of Fractals (See [2]). Thus we are now seeing the birth of a new analysis which does not use differential calculus. All our researches was guided by computer experiments before our proof on each above step.

§2. Discrete Mathematics

In 1860, G. Boole wrote in his text of calculus of finite differences (Treatise on Calculus of Finite Differences) that differential calculus and finite difference calculus are two completely different sciences even if one considers the cases when the mesh length tends to zero. And he wrote that this distinction arises in the study of non-linear problems. He gave as an example, the famous Caliraut differential equation, where discretization of the Clairaut differential equation has a family of discretized solutions which tend to a smooth function different from any solution of the original equation as their mesh size tends to zero.

Sometimes, it is said that difference equations are obviously technically and intellectualy much simpler than their counterparts among differential equations and that therefore, we can replace differential equations with difference equations and establish a complete analogy between these two methods.

I am very much doubt these assertions. My opinion is: The recent progress of computers has revealed that these easy arguments are not true, and that what was said by G. Boole is now very important.

Let me explain a little about that. In 1973, Robert May had observed that a very conventional discretization of the logistic differential equation has the solution of the initial value problem which is chaotic and completely different from the exact solution of the differential equation [4]. This research provided a starting point to the study of chaotic discrete dynamical systems. As for myself, I had proved with Matano that for any ordinary differential equation of the type

$$\frac{dy}{dt} = f(y)$$

which has at least 2 equilibrium points and one of them asymptotically stable, the Euler's finite difference scheme of this equation becomes a chaotic discrete dynamical system in the sense of Li-Yorke for fairly large mesh sizes [5]. A little later, we observed that the centered difference scheme of the logistic differential equation exhibits the same phenomena for any mesh size! S. Ushiki completely proved this [6].

Thus, we can say in the case of non-linear differential equations, that the asymptotic behavior of the solutions of differential equations and the solutions of the discretization of then are completely different. One can not replace the other.

Of course, one can construct a complete analogue discretization for differential equations. But for that, it is necessary to have a very sophisticated difference scheme. For example, R. Hirota used a discretization which is very similar to the heat equation:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} .$$

It has the following form:

$$\Delta_t \Pi_x^2 u(x, t) = \Pi_t \Delta_x^2 u(x, t)$$

where Δ is the centered difference operator $\frac{T^+ - T^-}{2}$, Π is the averaging operator $\frac{T^+ - T^-}{2}$, and T_x^+, T_x^- are one mesh shift operators, that is,

$$T_x^+ f(x) = f(x+h) , \quad T_x^- f(x) = f(x-h) .$$

This finite difference scheme conserves the L_1 norm of the initial data as the original equation and the L_2 norm in x of the solution dissipates as t increases. And then Hirota developed a calculus of difference operators which is complete analogous to differential calculus using in a very sophisticated way the operators Δ and Π. For example, he found a formula of chain rules for this finite difference calculus and also complete analogues of many special functions. This way of constructing a new analysis is seemingly very difficult. I can not recommend teaching it to freshmen.

Now, I can propose one thing for mathematical education. My proposal is not to reduce elementary education on continuous theory, for example, that of differential equations. However, I recommend preliminary numerical experiments with some discrete model before introducing differential equations, because the students who learn the difficulties of a discrete theory can understand well the easiness of continuous theroy.

REFERENCES

[1] YAMAGUTI, Masaya and HATA, Masayoshi: On some multigrid finite difference schemes which describe everywhere non differentiable functions.

[1] YAMAGUTI, Masaya and HATA, Masayoshi: Weierstrass's function and chaos. Hokkaido Mathematical J. XII(1983), 333-342.

[1] HATA, Masayoshi and YAMAGUTI, Masaya: The Takagi Function and Its Generalization. Japan J. Appl. Math. 1(1984), 183-199.

[2] YAMAGUTI, Masaya and HATA, Masayoshi: Singular Functions and Finite Difference Schemes.

[3] TOFFOLI, Tommaso: Cellular Automata as an alternative to (rather than an approximation of) Differential Equations in Modeling Physics. Physica 10D(1984), 117-127.

[4] MAY, Robert M.: Simple mathematical models with very complicated dynamics. Nature 261(1976), 459-467.

[5] MATANO, Hiroshi and YAMAGUTI, Masaya: Euler's finite difference scheme and chaos. Proc. Japan Acad. 55, Ser. A(1979), 78-80.

[6] USHIKI, Shigehiro: Central difference scheme and chaos, Physica 4D(1982), 407-424.

[7] J. E. Hutchinson, Fractals and Selfsimilarity Indiana. Math. J. 30(1981), 713-747.

Living with a New Mathematical Species

Lynn Arthur Steen
Department of Mathematics
St. Olaf College
Northfield, Minn. 55077
April 15, 1985

The history of mathematics can be viewed as a counterpoint between the finite and the infinite, between the discrete and the continuous. Although rooted in geometry, Greek mathematics was primarily finite, concrete, and specific. Modern mathematics, in contrast, is infinite, abstract, and general. Aristotle inveighed against the actual infinite, reflecting the Greek cultural distaste for the incomplete form. Centuries later, Leibniz and Newton overcame Aristotelean scruples in proposing methods of calculating with infinitesimals. Now, after three hundred years of Newtonian mathematics, computers are forcing a return to mathematical preferences of the pre-Newtonian age--to the finite, the specific, and the concrete.

This return to our roots is a natural consequence of increasing mathematical maturity. Weierstrass resolved the paradox of infinitesimals by reducing analysis to arithmetic; he showed how to interpret the infinite concepts of calculus in terms of the finite structures of arithmetic. Twentieth century mathematics has been dominated by the Weierstrass synthesis --a working intellectual compromise between the finite limitations of human mental processes and the infinite visions of human imagination.

Today we are forging a new compromise--or in Thomas Kuhn's terms, a new paradigm--binding computers with mathematics. Computers are both the creature and the creator of mathematics. They are, in the apt phrase of Seymour Papert, "mathematics-speaking beings." More recently J. David Bolter in his stimulating book Turing's Man [4] calls computers "embodied mathematics." Computers restore the specific and concrete to the ethereal world of mathematics, yet their very existence depends in crucial ways on the abstract and the theoretical. Although computers would never have been invented without the theoretical support of abstract, continuous Newtonian mathematics, both computer architecture and computer science depend primarily on finite and discrete methodology. Bolter describes the situation more colorfully: "The computer specialist has as little use for irrational numbers as the Pythagoreans had" [4, p. 64].

Computers shape and enhance the power of mathematics, while mathematics shapes and enhances the power of computers. Each forces the other to grow and change. Despite the weight of tradition, mathematics curricula and pedagogy must also change to reflect this new reality.

The Ecology of Mathematics

Until recently, mathematics was a strictly human endeavor. It evolved with human society, achieving a degree of universality equalled by few other aspects of human culture. Its ecology was a human ecology, linked closely to science and language, evolving as human science and language changed.

But suddenly, in a brief instant on the time scale of mathematics, a new species has entered the mathematical ecosystem. Computers speak mathematics, but in a dialect that is difficult for some humans to understand. Their number systems are finite rather than infinite; their addition is not commutative; and they don't really understand "zero", not to speak of "infinity". Nonetheless, they do embody mathematics.

Many features of the new computer mathematics appear superficial: notation such as ^ and ** for exponentiation, linearized expressions for formulas traditionally represented by a two-dimensional layout, a preference for binary, octal, or hexadecimal representations of numbers, and in early languages a new action-oriented meaning to the "equals" sign. Some variances are more significant, and more difficult to assimilate into traditional mathematics: finite number systems, interval arithmetic, roundoff errors, computational intractability.

As mathematics goes, linguistic and notational changes are truly superficial—it really is the same subject modulo an isomorphism. These differences can be very confusing to students learning mathematics and computing, although perhaps no more so than the differences in vocabulary and perspective between an engineer and a mathematician. The blend of computer language and traditional mathematics produces a kind of Franglais decried by purists yet employed by everyone.

The core of mathematics, however, is also changing under the ecological onslaught of mathematics-speaking computers. New specialties in computational complexity, theory of algorithms, graph theory, and formal logic attest to the impact that computing is having on mathematical research. As Arthur Jaffe has argued so well in his recent essay "Ordering the Universe" [12], the computer revolution is a mathematical revolution. The intruder has changed the ecosystem of mathematics, profoundly and permanently.

New Mathematics for a New Age

Computers are discrete, finite machines. Unlike a Turing machine with an infinite tape, real machines have limits of both time and space. Theirs is not an idealistic Platonic mathematics, but a mathematics of limited resources. The goal is not just to get a result, but to get the best result for the least effort. Optimization, efficiency, speed, productivity--these are essential objectives of modern computer mathematics. Questions of optimization lead to the study of graphs, of operations research, of computational complexity.

Computers are also logic machines. They embody the fundamental engine of mathematics—rigorous propositional calculus. So it comes as no surprise that computer programs can become full partners in the process of mathematical proof. The first celebrated computer proof was that of the four-color theorem: the computer served there as a sophisticated accountant, checking out thousands of cases of reductions. Despite philosophical alarms that computer-based proofs change mathematics from an a priori to a contingent, fallible subject (see, e.g., [27]), careful analysis reveals that nothing much had really changed. The human practice of mathematics has always been fallible; now it had a partner in fallibility.

Recent work on the mysterious Feigenbaum constant reveals just how far this evolution has progressed in just eight years: Computer-assisted investigations of families of periodic maps suggested the presence of a mysterious universal limit, apparently independent of the particular family of maps. Subsequent theoretical investigations led to proofs that are true hybrids of classical analysis and computer programming: the crucial step in a fixed-point argument requires a tight estimate on the norm of a high degree polynomial. This estimate is made by a computer program, carefully crafted using interval arithmetic to account in advance for all possible inaccuracies introduced by roundoff error [8]. Thus computer assisted proofs are possible not just in graph theory, but also in that bastion of classical mathematics--functional analysis.

Computers are also computing machines. By absorbing, transforming, and summarizing massive quantities of data, computers can simulate reality. No longer need the scientist build an elaborate wind tunnel or a scale model refinery in order to test engineering designs. Wherever basic science is well understood, computer models can emulate physical processes by carrying out instead the process implied by mathematical equations. Mathematical models used to be primarily tools used by theoretical scientists to formulate general theories; now they are practical tools of enormous value in the everyday world of engineering and economics. They focus mathematical attention on the relation between data and theory, on stochastic processes and differential equations, on data analysis and mathematical statistics.

In many respects mathematics has become the creature of the computer: by providing compelling tools in combinatorics, logic, and calculation, the computer has made an offer of intellectual adventure that mathematicians cannot refuse. But in a very real sense, mathematics is also the creator of the computer. David Hilbert's struggle with the foundations of mathematics--itself precipitated by the paradoxes of set theory elucidated by Frege and Russell--led directly to Alan Turing's proposal for a universal machine of mathematics:

> [Turing] proved that there was no 'miraculous machine' that could solve all mathematical problems, but in the process he had discovered something almost equally miraculous, the idea of a universal machine that could take over the work of any machine. He argued that anything performed by a human computer could be done by a machine. [11, p. 109]

It has been fifty years precisely since Turing developed his scheme of computability [26] in which he argued that machines could do whatever humans might hope to do. His was a formal, abstract system, devoid of hardware and real machines. It took 25 years for rudimentary machines to demonstrate in a productive way the genius of Turing's idea

During that same period abstract mathematics flourished, led by Bourbaki, symbolized by the "generalized abstract nonsense" of category theory. But with abstraction came power, with rigor came certainty. Once real computers emerged, the complexity of programs quickly overwhelmed the informal techniques of backyard programmers. Formal methods became _de rigueur_; even the once-maligned category theory was enlisted to represent finite automata and recursive functions:

> A quite formalistic approach is now both feasible and desirable, and nowhere is the transition of programming from art to science made more evident. One result of this more formal, disciplined approach ... is a sharp reduction in the programming effort needed to implement a compiler. [2, p.423]

Once again, as happened before with physics, mathematics became more efficacious by becoming more abstract.

The Core of the Curriculum

The circumstances that make computing a force for rapid evolution in the notation and practice of mathematics also put pressure on the mathematics curriculum in colleges and universities. The presence of a new and vigorous subject such as computer science produces enormous strains on faculty, curriculum, and resources. As different ecosystems respond in different ways to the presence of a new predator, so different institutions are responding in different ways to the incursion of computer science into the undergraduate curriculum.

Twenty years ago in the United States the Committee on the Undergraduate Program in Mathematics (CUPM) issued a series of reports that led to a gradual standardization of curricula among undergraduate mathematics departments [5]. Following two years of calculus and linear algebra, students took core courses in real analysis and abstract algebra (usually two apiece) and selected electives from among such options as topology, differential equations, geometry, complex analysis, number theory, probability, and mathematical statistics. While the faculty expectations and student performance on these courses varied greatly from institution to institution, consensus on a central core was always present.

The subsequent decade was good to mathematics education. The number of bachelor's degrees in the United States rose to about 25,000; the number of Ph.D.s, rose gradually from the low hundreds to over 1200. But while core mathematics was experiencing a renaissance, those exploring the frontiers detected evidence of coming change.

In 1971 Garrett Birkhoff and J. Barkley Rosser presented papers at a meeting of the Mathematical Association of America concerning their predictions for undergraduate mathematics in 1984. Birkhoff urged increased emphasis on modelling, numerical algebra, scientific computing, and discrete mathematics ("a course introduced over 10 years ago at Harvard by Howard Aiken while director of our computation laboratory"). He also advocated increased use of computer methods in pure mathematics:

> To my mind the use of computers is analogous to the use of logarithm tables, tables of integrals, ... or carefully drawn figures. Far from muddying the limpid waters of clear mathematical thinking, they make them more transparent by filtering out most of the messy drudgery which would otherwise accompany the working out of specific illustrations. Moreover, they give a much more adequate idea of the range to which the ideas expressed are applicable than could be given by a purely deductive general discussion unaccompanied by carefully worked out examples.
>
> Therefore, I believe that [computer-based] courses should be considered as basic courses in <u>pure</u> mathematics, to be taken by all students wishing to understand the power (and limitations) of mathematical methods. [3, p. 651]

Rosser emphasized many of the same points, and warned of impending disaster to undergraduate mathematics if their advice went unheeded:

> Unless we revise the calculus course and the differential equations course and probably the linear algebra course...so as to embody much use of computers, most of the clientele for these courses will instead be taking computer courses in 1984. ... If students cannot acquire the necessary computer proficiency and understanding in their mathematics courses, they will have no choice but to take computer courses instead. [21, p. 639]

In the decade since these words were written, U. S. undergraduate and graduate degrees in mathematics have declined by 50%. New courses in modelling, discrete mathematics and data analysis are emerging in every college and university. The clientele for traditional mathematics has indeed migrated to computer science. The former CUPM consensus is all but shattered.

The symbol of reformation has become discrete mathematics. Several years ago Anthony Ralston argued forcefully the need for change before both the mathematics community [17] and the computer science community [18]. Discrete mathematics, in Ralston's view, is the central link between the fields. College mathematics must introduce discrete methods early and in depth; computer science curricula must, in turn, require and utilize mathematical concepts and techniques. The advocacy of discrete mathematics rapidly became quite vigorous (see, e.g., [19] and [24]), and the Sloan Foundation funded experimental curricula at six institutions to encourage development of discrete-based alternatives to standard freshman calculus.

Five years ago CUPM issued a new report, this one on the Undergraduate Program in Mathematical Sciences [6]. Beyond calculus and linear algebra, they could agree on no specific content for the core of a mathematics major: "There is no longer a common body of pure mathematical information that every student should know. Rather, a department's program must be tailored according to its perception of its role and the needs of its students." The committee did agree that students need to learn to think mathematically and to study some mathematical subject in depth. But they could not agree, for example, that every mathematics major needs to know real analysis, or group theory, or any other topic formerly part of the advanced core of the major.

The niche of mathematics in the university ecosystem has been radically transformed by the presence of computer science in the undergraduate curriculum. As each institution reacts to particular local pressures of staff resources and curriculum tradition, the undergraduate mathematics major has disintegrated into countless local varieties.

Despite the pressure for radical change, the momentum of tradition still permits the strongest mathematics departments to continue the traditional CUPM major for a declining number of students. Reduced enrollment does make it difficult, however, to provide advanced core mathematics courses on a regular basis. In larger institutions, computer science operates as a parallel program, almost always attracting large enrollments, including some of the best and brightest students on campus. It is not uncommon for undergraduate majors in computer science to outnumber mathematics majors by ratios of 20:1 or more.

At smaller institutions a different pattern has emerged. Many such departments have been forced to drop regular offerings of such former core courses as topology, analysis and algebra. Where resources do not permit full majors in mathematics and computer science, the mathematics program often becomes a hybrid major consisting of some computer science some mathematics, and some statistics--introductions to everything, mastery of nothing.

The need for consensus on the contents of undergraduate mathematics is perhaps the most important issue facing American college and university mathematics departments. On the one hand departments that have a strong traditional major often fail to provide their students with the robust background required to survive the evolutionary turmoil in the mathematical sciences. Like the Giant Panda, they depend for survival on a dwindling supply of bamboo--strong students interested in pure mathematics. On the other hand, departments offering flabby composite majors run a different risk: by avoiding advanced, abstract requirements, they often misrepresent the true source of mathematical knowledge and power. Like zoo-bred animals unable to forage in the wild, students who have never been required to master a deep theorem are ill-equipped to master the significant theoretical complications of real-world computing and mathematics.

Computer Literacy

Mathematical scientists at American institutions of higher education are responsible not only for the technical training of future scientists and engineers, but also for the technological literacy of laymen--of future lawyers, politicians, doctors, educators, and clergy. Public demand that college graduates be prepared to live and work in a computer age has caused many institutions to introduce requirements in quantitative or computer literacy. Many educators are calling for a total reform of liberal education.

In 1981 Stephen White, a program officer with the Alfred P. Sloan foundation, initiated debate on the proper role of applied mathematics and computer experience in the education of students outside the technical fields. He termed these "the new liberal arts:"

> The ability to cast one's thoughts in a form that makes possible mathematical manipulation and to perform that manipulation, coupled with the fruits of that analysis, are modes of thought. ... In making use of those modes of thought one may think with enormous new efficiency. But it is thinking itself that is the creative element: thoughtless modelling and thoughtless computation, impressive as they may be, are devoid of real significance. ... It is precisely as modes of thought that they become essential in higher education, and above all in liberal education [14, p. 6].

Others echoed this call for reform of liberal education. David Saxon, President of the University of California wrote in a Science editorial that liberal education "will continue to be a failed idea as along as our students are shut out from, or only superficially acquainted with, knowledge of the kinds of questions science can answer and those it cannot" [22].

Too often these days the general public views computer literacy as the appropriate modern substitute for mathematical knowledge. Unfortunately, this often leads students to superficial courses that emphasize vocabulary and experiences over concepts and principles. The advocates of computer literacy conjure images of an electronic society dominated by the information industries. Their slogan of "literacy" echoes traditional educational values, conferring the aura but not the logic of legitimacy.

Typical courses in computer literacy, however, are filled with ephemeral details whose intellectual life will barely survive the students' school years. A best selling textbook in the United States for courses introducing computing to nonspecialists is full of glossy color pictures, but does not even mention the word "algorithm." These courses contain neither a Shakespeare nor a Newton, neither a Faulkner nor a Darwin; they convey no fundamental principles nor enduring truths.

Computer literacy is more like driver education than like calculus. It teaches students the prevailing rules of the road concerning computers: how to create and save files, how to use word processors and spreadsheets, how to program in Basic. One can be confident only that most students finishing such a course will not injure themselves or others in their first encounter with a real computer in the workplace. But such courses do not leave students well prepared for a lifetime of work in the information age.

Algorithms and data structures are to computer science what functions and matrices are to mathematics. As much of the traditional mathematics curriculum is devoted to elementary functions and matrices, so beginning courses in computing--by whatever name--should stress standard algorithms and typical data structures.

For example, as early as students study linear equations they could also learn about stacks and queues; when they move on to conic sections and quadratic equations, they could in a parallel course investigate linked lists and binary trees. The algorithms for sorting and searching, while not part of traditional mathematics, convey the power of abstract ideas in diverse applications every bit as much as do conic sections or derivatives.

Computer languages can (and should) be studied for the concepts they represent--procedures in Pascal, recursion and lists for Lisp--rather than for the syntactic details of semicolons and line numbers. They should not be undersold as mere technical devices for encoding problems for a dumb machine, nor oversold as exemplars of a new form of literacy. Computer languages are not modern equivalents of Latin or French; they do not deal in nuance and emotion, nor are they capable of persuasion, conviction, or humor. Although computer languages do represent a new and powerful way to think about problems, they are not a new form of literacy.

As computer science joins mathematics as a basic ingredient in secondary and higher education, liberal education must move beyond computer literacy. As mathematics employs the abstractions of algebra and geometry as tools for problem solving, so courses in computing must incorporate the abstract structures of computer science--algorithms, data structures--in a pragmatic, problem-solving environment. Such computer principles, firmly rooted in mathematics, are a legitimate and important component of the school and college curriculum for all students.

Computer Science

The confusion evident in university mathematics departments is an order of magnitude less severe than that which operates in university computer science programs. In the United States, these programs cover an enormous spectrum, from business-oriented data processing curricula, through management information science, to theoretical computer science. All of these intersect with the mathematics curriculum, each in different ways. The computer science community is now struggling with this chaos, and has a process in place for identifying exemplar programs of different types as a first step towards an accreditation system for college computer science departments.

Several computer science curricula have been developed by the professional societies ACM and IEEE, for both large universities and small colleges. Recently Mary Shaw of Carnegie Mellon University put together an excellent composite report on the undergraduate computer science curriculum at CMU, surely one of the very best available anywhere. This report is quite forceful about the contribution mathematics makes to the study of computer science:

> The most important contribution a mathematics curriculum can make to computer science is the one least likely to be encapsulated as an individual course: a deep appreciation of the modes of thought that characterize mathematics. We distinguish here two elements of mathematical thinking that are also crucial to computer science...the dual techniques of abstraction and realization and of problem-solving. [23. p. 55]

The converse is equally true: one of the more important contributions that computer science can make to the study of mathematics is to develop in students an appreciation for the power of abstract methods when applied to concrete situations. Students of traditional mathematics used to study a subject called "Real and Abstract Analysis"; students of computer science now can take a course titled "Real and Abstract Machines." In the former "new math", as well as in modern algebra, students learned about relations, abstract versions of functions; today business students study "relational data structures" in their computer courses, and advertisers tout "fully relational" as the latest innovation in business software. The abstract theories of finite state machines and deterministic automata are reflections in the mirror of computer science of well established mathematical structures from abstract algebra and mathematical logic.

An interesting and pedagogically attractive example of the power of abstraction made concrete can be seen in the popular electronic spreadsheets that are marketed under such trade names as Lotus and VisiCalc. Originally designed for accounting, they can as well emulate cellular automata or the Ising model for ferromagnetic materials [10]. They can also be "programmed" to carry out most standard mathematical algorithms--the Euclidean algorithm, the simplex method, Euler's method for solving differential equations [1]. An electronic spreadsheet--the archetype of applied computing--is a structured form for recursive procedures--the fundamental tool of algorithmic mathematics. It is a realization of abstract mathematics, and carries with it much of the power and versatility of mathematics.

Computers in the Classroom

Computers are mathematics machines, as calculators are arithmetic machines. Just as the introduction of calculators upset the comfortable paradigm of primary school arithmetic, so the spread of sophisticated computers will upset the centuries old-tradition of college and university mathematics. This year long division is passe; next year integration will be under attack.

Reactions to machines in the mathematics classroom are entirely predictable. Committee oracles and curriculum visionaries proclaim a utopia in which students concentrate on problem solving and machines perform the mindless calculations (long division and integration). Yet many teachers, secure in their authoritarian rule-dominated world, banish calculators (and computers) from ordinary mathematics instruction, using them if at all for separate curricular units where different ground rules apply. The recent International Assessment of Mathematics documented that in the United States calculators are never permitted in one-third of the 8th grade classes, and rarely used in all but 5% of the rest [25, p. 18].

The large gap between theory and practice in the use of computers and calculators for mathematics instruction is due in part to a pedagogical assumption that pits teacher against machine. If the teacher's role is to help (or force) students to learn the rules of arithmetic (or calculus), then any machine that makes such learning unnecessary is more a threat than an aid. Debates continue without end: Should calculators be used on exams? Should we expect less mastery of complex algorithms like long division or integration? Will diminished practice with computation undermine subsequent courses that require these skills?

The impact of computing on secondary school mathematics has been the subject of many recent discussions in the United States. Jim Fey, coordinator of two of the most recent assessments ([7], [9]), described these efforts as

> an unequivocal dissent from the spirit and substance of efforts to improve school mathematics that seek broad agreement on conservative curricula. Many mathematics educators working with emerging electronic technology see neither stability nor consensus in the future of school mathematics. [9, p. vii]

The technology wars are just beginning to spread to the college classroom. Lap size computers are now common--they cost about as much as ten textbooks, but take up only the space of one. Herb Wilf argues (in [28]) that it is only a matter of time before students will carry with them a device to perform all the algorithms of undergraduate mathematics. Richard Rand, in a survey [20] of applied research based on symbolic algebra agrees: "[Computer algebra] is virtually absent from undergraduate education in the sciences and engineering. ... however, it is destined for a major role in engineering and applied mathematics. It will not be long before computer algebra is as common to engineering students as the now obsolete slide rule once was."

John Kemeny tells a story (in [13]) about calculus instruction that sheds interesting new light on the debate about manipulating symbols. He asks for the value of $\int_0^{13} e^x\,dx$. A moment's thought reveals the answer to be $e^{13}-1$. That's the exact answer. Kemeny's first question is this: what is its value to <u>one</u> significant digit? With just paper and pencil, that's hard to do--beyond the likely skills of typical calculus students. (The answer: 400,000.) Now comes the second question: what's the difference between the original question and the traditional exact answer? They are both exact expressions for the value we seek, equally unenlightening. So the proper question is not to find an exact value, but to choose which of many possible exact values is more suitable to the purpose at hand.

The challenges of computers in the classroom are exactly analogous to those of calculators. The computer will do for the teaching of calculus algorithms just what calculators did for arithmetic computations--it will make them redundant. In so doing, it will challenge rigid teachers to find new way to reassert authority. Good teachers, however, should respond to the computer as a blessing in disguise--as a <u>deus ex machina</u> to rescue teaching from the morass of rules and templates that generations of texts and tests have produced.

<u>Following the Rules</u>

Mathematics students, perhaps more than other students, like to get correct answers. Computers, for the most part, reinforce the student's desire for answers. Their school uses have been largely extensions of the old "teaching machines": programmed drill with pre-determined branches for all possible answers, right or wrong. In colleges and universities, computers are still used most often as black-box calculators, spewing out numbers in answer to questions both asked and unasked.

Core mathematics courses continue this long-standing tradition, reinforcing the emphasis on rules and answers. Traditional calculus textbooks bear an uncanny resemblance to the first calculus text ever published: l'Hôpital's 1699 classic. They present rules of differentiation and integration, with or without proof: linearity, product and quotient rules, chain rule, substitution, etc. After each rule are exercises to practice on. At the end of the chapter are mixed exercises, where the challenge is to use all the rules at the same time.

Most students of even modest ability can master these rules. If there is one thing that school does well, it is to train students to learn rules. Strong students master them quickly, and yearn for tough problems that extend the rules (e.g., to x^x). Weak students work by rote, carefully adhering to template examples. Students of all types flounder when presented with "word problems" with which to "apply" their skills: "A farmer has 200 meters of fence with which to...." Too often such problems are merely mathematical crossword puzzles--stylized enigmas whose solutions depend in large part on recognizing the unstated problem pattern. Indeed, recent research in problem solving suggests that many students learn to solve such problems by establishing mental categories of problem-type, and of course many instructors teach students to identify such types.

The confluence of research on learning with symbolic algebra has produced a rich new territory for imaginative pedagogy. Symbolic algebra packages linked to so-called "expert systems" on computers of sufficient power (with high resolution graphics, mouse-like pointers, and multiple windows) can provide an effective intelligent tutor for learning algebraic skills. Computers can manipulate algebraic and numerical expressions as well as students can, usually better. They cannot however recognize, parse, or model a word problem except in the narrowest sense--by matching templates to canonical forms.

It is commonplace now to debate the value of teaching skills such as differentiation that computers can do as well or better than humans. Is it really worth spending one month of every year teaching half of a country's 18 year old students how to imitate a computer? What is not yet so common is to examine critically the effect of applying to mathematics pedagogy computer systems that are only capable of following rules or matching templates. Is it really worth the time and resources to devise sophisticated computer systems to teach efficiently precisely those skills that computers can do better than humans, particularly those skills that make the computer tutor possible? The basic question is this: since computers can now do algebra and calculus algorithms, should we use this power to reduce the curricular emphasis on calculations or as a means of teaching calculations more efficiently? This is a new question, with a very old answer.

<u>Let Us Teach Guessing</u>

35 years ago George Polya wrote a brief paper with the memorable title "Let Us Teach Guessing" [16]. Too few of us actually do that: most teachers, the overwhelming number, are authoritarian. Teachers set the problems; students solve them. Good students soon learn that the key to school mathematics is to discern the right answer; poor students soon give up.

But Polya says: let us teach guessing. It is not differentiation that our students need to learn, but the art of guessing. A month spent learning the rules of differentiation reinforces a student's ability to learn (and live by) the rules. It also, almost incidentally, teaches a computational skill of diminishing scientific value. In contrast, time spent making conjectures about derivatives will teach a student something about the art of mathematics and the science of order, in the context of a useful but increasingly unnecessary computational skill.

Imagine a class with access to a good symbolic calculus package. Instead of providing rules for differentiation and exercises to match, the instructor can give motivational lectures replete with physical and geometric interpretation of the derivative. The homework can begin with exploratory questions: ask the computer for the derivative of simple functions. Make conjectures and try them on the machine. After mastering linear functions, try products, then exponentials. Make conjectures; test them out.

The class can discuss their conjectures. Most will be right; a few will not be. Discussion will readily elicit counterexamples, and some informal proofs. With the aid of the mathematics-speaking computer, students can for the first time learn college mathematics by discovery. This is an opportunity for pedagogy that mathematics educators cannot afford to pass up.

Mathematics is, after all, the science of order and pattern, not just a mechanism for grinding out formulas. Students discovering mathematics gain insight into the discovery of pattern, and slowly build confidence in their own ability to understand mathematics. Formerly, only students of sufficient genius to forge ahead on their own could have the experience of discovery. Now with computers as an aid, the majority of students can experience for themselves the joy of discovery. Only when the computer is used as an instrument of discovery will it truly aid the learning of mathematics.

<u>Metaphors for Mathematics</u>

Two metaphors from science are useful for understanding the relation between computer science, mathematics, and education. Cosmologists long debated two theories for the origin of the universe--the Big Bang theory, and the theory of Continuous Creation. Current evidence tilts the cosmology debate in favor of the Big Bang. Unfortunately, this is all too often the public image of mathematics as well, even though in mathematics the evidence favors Continuous Creation.

The impact of computer science on mathematics and of mathematics on computer science is the most powerful evidence available to beginning students that mathematics is not just the product of an original Euclidean big bang, but is continually created in response to challenges both internal and external. Students today, even beginning students, can learn things that were simply not known 20 years ago. We must not only teach new mathematics and new computer science, but we must teach as well the fact that this mathematics and computer science is new. That's a very important lesson for laymen to learn.

The other apt metaphor for mathematics comes from the history of the theory of evolution. Prior to Darwin, the educated public believed that forms of life were static, just as the educated public of today assumes that the forms of mathematics are static, laid down by Euclid, Newton and Einstein. Students learning mathematics from contemporary textbooks are like the pupils of Linnaeus, the great eighteenth century Swedish botanist: they see a static, pre-Darwinian discipline that is neither growing nor evolving. Learning mathematics for most students is an exercise in classification and memorization, in labelling notations, definitions, theorems, and techniques that are laid out in textbooks as so much flora in a wonderous if somewhat abstract Platonic universe.

Students rarely realize that mathematics continually evolves in response to both internal and external pressures. Notations change; conjectures emerge; theorems are proved; counterexamples are discovered. Indeed, the passion for intellectual order combined with the pressure of new problems--especially those posed by the computer--force researchers to continually create new mathematics and archive old theories.

Until recently, mathematics evolved so slowly and in such remote frontiers that students in elementary courses never noticed it. The presence of computers in the mathematical ecosystem has changed all that: evolution of theories and notation now takes place rapidly, and in contexts that touch the daily lives of many students. Mathematics itself is changing in response to this intruding species. So must mathematics curriculum and mathematics pedagogy.

References

1. Arganbright, Dean E. Mathematical Applications of Electronic Spreadsheets. McGraw-Hill, 1985.
2. Beckman, Frank S. Mathematical Foundations of Programming. The Systems Programming Series, Addison Wesley, 1984.
3. Birkhoff, Garrett. "The Impact of Computers on Undergraduate Mathematics Education in 1984." American Mathematical Monthly 79 (1972) 648-657.
4. Bolter, J. David. Turing's Man: Western Culture in the Computer Age. University of North Carolina Press, Chapel Hill, 1984.
5. Committee on the Undergraduate Program in Mathematics. A General Curriculum in Mathematics for Colleges. Mathematical Association of America, 1965.
6. Committee on the Undergraduate Program in Mathematics. Recommendations for a General Mathematical Sciences Program. Mathematical Association of America, 1980.
7. Corbitt, Mary Kay, and Fey, James T. (Eds.). "The Impact of Computing Technology on School Mathematics: Report of an NCTM Conference." National Council of Teachers of Mathematics, 1985.
8. Eckmann, J.-P., Koch, H., and Wittwer, P. "A computer-assisted proof of universality for area-preserving maps." Memoirs of the American Mathematical Society, Vol 47, No. 289 (Jan. 1984).
9. Fey, James T., et al. (Eds.). Computing and Mathematics: The Impact on Secondary School Curricula. National Council of Teachers of Mathematics, 1984.
10. Hayes, Brian. "Computer Recreations." Scientific American (October 1983) 22-36.
11. Hodges, Andrew. Alan Turing: The Enigma. Simon and Schuster, 1983.
12. Jaffe, Arthur. "Ordering the Universe: The Role of Mathematics." In Renewing U. S. Mathematics, National Academy Press, Washington, D. C. 1984.
13. Kemeny, John G. "Finite Mathematics--Then and Now." In Ralston, Anthony and Young, Gail S. The Future of College Mathematics. Springer-Verlag, 1983, pp. 201-208.
14. Koerner, James D., ed. The New Liberal Arts: An Exchange of Views. Alfred P. Sloan Foundation, 1981.
15. Lewis, Harry R. and Papadimitriou. Elements of the Theory of Computation. Prentice-Hall. 1981.

16. Polya, George. "Let Us Teach Guessing." Etudes de Philosophie des Sciences. Neuchatel: Griffon, 1950, pp. 147-154; reprinted in George Polya: Collected Papers. Vol. IV, MIT Press, 1984, pp. 504-511.

17. Ralston, Anthony. "Computer Science, Mathematics, and the Undergraduate Curricula in Both." American Mathematical Monthly 88 (1981) 472-485.

18. Ralston, Anthony and Shaw, Mary. "Curriculum '78: Is Computer Science Really that Unmathematical?" Communications of the ACM 23 (Feb. 1980) 67-70.

19. Ralston, Anthony and Young, Gail S. The Future of College Mathematics. Springer Verlag, 1983.

20. Rand, Richard H. Computer Algebra in Applied Mathematics: An Introduction to MACSYMA. Research Notes in Mathematics No. 94, Pitman Publ., 1984.

21. Rosser, J. Barkley. "Mathematics Courses in 1984." American Mathematical Monthly 79 (1972) 635-648.

22. Saxon, David S. "Liberal Education in a Technological Age." Science 218 (26 Nov 1982) 845.

23. Shaw, Mary (Ed.) The Carnegie-Mellon Curriculum for Undergraduate Computer Science. Springer Verlag, 1984.

24. Steen, Lynn Arthur. 1 + 1 = 0: New Math for a New Age. Science 225 (7 Sept. 1984) 981.

25. Travers, Kenneth, et. al. Second Study of Mathematics: United States Summary Report. University of Illinois, September 1984.

26. Turing, Alan M. "On Computable Numbers, with an Application to the Entscheidungsproblem." Proc. London Math. Soc. 2nd Ser., 42 (1936) 230-265.

27. Tymoczko, Thomas. "The Four Color Problem and its Philosophical Significance." Journal of Philosophy 76:2 (1979) 57-85.

28. Wilf, Herbert. "The Disk with the College Education." American Mathematical Monthly 89 (1982) 4-8.

COMMUNICATION

Par Mme le Prof. Dr Josèphine GUIDY WANDJA
Universite Nationale
Departement de Mathématiques
ABIDJAN (COTE D'IVOIRE)

CONTRIBUTION AU SYMPOSIUM SUR " THE INFLUENCE OF COMPUTERS AND INFORMATICS ON MATHEMATICS AND ITS TEACHING"
organisé par l'ICMI
Strasbourg 25 - 30 Avril 1985

Introduction

En nous référant à la définition de l'UNESCO," L'Informatique est l'ensemble des disciplines et des techniques de traitement systématique des données et de l'information considérées comme un moyen d'accès au savoir,le but étant la conservation dans le temps et la communication dans l'espace de ce savoir...Dans le conteste actuel,on englobe dans l'informatique les activités de conception,de mise en place,d'evaluation,d'application et de mise à jour des systèmes de traitement ,de stockage et de communication des donnees ,en ce qui concerne tant les matériels que les logiciel,les aspects organisationnels et humains."

Il apparait donc que l'informatique est une science et une technique qui comprend à la fois et de manière indissociable:

- Les moyens de traitement et leur fonctionnement (c'est à dire la technologie des ordinateurs,ses fondements techniques et théoriques ainsi que ses applications)
- les methodes de traitement(c'est à dire tout ce qui est lié à l'utilisation des ordinateurs)
- Les domaines d'applications quasi illimités qui vont des sciences à l'Administration en passant par la commande des opérations industrielle la telecommunication, l'enseignement.

Pendant longtemps en Afrique,l'idee de l'informatique est restée liee à la notion de comptabilité,de gestion des entreprises;cependant l'avènement des microprocesseurs ,en entrainant une chute des coûts,a permis de généraliser l'utilisation de l'informatique. Mais son introduction dans le système educatif soulève encore des questions car l'informatique est à la fois une science et un outil d'enseignement.

Situation actuelle des pays en voie de developpement en Informatique

De façon général,le tiers-monde accuse un retard important dans le domaine de l'informatique.Le tiers-monde n'occupe qu'environ 5% du marche mondial;un deséquilibre informatique qui s'ajoute à d'autres déséquilibres.

En Afrique,sous la double impulsion des constructeurs d'ordinateurs et des sociétés de service et de conseil en Informatique,la plus part des Etatsafricains surtout francophones ont dès 1960 entame la mecanisation des activités de certais départements ministeriels.Des centres informatiques furent creent et en 1971 les chefs d'Etats reunis à Ndjamena (Tchad) décidèrent la création de l'Institut Africain d'Informatique pour la formation des analystes et des programmeurs dont ils ont besoin avec siège à Libreville (Gabon). Depuis l'ouverture de ses portes en Novembre 1971,l'IAI a formé 400 Analystes-programmeurs et une trentaine de programmeurs. Par la suite,d'autres centres furent crées par certains Etats tels que l'ISI (Institut Superieur d'Informatique) en Côte d'Ivoire,l'IUT de Dakar (Senegal),cée avec le concours de l'IBI qui dispose d'un centre régional à Dakar pour une formation en Informatique.

L'informatique représente un intérêt de plus en plus craissant dans les pays en voie de developpement.En effet ,l'utilisation des moyens et des methodes de l'informatique dans les pays developpés a fait de l'informatique un élement important de prise de décision au niveau les plus divers. De plus en plus maitrisée par ces pays, l'informatique apparait comme un instrument indispensable du developpement économique et social et permet de plus en plus de traiter l'information d'une façon qui correspond aux besoins des planificateurs et des responsables dans tous les domaines et à tous les niveaux d'activites. Son intérêt majeur réside dans le raccourcissement des délais de résolution de la plus part des problèmes, les possibilités nouvelles de résolution par les moyens de traitement actuel des problèmes que l'on ne pouvait pas aborder jusque là; la possibilité d'application de l'emploi des ordinateurs et des concepts de base de l'informatique dans pratiquement toutes les branches de la science, de la technique, de l'économie.

La formation

L'intérêt que représente l'informatique dans les pays en voie de developpement doit être accompagnée de la formation du personnel apte à utiliser. L'expension très rapide de l'utilisation des ordinateurs doit être accompagnée de la création et du renforcement des structures pédagogiques. Comme le souligne un rapport du Sécrétariat Général de l'organisation des Nations Unies : " L'acquisition d'un ordinateur peut fort bien ou au départ créer au moins autant de problèmes qu'elle en résoud. Entre les mains d'utilisateurs compétents, les ordinateurs peuvevent être à la fois utiles et efficaces. Mais c'est à des hommes qu'il revient de fixer leur rôle et d'exploiter les possibilites qu'ils offrent Leur efficacite dépend donc de la compétence et des connaissances de ceux qui planifient, approuvent, mettent en oeuvre, dirigent, surveillent et evaluent les activités pour lesquelles on utilise l'ordinateur."

Il apparait donc que la formation con erve une priorité dans tous les pays. L'enseignement a donc un rôle à jouer pour rémédier au manque de personnel spécialisé auquel se heurtent les pays mêmes developpés.

L'Informatique et l'enseignement

L'introduction de l'informatique dans les lyces et Université suppose l'élaboration d'un vaste programme pédagogique, et la formation de spécialistes pour l'éducation dans les lycées et Universites.

En effet, l'introduction de l'informatique doit être accompagnée par une recherche visant à concevoir, à mettre au point, à expérimenter et à diffuser des logiciels. La formation des enseignants est un préalable necessaire à l'introduction de l'informatique à l'ecole car elle est la garantie la plus sûre de l'introduction de nouvelles methodes d'enseignement et du developpement des didacticiels.

Dans l'Ouest-Afrique, l'introduction systématique de l'informatique à l'ecole n'est pas encore envisagée, à part quelques rares cas (3 collèges catholiques au Sénégal, le Lycee français d'Abidjan, l'expérience Logo à Dakar). Aucun établissement primaire ou secondaire ne possède de materiel informatique. Cependant, beaucoup d'etablissement de l'enseignemment supérieur possède du matériel informatique et presque tous dispensent des cours d'initiation en informatique.

Au Senegal, l'expérience du Laboratoire Informatique-Education consist à initier de jeunes enfants à l'ordinateur et au langage informatique afin de permettre les adaptations socio-culturelles necessaires à l'introduction de cette nouvelle technologie dans l'enseignement. A Dakar le centre mondial informatique et ressources humaines a demarré un projet en Mars 1982 au centre de Recherche pedagogique de l'Ecole Normale Superieure (ENS). Une équipe pluridisciplinaire composée d'I mathématicien, I informaticien, I sociologue, 2 instituteurs, I psychologue envoyes en formation au "Logo computer center de New-York" dirige le projet.

Le langage Logo crée par le professeur Seymour Papert, mathématicien du MIT, permet de solliciter l'esprit de l'enfant, son raisonnement, son imagination, sa créativité. L'enfant conçoit son programme, l'execute et vérifie la logique et l'exactitude. Ce langage permet de passer de l'intuition à l'abstraction; beaucoup de notions de la geométrie euclidienne sont percues intuitivement à l'aide de sa "geométrie de tortue". Ce langage permet de saisir intuitivement la notion mathematique de l'infini. L'expérience porte sur un echantillon d'élèves provenant de 5 écoles primaires de Dakar qui est représentatif des réalités socio-économiques et culturelles du Sénégal. Dans chaque école, 10 élèves sont choisis (moins bons,moyen,bons).Chaque séance Logo accueille une école et dure 90 minutes.Chaque élève est reçu 3 fois par semaine.L'équipe pluridisciplinaire a axé ses recherches sur la grammaire (conjugaison),sur le langage Logo-Wolof qu'elle a mis au point,sur la géometrie et les mathématiques modernes à l'école primaire. Il est envisagé actuellement l'installation de IO micro ordinateurs par école experimentale et la formation de futurs maître-instituteurs a éte entreprise.

Conclusion

L'informatisation de l'école est en train de devenir une réalité dans les pays industrialisés;elle doit devenir une nécessité impérieuse dans le Tiers-monde sous peine d'aggraver le fossé qui le sépare des premiers. A cet effet il est necessaire de créer des structures perma nents de re cherche et de formation des enseignants.

Dr. Leo H. Klingen

Bemerkungen zu

"THE INFLUENCE OF COMPUTERS AND INFORMATICS ON MATHEMATICS AND ITS TEACHING"

Diese Stellungnahme bezieht sich ausschließlich auf den Mathematikunterricht in der gymnasialen Oberstufe (16 - 19-jährige Schüler, pre-university-level).

Es wird zwischen 2 Zuständen unterschieden.
Zustand A ist die Situation, in der sich heute die meisten Schulen befinden. Zustand B besitzen heute nur wenige Schulen; es ist jedoch wahrscheinlich, daß er für die Zukunft der Regelzustand wird.

Der Zustand A ist dadurch gekennzeichnet, daß alle Schüler dieses Alters Taschenrechner mit wissenschaftlichen Funktionen besitzen, etwa ein Viertel von ihnen auch programmierbare Taschenrechner; die Schule besitzt eine Computeranlage, an der für den Mathematikunterricht numerische software kleineren Umfangs vorhanden ist, manchmal auch ein Plotter.
Das bedeutet z.B., daß der Graph einer konkreten Funktion nicht mehr unter Ausnutzung der Kenntnis seiner ausgezeichneten Punkte, sondern durch Berechnung von 20 beliebigen Punkten über den Taschenrechner ermittelt wird; auch Grenzwerte von Termen werden zunächst auf diese Weise heuristisch gefunden. Lineare Gleichungssysteme oder bestimmte Integrale, darunter näherungsweise auch uneigentliche, werden über den Schulrechner gelöst, die Lösungen von expliziten Differentialgleichungen ersten und zweiten Grades approximiert und ihre Lösungskurven ebenso wie andere Funktionenscharen über den Plotter dargestellt.

Im Zustand B werden symbolische Verarbeitungen möglich. Das gilt für die äquivalente Umformung von Termen und insbesondere für das Lösen von linearen und quadratischen Gleichungen und Bruchgleichungen sowie linearen Gleichungssystemen (auch mit Formvariablen) nach beliebigen Variablen. Ebenso ist symbolisches Differenzieren und symbolisches Integrieren der im schulischen Rahmen vorkommenden Integrale möglich. In kleinen zusammenhängenden Problemfeldern und Anwendungen wird die Verknüpfung einschlägiger Funktionen automatisch durchgeführt. (Beispiele: arithmetische und geometrische Folgen und Reihen, Trigonometrie, Zinsrechnung, Kinematik der geradlinigen gleichförmigen und gleichmäßig beschleunigten Bewegung, Strom-Spannungs-Widerstands-und Energierechnungen aus der Gleichstromrechnung usw.)

Didaktische Konsequenzen.

Zustand A.
Während die Lehre für die frühen Jahre der Sekundarstufe I bestrebt sein muß, gewisse Handfertigkeiten ohne Taschenrechner (z.B. Prozentrechnen) zu erhalten, was zu teilweise defensiver Einstellung gegenüber dem neuen Medium führen muß, kann die Entlastung durch den Rechner für die älteren Schüler offensiv bejaht werden, weil sie Anlaß zu vielen mathematisch interessanten Fragen liefert. (Beispiel: Ein Suchverfahren hat am Computer innerhalb eines Intervalls keine Nullstelle ergeben. Existiert überhaupt eine Nullstelle, lassen sich ggf. Schrankensätze für ein anderes Suchintervall finden?) Das Verständnis eines effizienten Algorithmus soll auch zur Not, wenn der Taschenrechner nicht zur Hand ist, die Handrechnung erlauben (den ggt über den Euklidischen Algorithmus und nicht nur über eine Zerlegung in Primfaktoren, die Quadratwurzel über das Heronverfahren und nicht nur über die quadratische Ergänzung u.ä.)

Man wird außerdem auf die vollständige Darstellung der Theorie an den durch die Maschine "abgenommenen" Stellen nicht verzichten können, kann allerdings die Darstellung zugeordneter Beispiele auf einfache Fälle beschränken. So werden z.B. die aufwendigen Diskussionen konkreter zusammengesetzter transzendenter Funktionen, wie sie zur Zeit das Zentrum von Abituraufgaben darstellen, teilweise überflüssig. Stattdessen sollte man allgemeine Funktionseigenschaften ("Jede ganze rationale Funktion 3. Grades besitzt genau einen Wendepunkt und ist punktsymmetrisch zu ihm") bearbeiten lassen.

Zustand B.
Da das Arbeiten auch mit aufwendigem Kalkül durch die Maschine übernommen wird, (dazu gehört z.B. auch die Vereinfachung der Koordinatentransformation für den gerade zitierten Satz) werden im bisherigen Zeitrahmen neue Gebiete zugänglich; hier reicht die Erarbeitung der wesentlichen mathematischen Grundgedanken, um zusammen mit maschineller Hilfe vielfache Anwendungen anzugehen. Wichtig ist, daß Tragfähigkeit und Reichweite solcher erweiterter Methoden so durchsichtig werden, daß Ansatzvoraussetzungen und Ergebnisrestriktionen sinnvoll durchgeführt werden können.
Im einzelnen kann man in Auswahl denken an

- Ansetzen von Differentialgleichungen
- Ansetzen von Differenzengleichungen und Diskussion der linearen Differenzengleichung
- Propädeutik der Fourier-Analyse
- Reelle Kurven und andere Elemente der Differentialgeometrie
- Kubische Splines, auch parametrische
- Chiquadrat-Verfahren
- Regressionskurven und anderweitige multivariate Statistik
- Markow-Ketten
- Algorithmen der Graphentheorie in Anwendungen
- Komplexe Abbildungen
- Numerik, insbesondere Fehlerfortpflanzung.

Pädagogische Schlußfolgerung.

Es kann kein Zweifel darüber bestehen, daß eine bisherige didaktische Notlüge aufgedeckt wird: der durchschnittliche Schüler bekam durch perfektes, wenn auch schematisches Umgehen mit trivialen Umformungen an den meisten Schulen befriedigende oder gute Zensuren und mußte annehmen, deshalb Mathematik leidlich zu beherrschen. Wenn ihm die Maschine diese Leistung abnimmt (Zustand B), könnte ein Zustand resultieren, daß zwar eine Fülle interessanter Problemstellungen für die Schule übrigbleibt, jedoch diese weder durch die Maschine noch durch unsere durchschnittlichen Schüler gelöst werden können. Es stellt sich dann die Frage, ob die schulische Lehre von Mathematik in der Lage ist, dasselbe Kunststück zu vollbringen, welches die schulische Lehre des Faches Kunst in den letzten 50 Jahren vollbracht hat, nämlich einen erstaunlichen Anteil von Schülern zu kreativen Leistungen zu erziehen. Für die hier gedachte Mathematik würden schon Fähigkeiten zum problemlösenden Denken reichen.

Ein experimentum crucis in dieser Beziehung ist die Konstruktion eines neuen Stils von Klassenarbeits- bzw. Klausuraufgaben. Dabei muß man davon ausgehen, daß keine komfortablen Computer-Einzelplätze für mehrere Kandidaten oder gar einen ganzen Kurs aus ökonomischen Gründen zur Verfügung stehen können, wo ad hoc detaillierte Programmierung stattfinden kann. Dagegen erscheint es tatsächlich denkbar, daß gut dokumentierte und vorübersetzte softwaretools in großem Umfang so vorhanden sind, daß Schüler bei geeignetem Aufruf (und unter entsprechender Aufsicht) schnell ihre Ergebnisse sukzessive holen können. Die Lösung solcher Aufgaben verlangt dann vom Schüler mehr Verbalisierung als Rechnen, mehr Begriffs- statt Kalkulorientierung. Sie besteht in einem wohlbegründeten Ansatz, dem Aufruf vorhandener Prozeduren mit genauer Bezeichnung der richtigen Parameter und ggf. dem Verbund solcher Prozeduren, schließlich einer ausführlichen Ergebnisdiskussion.

Es wäre absurd, aus der neuen Lage schließen zu wollen, die Schule sollte weniger Mathematik behandeln; im Gegenteil: weil der numerische und der algorithmische Aufwand entfallen, können viel reichere mathematische Aussagen als bisher erschlossen werden und für die Schüler die Beziehungshaltigkeit dieser Wissenschaft wesentlich erhöhen.

INFORMATICS AND VS. MATHEMATICS IN SCHOOL EDUCATION

Klaus-Dieter Graf

Freie Universität Berlin
Institut für Didaktik der Mathematik und Informatik
Berlin (West), Federal Republic of Germany

1. Problems at the university level

The ICMI discussion document on informatics and mathematics as well as the symposium in Strasbourg have underlined, that the computer is a very powerful tool for teaching and learning mathematics. They have also shown that many effective programs and models for math courses relying on computers do already exist at some colleges and universities.

Little notice, however, has been given to another development, which is occuring in the Federal Republic of Germany at least: More and more students at our universities want to study computer science or informatics instead of mathematics. Very often the reason is that they think that in vocationary life informatics will play the part which mathematics has been playing until now.

This may be right to a certain extent. It does not follow, however, that they will need less logic and mathematics. In Question 3 of the second chapter of the ICMI document you find this statement: "the use of computers and of informatics demand more mathematics, better understood". This has to be stressed clearly when advising students.

Still, mathematicians at schools, colleges and universities have to face informatics as a challenge, and they have to become aware of the fact that they have to do more than just to use computers for their own purposes. They have to justify the importance of each field of mathematics again as to application in science and practice and they have to become good partners of informatics. As a consequence all mathematicians have to learn something about informatics, e.g. elementary notions of complexity, correctness, effectiveness and termination of algorithms.

2. Problems at the Secondary School Level

This argument cannot be discussed in more detail here. Instead of this attention shall be turned to its importance and consequences for our schools. Also there the relations between mathematics and informatics have to be settled more profoundly than until now.

In Germany informatics courses have been introduced in higher secondary education. They are elective; in some schools A-levels can be reached.

In lower secondary education there is a very powerful but somewhat chaotic development going on. Some experts and officials vote for independent computer science courses there, some propose and practice integration of computer science into other subjects, like mathematics of course, but not only mathematics. Political authorities at the moment are backing the second way, pushing computers and related curricula into schools. The results are not very satisfactory so far, since there is little real integration but mostly addition of computer science methods and contents to mathematics, technology, physics etc. .
This may be programming or switching algebra or using program packages. So there is still the danger of separation instead of integration, which would be more useful with regard to the situation in vocationary life mentioned before.

More details about the state of the art in Germany can be found in a paper produced for ICME 5 at Adelaide in 1984 :

> Baumann/Graf/Klingen/Winkelmann :
> Informatics - A challenge to mathematical education.
> In : ZDM (International Reviews on Mathematical Education) 1984/4, p. 121 - 131.

There is also a number of very detailed contributions about experiences and propositions in Germany in this volume by A. Engel, L. Klingen, B. Winkelmann, H. Meissner, R. Straesser and R. Biehler.

More models and experiences can be found in a small book "Computers in school - models for mathematical education (5) .

3. Mathematical Education strikes back

It follows from the above that informatics also is a challenge to mathematical education, possibly more severe and more aggressive than sized up by ICMI. Informatics pulls off pupils from mathematics and it pulls off time for math education.

So also on the school level we have to think mathematical education over and prove its importance again. More, we have to show that math education already fulfills many of the objectives which informatics has written on its flag for the schools. This means that math education should also challenge informatics, at least as to its general educational background. The experiences in math teaching should be considered carefully when planning informatics education - integrated or not.

The following cites some more detailed arguments about this topic in a paper prepared for the world Conference on Computers in Education at Norfolk, Virginia in 1985.

Since 1980 intentions for informatics education in lower sec. education have been formulated by several institutions in Germany. It turns out that many of these are very close to corresponding intentions of math education, some using different terms. There is no use in cancelling such intentions in math education, since they have proven important for many generations already. If they are mentioned in the discussion about informatics education, this only means that they have been neglected in mathematics at school. Thus they have to be put into the centre of attention again.

What are such intentions? A fundamental paper of Uwe Beck from 1980 (2) lists for instance:

- systematic finding of algorithmic solutions of problems
- formulation of algorithmic solutions of problems in programs
- dealing with problems close to real life, using adequate data structures and forms of date processing organization.

These intentions imply more detailed aspects for the ways of teaching:

- teaching should be oriented towards application
- non-numerical problems should be considered, also complex ones
- solutions should be constructed in an engineer-like way
- solutions should be found in groups and working on projects
- programming should be preceded by structuring

The paper outlines that most of these demands are fulfilled today (and before) in mathematics education in German schools. As to algorithms one even can go back to F. Klein, who pointed out their importance in 1935. But there are more points: In math education the "genetical principle" plays an important part. It leads to viewing instruction in school as a process. Also, math education today is oriented clearly towards applications, beginning in primary schools. Modelling is an important step in this area and it is done in a way very similar to structured programming. It is done less strictly 'top down' than in programming but it admits different levels of precision at each step of the process. Math education is also dealing with complex problems and it admits work in a group.

The work of H. Löthe (3) about different methods of programming (from FORTRAN via LOGO to FP) also proves that mathematics and informatics should not be taught independently in school. A right combination of their methods will arm students with effective strategies for problem solving. One of these will be using the computer not only for executing solutions of problems but first for finding and formulating the solutions.

Thus mathematics teachers can rely on strong resources in the theory of math education, but also in reality, when facing the challenge of informatics. Many intentions just have to be reformulated and have to be attacked with new means, e.g. another language for describing and solving problems, maybe a programming language. Two examples of textbooks for teachers and teacher students following these ideas are (4) and (5).

REFERENCES

(2) Beck, U., Ziele des zukünftigen Informatikunterrichts. In: Journal für Mathematik-Didaktik 1, H. 3 (1980) 189-197.

(3) Löthe, H., Arten des Programmierens und Programmiersprachen - ihr Stellenwert im Mathematikunterricht. In: Zentralblatt für Didaktik der Mathematik 15, H. 5 (1983) 220-224.

(4) Graf, K.-D., Informatik. Eine Einführung in Grundlagen und Methoden, Verlag Herder, Freiburg (1981) 256 p.

(5) Graf, K.-D. (Hrsg.), Computer in der Schule - Perspektiven für den Mathematikunterricht, Verlag Teubner, Stuttgart (1985) 203 p.

V.M.MONAKHOV

correspondent-member to the Academy of Pedagogical Sciences of USSR, doctor of pedagogical sciences, professor

The information science and computing technique study influence upon school education

Modern economic and social conditions assigned Soviet school with new major tasks. They were concretized in the "Guidelines of the Reform of General Educational and Vocational schools" and Party and State documents adopted to cope with its tasks.

These documents name among the major the task of "arming the pupils with knowledge and skills of modern computing technique utilisation, fostering extensive usage of computers in the school process", ensuring the transition to general computer competence of the younger generation. These tasks did not arise apropos of nothing. The school is carrying out a direct social demand: to ensure a well-educated person ability to meet the requirements of modern science, technology, industry and culture.

The exponental growth of computing technique, the expansion of microprocessors nowadays determine among other factors industry automation and changing of the nature of labour in all spheres of productive and scientific activity of a modern man, his social and private life. Under these conditions the documents of the reform represent a new social demand from school. At the same time they develop Lenin's principle:"It was thanks to labour that man became what he is and now labour must make him a personality developed in an all-round way.

Soviet school begins the fulfilment of the Party and Government decisions not from the starting-point. There is a 25 years experiences of the basics of computing technique and programming school teaching.

The teaching of programming and electronic calculating machine operation (as the course was then called) began in advanced mathematics study classes in 1959. There was a number of major fundamental investigations (carried out by V.M.Monakhov, S.I.Shwarzburd and other research-workers) and significant results were attained. For example it was found out that the system of advanced mathematical training, that includes some elements of programming and electronic calculating machines operation study, ensures beneficial conditions for outgoing pedagogical experiments, contributes to the perfection of the secondary (and mathematical in the first place) education.

The working out of algorithms and programming secondary school study facultative course began in 1966 (Monakhov V.M., Antipov I.N. Demidovich N.B., Lapchik M.N. and others). Major aspects of algorithms teaching to schoolchildren such as making use of flow-synthesis method of algorithms notation were technically worked out at the stage of information science and computing technique introduction at school. General methodological approaches to the new school subject teaching were determined, and its main educational value and interdisciplinary relations were established.

The experience gained as a result of facultative studies introduction probably lacked mass corroboration but still it was considerable. The next stage of its amplification and further investigation began in 1970, when a new unit was introduced into the manual of algebra for the 8th form:"Algorithms and programming elements". As a result every pupil of the 8th form has been getting acquainted with algorithms and types of programmes and has been getting a general idea of the main principles of electronic calculators operation and their role for more than 15 years already.

The 25 year Soviet school study of the problem of information science and computing technique introduction into the general educational ~~xxx~~ subjects cycle resulted in working out of methodical and educational background for the ways and means of ~~xxxxx~~ teaching. New manuals, teacher's books and practical courses for pupils were developed. A general approach to this subject secondary school teaching was worked out.

This year (1985-) opens a new stage in information science and computing technique introduction into the Soviet school process. Mass production of reliable and comparatively cheap microprocessors made it possible to supply all secondary schools with computing technique. A new subject is introduced next year in all the secondary educational institutions of our country, called "Basics of information science and computing technique". In the course of this subject teaching, according to the curriculum, pupils will study a wide and diverse spectre of concerning problems. What is more, the school subject course curriculum aims at bringing this discipline study into harmony with other school subjects that are the components of the secondary education course, and establish this discipline in its full value.

The prescribed duration of the new subject course is two years of senior classes study and it will be preceded with a preparatory course, organized by means of dwelling on a number of constituent problems in the course of other school subjects, mainly mathematics.

The main problems that constitute a course of information science and computing technique study are as follows:

1) The nature of electronic calculating machine, its construction and principled of work; what a man needs calculating machines for.

2) Algorithms, systems of their notation, types of algorithmic process, their significance.

3) Elements of calculative mathematics and calculus arrangement.

4) Introduction into access to programming language, computer program supply, applied program sets and their sugnificance and usage.

5) Elements of the computer and principles of their work, ways of presentatoin of information in computer, the structure of machine command and it's implementation.

Besides, the curriculum prescribes a large quantity of practice in all items of the course.

The new subject, introduced into school curriculum is based on three main notions.

First of all it's a notion "algorithm - program", i.e. ways of notation of problems solving.

Secondly - it's a notion of information, i.e. of object and result of problem solvig.

Thirdly - a programmed computer as means of problems solving on the basis of an algorithm.

So, the notion of autometic treatment of information cleared up by the Soviet school methods came into the curriculum naturally.

It's important to note some more peculiarities of the new curriculum. The parallel into-duction of two forms of algorithms notation is suggested in the new curriculum: by means of flow-synthesis algo-
and the so-called algorithm notation. The method of flow-synthesis algorithm
~~rithms~~ is worked out in our school in rather a detailed way, they were compnents of algebra course for the 8-th form, components of facultative course. The flow-synthesis algorithms are available for pupils, universal, easily become apparent by any way of visual demonstration. They ensure the moulding of the total amount of knowledge and habits of solving applied problems, dealing with algorithms. Flow-synthesis

algorithms help pupils to clarify the essense of the notion "algorithm", its properties, visually describe all types of algorithmical processes. Algorithmic notation, in its turn, treated and checked experimentally in schools of Novosibirsk (under the guidance of academitian A.P.Ershov) as being easily made obvios by means of visual demonstration, maximally approximates in form to the modern program language of the higher level. Its usage facilitates the transfer to the compiling of routine by means of algorithmic language. The latter is of extreme impotance, as teaching pupils a concrete program-language is not the main aim of the new course. The course aims at the working out of the idea of algorithmic language programming comprehension by students, it being a a form of algorithmic notation, "understandable" by electrinic calculating machine.

The usage of several forms of algorithmic record (flow-synthesis method, algorithmic notation program) illustrates for the pupils the diversity of algorithm describing ways, train them in using a form appropriate for the concrete problem solving, facilitates, if necessary, their future mastering of other program languages.

We would like to mention the inclusion of problems dealing with the form of calculater information supply into the course of basics of information science and computing techniques.

Reaching the level of the computer commands will allow the pupils to better realise its operation which reflects and will illustrate the poly-technical principles of school teaching.

The main aims of teaching information science and computational technique at the secondary school are as follows:

1) to develop algorithmic culture of the pupils (naturally, this aim is far beyond the scope of teaching programming and it is becoming to more and more extent the aim of a number of educational subjects);

2) to mould in pupils' minds adequate ideas of the main principles of computer operation and their potential;

3) to develop primary skills of microcomputors operating and their usage in education and practice.

Here is the outline of the most important methodical problems. The skill to present a complicated problem in the form of its simplified components must be a part of algorithmic culture: it implies the ability to mould informaticaly closed subproblems and to solve them in the necessary succession. Using the terminology of programming we call it the skills of unit programming which are of extreme value in practical work with a computer.

Teaching pupils the principles of electronic computer operation implies the necessity to consider the following. AS a rule, modern electronic computer operation is based on a few very simple and highly generalised principles (program control, uniformity of memory, moulding of complex operations out of simple ones etc), that determine the general potentialities of an electronic computer. In modern computers these simple principles are not so obvious (which was characteristic of the first computer generation in the fifties), in order to exibit them to pupils while teaching, it is necessary to help them to single out the principle essense from complex technical realization details.

WE should say, that the details of this realization include its hardware and software as well, the latter being a system of complex programs that constitute an integral part of any modern electronic computer. School cannot keep pace with computing technique development (and it is unnecessary), the rate of new generation of computers production is increasing. That is why the proper accentuation of educational values is essential.

While developing pupils' skills of programming and microcalculators practical usage, our school does not aim at making them professional programmers. Programs composed by pupils must be compact, coherent and follow the natural order of record. We should not insist that pupils optimise their programs in the best possible way under the present mass school conditions. Whereas a professional values the result, i.e. the exactness of calculation and minimization of the time of a program operation, it is the problem solving process itself, that is essential for the pupil.

We should train pupils not only to compose programs, but at the same time to use the ready-made sets of applied programs, understand them, to be able to choose and use the program they need. School must prepare the pupils for their future productive activity, train them in making use of the systems, worked out by specialists, instead of "inventing a bicycle" by themselves.

The introduction of general course "The basics of information science and computing technique" into Soviet schools will greatly influence in the coming years the methods of moulding general mathematical and algorithmic culture of pupils. The principle changes in traditional structure of school mathematics course and the structure of applied sections of general education courses dealing with electronic computers and their use in teaching and practice, are expected. Computor is a powerful reseach instrument in practically all branches of modern science, and school must teach pupils to use this instrument in their studies. Electronic computor can facilitate the processing of chemical and physical experiments results for the pupils, help them in bringing different regularities to light. The graphical possibilities of computors help in visual presentation of the material studied. It is necessary to point out the role of modelling by means of electronic computors, not only does it favour a better comprehensoin of the material studied, but also carries out great ideological function by ways of helping the pupils to understand the essense of a studied process and arming them with a powerful means of cognition.

The bylateral positive relation of information science with other school disciplines should also be mentioned. On the one hand the new technology attracts pupils' interest and stimulates them in better studying other school courses for it requires a higher level of knowledge in other sciences (mainly mathematics and physics).

On the other hand, pupils who are mostly interested in other subjects realize that under the modern conditions not a single branch og science aand national economy can do without electronic computors and thus they ought to give more attention to the information science and computing technique study.

The strengthening of algorithmic approach to problem-solving with its postfollowing realization by means of a computor implies significant changes in the methods of teaching. These changes are first of all a result of raising a standard of strictness of reasoning, and preciseness of substantiation and in the long run a result of raisnig a scientific level of the course itself. The fact that these demands do not appear from outside, that is from the teacher, but by some inner way, by way of pupils' cooperation with the task, is also important to mention. And even more important is the fact, that a program composed by a pupil can be tested by means of the iron machine logics, which interprets the program as it is, i.e. as it is written and not as the pupil meant it to work. The comparison of what was mentally planned with its realization is a fundamental means, the educational effect of which is impossible to underestimate (as V.P.Kravtsova points out).

The experience gained by the Soviet school system in the course of a 25 year period that passed from the first attempts of information science and computing technique teaching introduction enables us to make some conclusions.

Analizing the tendency of the basics of information science and computing technique introduction into the school course,most research-workers point out that the electronic computing machine programming process is in itself a powerful educational, methodological and didactical stimulus that fosters accumulation of algorithmic and mathematical culture of pupils.

The specific habits and skills built up in the process of working out problem-solving algorithms, program composition and arrangement of the information for computor processing straighten out the way of pupils' thinking in a natural way.

Algorithmic orientation of the contents of school education helps in building up skills of broad mathematical generalisations, favours the development of mathematical abilities, creative potential, stirs the minds of the pupils to greater activity.

Further development of algorithmic orientation at school helps to prepare a future industrial worker for appreciation and practical implementation of algorithmic skills in the modern national economy.

Proceeding from the essence of the modern national economy demands of school education, we must paint out that the necessity of moulding a certain standard of modern algorithmic culture of pipils will yearly become greater.

Thus the introduction of a new discipline "The basics of information science and computing technique" into school curriculum is an important and timely event, for most psychologists think that it is easier to begin the study of a new content at school than later on. It is at school age that first impressions of a new subject are fixed in pupils' minds, become their conviction and way of thinking.

The introduction of a new school subject undoubtedly requires a sufficient reconstruction of the whole studying process, but the experience gained by the Soviet school makes it possible to hope that our school will cope with this task.

COMPUTERS AS A UNIVERSITY MATHEMATICS TEACHING AID: TOWARDS A STRATEGY

Dr. Michael Thorne, University College, Cardiff, Wales, UK.

Introduction

The fundamental assumption of this paper will be that there is at least a prima-facie case for the use of computers as teaching aids within undergraduate mathematics. Herein, we shall be concerned with the next stage - organising pilot studies - and the difficulties of doing so in a cost effective way across various degree awarding establishments in a given nation. In doing so we shall draw heavily upon lessons learned from the leading role the UK has taken in Computer Assisted Learning (CAL) in schools, and from the current situation within mathematics departments in UK universities. Discussions with colleagues from the USA, Canada, Australia and Tasmania and a visit to Sri Lanka suggest that there are at least elements of the UK experience which have relevance and/or counterparts internationally. The issues we shall raise affect both university teachers and the entire university administration system, from government level downwards.

Learning from Experience

It is obvious that spending a lot of money on new computers and software would almost certainly improve our mathematics teaching in some respect. But to maintain a government's long term respect - and hence, long term investment - the resulting achievements must be cost effective. From a government's point of view, that means our undergraduate teaching must get a better report card from industry. If our graduates meet the modern demand for flexible thinkers capable of adapting to both the changing demands of a given problem as well as different product bases, pressure will come from industry for greater investment in these new teaching methods. Few countries have a surplus of able mathematicians, yet there is often a small proportion of graduates who find it hard to get jobs. Faust [1] summarises recent UK experience:

> Recent figures indicate that about half those graduating from university with a first degree in mathematics start employment in the UK, whilst another quarter continues in full-time education here. The final quarter covers those going overseas, those not available for work or study, and the unemployed who, 6 months after graduation, account for just under 10 per cent of the graduates.
>
> If those who do postgraduate teacher training are lumped with those entering employment directly after graduation, the employment sectors of first degree mathematicians are commerce (40 per cent), industry (30 per cent), education (20 per cent) and the public service, public utilities and transport (10 per cent). The main commercial employers are the financial institutions, chartered accountants and computer software houses; in industry they are electronics firms and computer manufacturers.

Nearly one in ten take a relatively long time to find work, evidence, indeed that there is room for improvement in our teaching of mathematics and possibly a reminder that our students must emerge as well balanced <u>people</u>.

Faust concludes his paper with a warning:

> ..... employers were seeking recruits who were intelligent, numerate, well educated, capable of solving a variety of problems, able to communicate clearly in speech, and in writing, and personable..... During their careers they must develop skills and learn much that is new so that they may fill more senior posts or transfer to new fields of work. To this end recruiters, through their selection procedures, attempt to assess the "potential" of the applicants, and one important aim of a university should be to provide the stimuli that will result in the personal as well as intellectual development of their students.

It follows that the involvement of computers in mathematics teaching must avoid encouraging undergraduates to become computer junkies, spending most of their free time with machines rather than humans.

A decade ago, a five year £2.5 million government sponsored National Development Programme in Computer Assisted Learning (NDPCAL) was undertaken in the UK. Its aims were:

> to develop and secure the assimilation of computer-assisted and computer-managed learning on a regular institutional basis at reasonable cost
>
> (Hooper [1])

with a bias towards undergraduate level science teaching but away from mathematics in its own right. NDPCAL was not an unqualified success, the following criticisms (from O'Shea and Self [7]) being typical:

> .... as usual, there is more to be learned from the shortcomings than from the successes of the enterprise. The 'institutionalisation' aim led naturally to the selection of projects which were more likely to be accepted by the host institution. Projects tended to play safe by attempting to implement existing objectives, and to avoid significant innovatory developments, knowing that in only five years or less they were unlikely to bring about major changes in the educational system itself. Projects were led away from research to applications, the NDP being explicitly a development programme with no research policy. The idea that in 1973 there existed a body of knowledge about computer-assisted learning which it was worth developing without further associated research seemed at the time fanciful, and in retrospect absurd.
>
> The absence of a proper experimental design resulted in a proliferation of 'case-studies', the significance or success of which is virtually impossible to determine.

And if all that were not warning enough, O'Shea and Self point to the technological short-comings of NDPCAL, which, as we shall later attempt to indicate, may seriously restrict the 'assimilation' of computers into university level mathematics education.

> Technologically, the NDP projects were unadventurous. They made use of general-purpose computer systems, typically mini-computers, not specifically designed for computer-assisted learning and almost all the teaching material was written as small programs in FORTRAN and BASIC, two languages whose design reflects their vintage, but which do alas provide the desired transferability since almost all computer manufacturers have felt obliged to provide compilers or interpreters for them. Some material was written in a conventional author language.

.....In the event, the NDP has been submerged under the wave of microcomputers, of which there is no mention in the final (NDP) report. The report's conclusions on technical matters have thus been rendered obsolete and much of the original development work is now seen to be irrelevant.

Not one NDP-funded package has been adopted by all UK universities, or even by the majority of them. Those which are still in use show their age because of a dependence upon 'teletype-style' interactions with their users wherein hardware, now obsolescent, could only accept and react to one line of communication at a time. In our present context this remains a major problem since the computing power we must make available to the learner demands the university mainframe but most university mainframes either still have the obsolescent teletypes or visual display units which are incapable of proper graphical displays, offering less flexibility of presenting and gathering data than the cheapest of high street microcomputers. Increasingly this problem is being overcome by individual university departments on an ad hoc basis with the result that, say, applied mathematics has adopted the Apple II as a graphics terminal to the mainframe whereas statistics are using a BBC Microcomputer.

Multiplying the problems caused by this non-standardisation with the different types of university mainframe machines and with the different operating systems on those machines it is easier to have sympathy with what O'Shea and Self described as the technologically 'unadventurous' spirit of the NDP.What do Cyber, DEC, Honeywell, IBM, ICL and Prime have in common except FORTRAN? But this problem must be

overcome now through government leadership. Hardware prices have continued their exponential rate of fall since NDP but software costs have rocketed. Cost-effective software development will involve producing teaching and learning packages which many sites are able to use. Disparate hardware forces developers into assuming the least common denominator at each site making it very difficult for computers to play any key role in the mathematics curriculum at all - as we shall discuss later.

UK primary schools have avoided this problem to a great extent by standardising on three machines. One of these has been adopted by more than 80% of primary schools and another accounts for the vast majority of the remainder. Apart from its small take-up within schools, the lack of availability of proper software development tools for the third machine and its slow graphics capabilities make software development for it expensive. As a result, it has all but been dropped by the official software producing agencies.

The second most popular machine was designed in the UK for school use and is not marketed to the home buyer. As a result its user base is so small that educational software development for that machine has been almost entirely at the taxpayer's expense. Commercial publishers cannot hope to recoup their outlay on sales of a few thousand packages with retail value about US$10. University level software sells in low volume and at a high price but there would be a reasonable commercial opportunity internationally for IBM based Mathematics teaching software at the moment, for example. But the costs of any large system development are enormous and it would be foolish to ignore the contribution hardware manufacturers could make: software sells hardware.

Hardware Selection

It cannot be emphasised enough that the hardware is the cheap part of a computer system. Saving $10000 by buying terminals without graphics may involve $20000 or more putting a workable user interface into a single software package. Moreover, the facility for non-linear input is crucial. Devices such as the Apple Mouse, light pens and touch sensitive screens are not frippery - the QWERTY keyboard is very, very limiting.

As indicated before, we do not regard microcomputers as a very useful hardware base for the involvement of computers in undergraduate mathematics, given their current level of computing power. Algebra systems with wide applicability and non-trivial computational ability cannot be mounted on a 64K, 8-bit, floppy disk based micro. But the argument for making high computing power available to each learner really springs from the lack of time for lecturers to familiarise themselves with software packages. If the teaching packages are actually professional mathematical tools, any effort expended on learning how to use them is easily justified by a lecturer. In algebra, for example, the package CAYLEY (Cannon [1]) could be used for both teaching purposes and research but it was designed as a research tool. Yet CAYLEY will not run on a present-day microcomputer. Networked systems of micro-computers are not adequate either. In general these are intended as resource-sharing links and do not enlarge the computing power available to each user on the network. Moreover, a network manager is required which either involves paying a technician or steals time from a lecturer's research. The same is of course also true of a laboratory of free standing micros: an individual university department in this way takes on work which the university's Computer Centre is paid to do.

To redress the balance slightly, I would like to conclude this section by admitting two useful roles microcomputers could play in undergraduate mathematics education. The first is in electronic blackboard applications and the second as in CATAM (Harding [1]) where the students are expected to do some programming.

Software Development and Maintenance

During an address at CAL 83 Bryan Spielman classified CAL software into 'amateur' and 'professional' types. Amateur software is programmed by the lecturer who uses it and is not robust in its user interactions. For example, certain keyboard combinations will cause the program to crash and the program doesn't explain what has to be typed in at every stage. This sort of software is very useful to its author but highly non-exportable because there is no adequate documentation describing how it works or what it does and the program is known to fail under certain circumstances. Professional software, on the other hand, is fool-proof in every sense. The program can cope with random key depressions at any point of its operation, the documentation is detailed and complete and the whole package has been thoroughly tested before release.

For small teaching points, amateur software may be cost effective, but no amateur or group of amateur programmers could have produced a system the size and complexity of CAYLEY which took several people fifteen years to develop. In general, governments and faculties may see the economic need for a computerised first year analysis or algebra course but category theory third year courses with an average of 4 students a year will not warrant professional software unless this is developed as a serious research tool. Moreover, there will be a good market for the commercial publishers in support materials for these first year courses which would therefore warrant the necessary outlay for their development.

But with all large-scale software, the major costs are not only in the initial production but also in maintenance:

> In general, it is impossible to produce systems of any size which do not need to be maintained. Over the lifetime of a system, its original requirements will be modified to reflect changing needs, the system's environment will change and obscure errors, undiscovered during system validation, will emerge. Because maintenance is unavoidable, systems should be designed and implemented so that maintenance problems are minimised.
>
> The costs of maintenance are extremely difficult to estimate in advance. Evidence from existing systems suggests that maintenance costs are by far the greatest cost incurred in developing and using a system. In general, these costs were dramatically underestimated when the system was designed and implemented. As an illustration of the relative cost of program maintenace, it was estimated that one US Air Force System cost $30 per instruction to develop and $4000 per instruction to maintain over its lifetime.
>
> Sommerville [1]

Educational Software Houses in the UK and USA producing material for home and school use have learnt the importance of having the teachers specify the software which is to be developed and then putting professional programmers to work on production. It has also become clear that trialling and feedback from comments obtained as a result of trials (which is then incorporated into the design) are absolutely essential. But however remarkable it is, after all this experience and that described in Sommerville [1] and in the literature in general, some people still try to develop software in an ad-hoc manner in the belief that they can do it more cheaply that way.

Unfortunately there cannot be short cuts. The financing of professional software has to be on a firm basis - either commercially or government sponsored or both. This will involve compromise on content between that which is ideally desirable and that which will appeal to sufficient other institutions to warrant the development costs. Just imagine, two universities forced to agree on whether continuity is introduced via $\varepsilon - \delta$ conditions or via nested intervals and whether operators are to be written on the left or right!

Research tools as teaching vehicles: Algebra and CAYLEY

Computers could transform our undergraduate teaching of algebra. At the moment students learn facts - definitions, lemmas, propositions, theorems - and rules: for example, how to find the inverse of a matrix. The students are then set problems which can be done in the short time allowed for homework assuming, in effect, that they work on their own. Stacks of Schaum's Outline series in university bookshops demonstrate that there is a template from which the majority of such problems stem. Motivation of the concepts gets very little time and when it comes to proving things rather than calculating students often have difficulty knowing whether what they've constructed is a proof. The very act of symbolism gets no motivation at all, a situation which could be easily rectified by giving students computer systems the behaviour of which they are to symbolise.

Better than this, it would be possible using the language Prolog as a base to design a system which would accept facts and rules about an algebraic system as and when they were discovered by the students and seemingly decide if enough information had been presented to prove a given hypothesis. The computer system would have no objection to student's erroneous or irrelevant deductions and would thereby encourage students to 'think round' problems and prevent the 'drying-up' syndrome (Buxton [1]).

CAYLEY allows serious calculations in very large finite groups (and small ones!). Playing with group elements in CAYLEY allows students to experience the great stride a subgroup concept really is. They can hunt for, try to recognise and attempt to define subgroups in intellectually demanding groups, not just S_3. They can come away actually knowing some groups and their subgroups. Given time they will discover some of the concepts like normal subgroups and centralisers for themselves: their role in mathematics will be more active. Matrix calculations are also done for you in Cayley. Thus, fairly large groups can be investigated for conjugacy classes, orbits etc in permutation or matrix representations. But best of all whether ultimately they turn out to be weak students or mathematical researchers, they will also have learnt how to use an important research tool, which has facets affecting every year of the undergraduate algebra curriculum.

Concluding remarks

All temptations to adopt programmed learning or drill and practice techniques through computer use should be eschewed by mathematics teachers everywhere. Whether our students enter research or industry a flexible mathematical approach is vital; behaviouristic training does not encourage flexibility. The proper integration of computers into university mathematics curricula most involve more than Papert's linear mix of technologies (see Papert [1]), which as he ably describes, is doomed to failure. A good and economical first step towards this proper integration is to change our teaching approach so that tools like automatic integrators are used essentially. The next harder step will spring from asking - as was done when CAYLEY was first conceived - what computer-based tools would assist both teaching and research in this subject? Sometimes this will be obvious; sometimes the historically motivated approach may be suggestive (as in Toeplitz [13] for example); sometimes it will be sheer ingenuity: Papert and Feurzig's Logo language and Abelson and diSessa [1] together offer a differential geometry course teachable at a much earlier stage than ever before thought possible and their Turtle casts light upon the subject even for the experienced eye.

REFERENCES

ABELSON H. and diSESSA A., [1], Turtle Geometry, MIT Press, London, 1980.

BUXTON, L., [1], Do you panic about maths?, Heinemann Educational Books, London, 1981.

CANNON J., [1] A Language for Group Theory, University of Sydney, 1982.

FAUST R.C., [1], Graduate Entry to Work, Bulletin of IMA, 3/4 March/April 1983.

HOOPER R. [1], The National Development Programme in Computer Assisted Learning: Final Report of The Director, Council for Educational Technology, London, 1977.

O'SHEA T. and SELF J., [1], Learning and Teaching With Computers Harvester Press, Brighton, Sussex, UK, 1983.

PAPERT S., [1], Mindstorms, Harvester Press, Brighton, Sussex, UK, 1980.

SOMMERVILLE I., [1], Software Engineering, Addison Wesley, London, 1982.

TOEPLITZ O., [1], The Calculus: A Genetic Approach, University of Chicago Press, Chicago, 1963.

COMPUTERS AS TEACHING AIDS

D.R. SARGENT

The Open University
Walton Hall
Milton Keynes
United Kingdom

Abstract

This paper concentrates on two phases of the use of computers as teaching aids, that when it is being used by teacher or student and the activities which can follow such use.

Although most of mathematics is still presented in symbolic form, illustrations and diagrams have always played their part in attempting to give substance to abstract concepts being taught to students. The advent of cheap microcomputers with graphics capabilities has now enabled movement as well as static diagrams to be portrayed. This paper looks at the methods by which the computer can be incorporated into the armoury of the teacher. It treats the computer as an addition to the traditional tools such as text books, blackboards, overhead projectors, cassette tapes and video. It does not address the problems of programming nor is it concerned with programmed instruction.

1. BACKGROUND TO THE PAPER

This paper is based upon the use of microcomputer graphics packages in the teaching of undergraduates of the Open University (OU), United Kingdom. Part of the Charter of the Open University reads:

"The objects of the University shall be the advancement and dissemination of learning and knowledge by teaching and research,

by a diversity of means such as broadcasting and technological devices appropriate to higher education,

by correspondence tuition, residential courses and seminars and in other relevant ways,

and shall be to provide education of university and professional standards for its students,

and to promote the educational well-being of the community generally."[1]

This extract shows how, even in 1969 when the University was founded, it was intended that all technological advancements should be considered when teaching students. It also implies that the students should be drawn from the 'community generally' and so OU students cannot be assumed to possess any formal educational qualifications. The main method of study in the OU is still by correspondence using specially prepared correspondence texts but students do have limited access (about 20 hours in their first year) to a tutor who is a qualified person employed on a part-time basis by the University. In addition students are expected to attend a week long residential school for some courses in order that more intensive tuition can take place.

The 'open entry' policy of the University means that students enter it with a variety of backgrounds. In order to create a common base of knowledge the student must undertake what is known as a Foundation Course in his/her first year. The Mathematics Foundation Course [2] starts at a low level in order to attract students with little formal mathematical knowledge. In designing the course it was felt that an informal geometric approach would be one way of exploiting the students' natural abilities. Most of the geometry used is intuitive and concerns transformations of the plane, expressed to students in terms of 'movement'.

A variety of media are used to reinforce students intuition: correspondence texts, transparent overlays to show physical movement; tape cassettes to describe it; television to illustrate more complex movement; and more recently computer graphics used interactively during the residential week.

Many of the views expressed in this paper result from discussions with part-time staff on their experience of computer graphics usage. The discussions took place over several years at OU residential weeks where computer graphics specific to the material being taught were available. Given the variety of the backgrounds of the part-time staff, from secondary teachers to university lecturers, there is no reason to suppose their views are not pertinent in a wider setting than that of the Open University.

Three phases of the use of a computer are considered:

- things to be done before the machine is switched on;
- the use of the machine when it is on;
- the activities to be done after it is switched off.

For most teachers the first phase is likely to be most problematic involving difficulties such as hardware and suitable software availability. However, this paper does deal with such a wide perspective because the software, specific to the subject being taught, is available. Nevertheless there are still problems to be encountered by the teacher in this phase. The second phase describes two uses of a software package in the classroom. The final phase is concerned with what the student has learnt from the experience and methods by which that knowledge may be reinforced.

2. BEFORE THE MACHINE IS SWITCHED ON

Despite the availability of cheap microcomputers with built-in graphics capabilities there is growing evidence that the machines are being left to the computing scientists and not switched on by other teachers. This has led to situations where computer graphics and the technology in general are being ignored in favour of less useful media. This would seem to be the case even when suitable hardware and software is made available to the teachers. Reasons for this avoidance would seem to include:

(i) Hardware

Such is the variety of hardware that it is reasonable to expect teachers to have some uncertainty when faced with an unfamiliar or infrequently used machine.

Equipment ideally needs to be set up so that it can be used with the minimum of 'technical' knowledge, preferably just the location of a single on/off switch.

All too often teachers see the setting up of hardware as a problem which exceeds the benefits which will accrue as a result of using it.

(ii) Software

In the general context this is a major problem area. However, even when software is available which addresses the detailed content of what is being taught, problems still arise. The software will rarely meet the exact needs of the teacher in terms of the way he/she presen the material. Moreover the software may not permit the teacher to pick and choose sections from it.

(iii) Teaching style

Most mathematics teaching of students over 16 years of age is done in the lecture/exposition style.

The use of a micro requires a different strategy where the teacher becomes part of the audience and where teacher and students work actively together. Teachers may feel insecure in these transitions from formal to informal styles.

(iv) Uncertainty

There is the obvious uncertainty which some people experience when using hardware together with that arising from wondering if the software will behave as it should.

But there are more subtle fears. A teacher may not have a particularly well developed sense of the imagery which say, a graphics package is trying to convey. Uncertainty can also arise in interactive programs where the student chooses the input parameters and the teacher is left to explain the subsequent action. Indeed in some investigative software the teacher may not have experienced all possible parameters and so students will need to be disabused of the notion that "teacher knows it all".

(v) Preparation

Using a software package requires much more preparation than a traditional lecture or tutorial. It may take some time to gain sufficient familiarity with program in order to use it efficiently in a classroom situation. As the purpose of the package is to suppor other material, students do not want to be burdened with a teacher wh is struggling to understand the operating system of the package.

. WHEN THE MACHINE IS ON

In this section we describe in brief a particular package used at OU residential schools in order to illustrate two methods of its use.

As mentioned earlier the Mathematics Foundation Course at the OU takes an informal geometric approach. In particular it discusses certain geometric transformations of the plane with a view to obtaining their matrix representation. The concepts of a rotation of the plane and reflection in one of the coordinate axes are easily illustrated in text. Their matrix representation is also easily derived (Bases are not discussed and all representations are with respect to (1,0) and (0,1).) Scalings are illustrated by photographic enlargements and are then generalised to situations where the scaling factors are not equal.

Each of these transformations is then generalised. For example a reflection in a general line through the origin is considered. The conceptual step is small but to find the resulting matrix **or the matrices** already found are utilised by breaking down the given transformation into the three successive transformations

first rotate the given line to an axis

then reflect in that axis

finally rotate the line back to its original position.

Students are encouraged to convince themselves that this combination is equivalent to the original transformation. They do so with the aid of diagrams printed on transparent material.

The generalisation of a scaling is what the course calls a 'dilation in shew directions' which means that the scalings are no longer along the axes but along general lines through the origin [3]. It is far from obvious to the novice what effect such a transformation has on a given point and it is even less obvious how its matrix can be determined. A graphics package was developed in 1980 on an Apple][e to aid the teaching and understanding of this material. Although graphics facilities have improved tremendously in this time the package is still in use.

The package is interactive and contains all the necessary operating instructions which a novice would require. However, it is constructed in such a way that a teacher can use an extract in the classroom. This gives two possible scenarios for its use.

Use by the teacher

In a classroom situation the question of generalising a scaling to a dilation in shew directions would be raised. In particular what does the latter mean and what is its effect on a given point? Rather than attempt to do this on the blackboard the package can be used. In the first instance it may be desirable to work only in the first quadrant until such time that confidence permits all four quadrants to be considered. Whatever the choice the classroom atmosphere changes from the lecture/exposition style to that of experimentation/investigation with students taking up a much more active role.

The students choose the dilation lines, the scaling factors and the point whose image is to be found. They are then invited to estimate where on the screen the image lies. Views may be elicited from students as to the image point and justifications sought for these views with the students themselves offering support or criticism. Once a concensus is formed the machine can be permitted to illustrate how to find the image. In this sort of atmosphere the transformation can be investigated. All four quadrants can be used. What happens with negative scale factors? What happens if the scale factors have absolute value less than 1? It may be that the teacher would wish to keep some of these questions as a follow-up activity.

Once students have a mental image of the transformation the problem of determining its matrix representation must be considered. The process for reflections is generalised and the dilation is broken down into the successive transformations

first 'unshew' the lines so that they align with the axes

then do an 'ordinary' scaling

finally shew the lines back to their original position

students need to be convinced that this combination is equivalent to the original and to get an idea of what the 'shew' transformation does. The package supports this graphically and can be used in a similar way to that described above.

Use by the student

As the package is interactive, students are first invited to recap on their understanding of rotations and reflections, particularly on the breakdown of the three successive transformations described above. The student may thus work through from this material to dilations in shew directions. At each stage he/she is encouraged to do the matrix calculations and thus compute image points. Computed values as well as graphical images are available.

Although a very specific situation has been described, two points emerge which are widely applicable.

(i) Using software, particular graphics, requires a change of classroom style to encourage experimentation and investigation. Activities to follow such use require careful consideration.

(ii) There should be an opportunity (at some stage) for the student to use the package to reinforce the ideas it puts forward. This has implications for the user-friendliness of the package.

4. WHEN THE MACHINE IS TURNED OFF

Experience within the OU has shown that students remember little of the detail from a single viewing of a half hour television program. There is no reason to suppose that exposure to a computer graphics package would yield a different response. Therefore whenever a package is used thought needs to be given to the ways by which the material can be reinforced. Unlike reinforcement of other media the computer follow up activities must guard against the possibility that the machine is doing all the work and providing all the answers. What we would wish to happen is that the machine helps the student grasp some concept and get a mental image of that concept which can be exploited and developed further in subsequent work. This section makes some suggestions for follow-up activities and although they are presented in the context of the reinforcement of the material of section 2, they have merit in situations unrelated to computing.

(i) Further examples

This would be a tutorial tupe session where students are given access to the package and can explore the effect of changing all the available parameters. Once the matrix of a skew dilation had been found the package could be used by the student to verify his/her own calculations.

(ii) Investigation

This is a more challenging way of achieving the objectives in (i). It can also be much more stimulating for the student and teacher. The investigation may be specified quite tightly by a sequence of structured questions or if students are already familiar with processes which aid mathematical investigation a more open-ended session could be attempted.

A possible outline for a skew dilations investigation might be:

(i) Draw rough diagrams which show images of the following points (using the computer to help if necessary) (list of points is specified)

(ii) What happens to points on the skew lines? Take some special cases and conjecture a result.

(iii) Find the image of the following line by finding images of particular points on it and generalising the result. (Line would be specified.)

(iv) What can you say about the image of a pair of parallel lines?

(v) Investigate images of things other than lines.

(vi) What happens if one of the dilation factors is zero?

(vii) Can you say anything about successive dilations in skew directions?

(The processes, introduced here, of generalising, specialising, conjecturing and proving are given a fuller treatment in [4].

(iii) Reconstruction

This is a simple technique where students are asked to write down all the technical terms they encountered and then to weave these terms together into an account of what the topic has been about. Work can be done individually but small group working is probably preferable so that several minds can respond to the memory activity. Each group would then give an account to the other groups who would contribute amendments. The principle behind reconstruction is that when the group has constructed its account, that reconstruction is internalised and can be reused at a future date. The role of the teacher is as listener and motivator and only as a last resort as a source of information. Pertinent questions like 'what is the role of the skew lines when drawing sketches of image point?' may help unlock a students memory. However, a danger to be guarded against is the possibility that a reconstruction may be internalised which is in fact in error.

4. CONCLUSIONS

Experience within the OU would suggest that those teachers responsible for the development of a piece of teaching material which uses a variety of media are more likely to use all the media used whilst those who are not involved in the development may need a lot of encouragement before they use all the materials at their disposal. In the context of computers as teaching aids, reasons for their reluctance include uncertainty about hardware and software but perhaps most important that of time. To be effective the classroom use of a computer package must be carefully planned as must the activities which follow it. This preparation is a very high overhead for which many teachers cannot find the time.

REFERENCES

1. The Royal Charter of the Open University, Article 3.
2. M101 "Mathematics Foundation Course": The Open University 1978.
3. M101 "Mathematics Foundation Course": Block IV Unit 4, The Open University 1978.
4. J.H. Mason, Learning and Doing Mathematics, The Open University 1984.

COMPUTER EXPERIENCES AS AN AID IN LEARNING MATHEMATICS CONCEPTS

Ed Dubinsky
Clarkson University
(Potsdam, New York 13676, USA)

As a teacher of post-secondary mathematics courses for the last 30 years, I have never felt very successful in teaching abstract concepts, except in the case of those students with a special talent for mathematics. Of course it is possible to train students to exercise various mathematical techniques, and even to apply these to phenomena in the physical world. But if one is speaking of understanding concepts such as composition, induction, linear independence, compactness, limits, continuity, homomorphisms, etc., then even with students who are quite successful in all of their other subjects; in mathematics, experience suggests that they do not learn these ideas. There seems to be general agreement on this point (see, for example, [6]).

Any serious attempt to alleviate this situation would have to include at least two kinds of activities. First, theoretical investigations are needed to explain what is going on in the mind of a student when he or she is trying to learn concepts as sophisticated as the above and second, if the standard approach of lectures, exercises, recitations and tests is not working then new methods will have to be devised, evaluated and implemented.

Although there is a vast literature on these matters relative to concepts in elementary, secondary and even early post-secondary education, there seems to be relatively little study of learning concepts in undergraduate mathematics. There is some work regarding representation (e.g., [5]) a lot of material on teaching problem solving (for example [10], [11]) and perhaps a little about the use of computers for discovery learning on this level ([1], [2]).

In this paper I would like to describe three on-going projects which attempt to provide genetic (that is, developmental) data on the evolution of some of these concepts in the minds of undergraduates and at the same time use computer experiences as an integral part of the learning process. The approachis different from discovery learning or computer assisted instruction in the usual sense.

In the first section the general approach and some of the ideas behind it are discussed. Then I describe the projects: an experiment to test the effect of using experiences with UNIX in teaching function definition and composition; observations on the development of the concept of induction in undergraduates; and a full semester course in which a very high level programming language, SETL, is used to help students develop mental images to represent various concepts in discrete structures. Next I present some of the partial results obtained so far and, finally, there is some discussion of these results and a description of how I think all of this could be used in an integrated system for teaching abstract concepts in mathematics.

THEORETICAL CONSIDERATIONS

Although it is premature to think of the ideas expressed here as representing a completed theory that can be applied to the cognitive issues I am raising, I do believe it is possible at this stage to make some points suggested by genetic data which has already appeared, to see how they relate to subsequent data and to use them as a guide in designing further studies.

My starting point is Piaget's theory of cognitive development ([8]). Although many authors suggest that according to this theory the development ends with the attainment of formal operations during adolescence, Piaget himself is quite clear in his position that "natural thought is a ... hierarchy of levels ... each of which corresponds, in ... adult intelligence to successive stages of which it is the ... stratification. Thus it is never complete ..." [3]. He also admits at least the possibility that many adults never achieve formal operations, [9]. Finally, statements like, "from the psychological point of view, new mathematical constructions proceed by reflective abstraction" [3] can be taken as an invitation to attempt to accomodate this theory to the phenomena of learning abstract mathematics.

This is not the place for a full analysis of the notion of reflective abstraction which is one of the key ingredients of Piaget's theory. Suffice it to say that it is a mental process for acquiring concepts by building new cognitive structures out of structures already constructed. It has many characteristics including: becoming conscious of old structures that were used implicitly; reconstructing old structures by abstraction and generalization; coordinating several several of these reconstructed structures to form a new system; and applying this new system to phenomena that could not previously be assimilated.

In studying the development of a particular concept from this point of view, there are two things to consider. First is the genetic question of how this concept decomposes into simpler structures which the student already has. One cannot simply pick one of the possible logico-mathematical decompositions because as pointed out by Piaget [3,7] in the case of the concept of number, the psychological decomposition may be quite different. In the description of the induction project below one example of how this question might be investigated is shown.

Second, there is the pedagogical question of what might be done to help the student through the various steps in the reflective abstraction process of understanding a concept. As a teacher, I consider this the main issue and I feel that major efforts should be made to develop effective methods. The projects described in the remainder of this paper are intended to be a contribution to such efforts.

Regarding this second consideration, it it necessary to look a little more closely at the process of acquiring a concept. Often the last step in completing the construction of a cognitive structure is to apply it to various phenomena. At this point the structure may only be used implicitly and in order to proceed with the development of higher structures it maybe necessary to become conscious of the older structures. Thus, in induction the subject uses the structure of logical necessity to derive $P(n+1)$ once $P(n)$ is known. However, in order to understand that the method proves $P(n)$ for all n, it is necessary to be aware of the total structure $P \Rightarrow Q$ and to realize that $P(n) \Rightarrow P(n+1)$ is an application of this structure to infinitely many situations. Only then is it possible to think about $P(1) \Rightarrow P(2)$, $P(2) \Rightarrow P(3)$,... which is a family of implications indexed by n.

I believe that these two activities - using the concept implicitly in concrete situations and becoming conscious of the concept - are very difficult to induce when one is concerned with abstract mathematical concepts. The latter cannot be achieved without having done a fair amount of the former and, since the concepts are abstract, the only useful way to exercise them is by the manipulation of mental images. This is, I believe, the main difference between the mathematics professor and the students. When the professor lectures, words and chalk are used to refer to mental images which he or she possesses. Unfortunately, the student very often does not possess such images and only hears words and sees chalk. Nothing happens that induces the student to build and use mental images.

The situation is somewhat better if there exist problems involving calculations that exercise the concept. This is actually quite rare and even when present, the result is often that a student simply memorizes an algorithm for solving a specific kind of problem and does not come to any new understandings. For example, in the case of composition of functions, one can give students plenty of practice in calculating $f \circ g$ by substituting $g(x)$ for x in $f(x)$ and they will learn to do this in formulas.

Unfortunately this activity does not help the students to encapsulate the function process and make it a cognitive entity that can be manipulated. In particular, it is hopeless to ask students to compose functions for which substitution is not possible.

It is at this point that computer experiences can help. There are many examples in which specific activities in writing and running programs or even just deciphering syntax serve as manifestations of quite abstract concepts and can be used by the student as an aid to thinking of a process as a structured whole. One way to use this for teaching a particular concept is to first make sure that the students have had and remember the computer experience. Then as the concept is explained (using the experience as an example) the students are explicitly advised (drilled?) to use the experience to form a mental image and to work with the concept in terms of that image.

For several years I have been collecting examples of these experiences and using them ad hoc in various courses where appropriate. It seems to help. At the very least, when a student says to me, "I simply don't know how to get started on this question", I am able to recall a specific computer activity and suggest that the student try to relate that process to the problem in question. Invariably this helps the student to begin thinking.

More recently I have been involved in three projects in which there is an attempt to employ this approach systematically for specific concepts (or groups of concepts) and to evaluate the results. The remainder of this paper is concerned with those projects. All of them are presently in progress so this discussion will not be complete. Subsequent papers will contain a full report on each project.

UNIX, FUNCTIONS AND COMPOSITION

There are two concepts in this project: creating a function (by definition, selection from a class, composition, etc.) and composition as an operation firmed by a succession of two operations (as opposed to a simple substitution in a formula). The subjects were a group of 34 first year students at Clarkson University. They formed the entire population of a math lab course (not a required course). The experiences took place on DEC PRO 350 personal mini-computers operating a version of UNIX called VENIX.

In VENIX it is possible to define a new command to be any sequence of commands and it is possible to pipe two commands together so that the output to the first is given as input to the second. A software package was developed so that subjects with no knowledge of the machine or VENIX could practice with these operations using a long list of examples of commands that included text manipulation as well as simple algebraic expressions. There was the possibility to perform commands, to define a new command by selecting from a class of commands, to pipe two commands together, and to define a new command to be the command obtained by pining two commands together.

All students were given a pre-test to determine how much they already knew about functions and composition. Then they were divided (arbitrarily) into an experimental group and a control group. Each group was given 4 hours of practice with exercising various function definitions and composition. The experimental group used the software package interactively at the machines and the control group worked with substitution examples using pencil and paper in a standard classroom situation. Everyone was brought together for a traditional 2 hour lecture on functions and compositions. Finally each group had a 2 hour session in which connections were drawn between the experiences during practice and the ideas expressed in the lecture. In this session the students were repeatedly advised to develop and use mental images (taken either from the computer activities or their calcuations) in thinking about these ideas.

A post-test was administered and the results analyzed to see if they indicated that thinking about defining UNIX commands and piping two commands together to form a third helps students to understand functions and composition.

MATHEMATICAL INDUCTION

The initial goal of this project was to use experience with while-loops in teaching induction to an advanced calculus class of 18 third and fourth year mathematics majors with varied computer backgrounds. Later the emphasis shifted to gathering genetic data on the development of the concept of induction in these students.

The students were asked to read the (rather brief) description of induction in their text and to do one very simple induction proof for homework. Then, in class they were offered the following analogy to think about when trying to use induction.

> Assume that you have written a very large program which contains an infinite loop and have asked me to find it. After some time, I point out the following code in your program,
>
> ```
> N := 1;
> while P(N) do   $ P(N) is a boolean expression
>     N := N+1;
> end while;
> ```
>
> Now think about what I must say (regarding P(N)) to convince you that this is an infinite loop.

Two additional, more difficult, induction proofs were assigned for homework. Then each student was interviewed about induction. They were asked to explain the method; to explain why, after making such a proof, one could be sure that the statement was true for any specific N; to describe any mental images that they used in thinking about induction; and finally, to work a problem.

The transcript of the interviews showed that the students could be divided into three groups as follows:

I. Did not have a completed concept of induction,

II. Had a completed concept of induction but did not use it as a strategy in making a particular proof,

III. Had a completed concept of induction and used it as a strategy in making a proof.

It was then decided to analyze further the interviews of the students at stage I in order to obtain genetic data on the development of the concept of induction. This analysis gave rise to several substructures which needed to be present and then linked together in order to form the concept of induction. It was possible to partially order these structures in such a way that if A < B then B did not appear without A also being present. The substructures are such that it is reasonable to consider induction as being logically composed of them and this in fact may well be the actual psychological decomposition.

SETL AND DISCRETE STRUCTURES

This project is, in several ways, the most extensive activity described in this paper. A full semester course was designed to implement the ideas about using computer experiences that we have discussed. It is now a regular course at Clarkson University and has been given twice. A version of it will also be given in Fall, 1985 at Dickinson College.

The SETL programming language implements many of the basic constructs of discrete mathematics. These include set-formers, existential and universal quantifiers, vectors, sequences of finite but arbitrary length, relations (finite set of ordered pairs) and maps on finite sets. Moreover, the notation is very close to standard mathematical notation. For example, most mathematicians will guess very quickly, if told that S, T are vectors (of unknown dimension), that the following SETL expression

```
#{I : I IN {1..#S} ST (EXISTS J IN {1..#T} ST S(I) = T(J))}
```

represents the number of components of S which have the same value as some component of T.

Another feature of this language is that, because of these powerful constructs, the use of standard Algol-like syntax in general and the fact that data types need not be declared, it is quite easy to write programs in SETL. At the beginning this means that students enjoy learning the language and fairly soon they are able to program quite sophisticated expressions and algorithms.

The course proceeds by having the students learn to program in the SETL language. From time to time this process is interrupted for one or more class periods by a discussion of a particular mathematics topic consisting of one or more concepts for which the programmming activity provides useful experiences.

These experiences occur in two different ways. The first is simply the activity of using SETL syntax for a particular construct. For example, consider the following SETL statement format which forms or constructs a set and establishes the value of the variable A to be that set.

A := {expression IN x: x IN S ST boolean expression IN x};

Here underlined words are key words in the language and S must be a finite set (either previously constructed or actually an explicit set former itself).

The students are advised to think of a set in terms of its actual construction by the computer as it is in the process of running a SETL program which contains an expression that forms the set. Thus it is strongly suggested that they form some mental image of the set S (early on, one uses something quite specific for S such as the integers from 1 to 100) and to imagine the computer calculating the expression for each x in S and then testing the Boolean expression. If the test is successful, the value of the expression is placed in the set A, otherwise it is ignored. It is helpful to talk discuss in class something about the representation of SETL values in the machine so that students can think about this process even more concretely.

The course then goes into elementary set theory (unions, intersections, cartesian products, etc.) and the students are constantly reminded of the value of

thinking of these ideas in terms of mental images such as the above.

The second way in which the experiences are used is through writing specific programs. For example, the students were given the problem of evaluating choices of ingredients in manufacturing a certain chemical. There were several ingredients and for each there was a list of possible materials. The materials had various properties and certain combinations of attributes had to be satisfied. In their program to find valid selections, the students had to use existential and universal quantifiers. The possibility of using the SETL constructs that implement these quantifiers directly rather than detailed loops helped the students to construct concepts of quantification.

Aside from attending class, the main activity of the students was contained in a large set of homework problems (about 50) of which 10 were programs (some, such as a data base problem were quite long). These formed the main vehicle for students to have experiences. They also provided drill to tie down some of the ideas and challenges for the brighter students. Class activities included explanation of concepts and exhortations to develop and use mental images (based on the SETL experiences) in thinking about the concepts.

It turns out that many important concepts are amenable to this approach and I will mention the main ones that are discussed in this course. In additon, there will be a second semester (starting in 1985) that will try to discuss more sophisticated concepts, but this will be described elsewhere.

The set former described above turned out to be very useful. The formal notation and its connection with a concrete activity performed by the computer led to the development of useful mental images of set construction. Also, the three separated parts - expression, domain specification, boolean expression - helped clarify thinking. The expression in x followed by explicit mention of the set from which x comes provided for many students their first understanding of the domain of a variable which appears in a function. Finally, using this set former notation to describe complicated situations (e.g. the median of a set of numbers, the set of letters which appear twice in a word, etc.) helped students develop their skills in simple modelling and global thinking.

Students have a lot of trouble negating Boolean expression, especially when they are quantified. In SETL one can write statements such as

<u>FORALL</u> x <u>IN</u> S <u>EXISTS</u> y <u>IN</u> T <u>ST</u> <u>FORALL</u> z <u>IN</u> U | P(x,y,z)

(here S, T, U are sets and P is a Boolean expression), and indeed the nesting can be to any depth. In a typical problem the students are asked to program a logically complicated statement in English. Of course this simply amounts to expressing it in the notation of symbolic logic such as the above. They can negate it in the program by prefixing it with the word, <u>not</u>, or they can develop a more detailed statement by using a specific algorithm. This is then translated back to English. All the time the students can think about the operation of the program but eventually they are asked to go directly from the English statement to its negation in English. As a teacher I have few greater pleasures than come from the experience of asking the students to do this in class and then look at the strained expression on the faces of a group of 50 individuals and literally watch the process of construction of the mental structures. I sometimes feel like the Kansas farmer in August who goes out to the fields at dawn and listens to the corn grow.

An entire unit in the course is devoted to operations with vectors and matrices. SETL implements the compound operator construct of APL so these operations are not only easy to program but the result is quite close to mathematical notation. For example, the result of applying the N x K matrix A to the vector x looks like

[+/[A(I)(J)*x(J): J <u>IN</u> [1..K] | I <u>IN</u> [1..N]]

The students learn to translate this to

$$\left(\sum_{j=1}^{k} A_{ij}x_j\right)_{i=1}^{N}$$

and eventually to skip the intermediate SETL notation. In the end the students are asked to prove that matrix multiplication is associative.

The last major concept considered in the course is relations. In SETL a relation is a set of ordered pairs (vectors of dimension 2). Domains, ranges, inverses and compositions now all have concrete manifestations in SETL programs and the students use these to develop their mental images.

At the end of the course each student is interviewed. Various questions are asked about these concepts and the student is requested not only to give the answer but to describe the thought processes. The main purpose of these interviews is to see if the students are, in fact, acquiring these concepts. A second purpose of the interviews is to generate genetic data on the development of these concepts.

Most of the students in this class will go on to take a standard course in Discrete Structures and Applied Algebra, which is a required course for majors in Mathematics or Computer Science at Clarkson. Thus there will be many students who did not take the SETL course. A comparative study is planned to see if this experience has a measurable effect on performance in the second course.

RESULTS OF THE THREE ACTIVITIES

The test results on the UNIX experiment must still be subjected to statistical analysis in order to determine significance, relation to standard predictors such as SAT scores, and comparison with performance in other areas. It is possible however to make some remarks about the raw data and to observe that the results appear to be rather striking.

Overall, the average score of the experimental group (who had the computer experiences) was more than 50% higher than the average of the control group. The experimental group did better than the control group in each of the seven question areas although in two of them the difference was small and may turn out to be not significant, statistically. In 4 of the question areas there were three types of problems. First a problem involving functions (similar to those emphasized with the control group of the kind usually discussed in calculus and whose solution involved substitution of variables. Here there was very little difference between the two groups. Second there was a problem involving functions similar to those used in the computer experiences (which had also been discussed briefly with the control group) and requiring the student to think about the action of the function. In this case the experimental group's scores were 40% higher than that of the control group. Finally there was a problem using functions unlike anything that had been discussed with either group but also requiring the student to think about the operations. This time the experimental group's scores were again 40% higher.

It should also be mentioned that on this test, which colleagues have suggested is rather difficult, the experimental group's overall average was about 61%. The exam did not count for their grade in the course nor were they expected to study for it. It was clear from the time spent and the scratch work shown that both groups took the exam seriously.

Finally, although half of the experimental group was taught by me and the other half by the regular instructor, there was little difference in the scores of these two sections. All of the control group was taught by the regular instructor.

The results on the induction project were much less positive. In the first place, the overall performance was distressingly poor and in particular almost no one referred to the while loop analogy in explaining how they thought about induction. Of the 18 students, 6 were in the first stage (described above), 9 were in the second stage and only 3 were in the third stage. Of the latter group, only one actually succeeded in solving the problem.

On the other hand, much of the information provided by the interviews was quite interesting. The definition of stages and membership therein was quite sharp (three different people read the interviews without much disagreement on this) as was the

relation of the 6 students in stage I to the sublevels percieved in this state. Also, the difference in understanding of induction displayed by the students in different stages was much wider than one might hope for with individuals in the same course who have studied the topic.

Regarding the SETL course it will be some time before detailed results are available. The interviews must be transcribed and analyzed while the longitudinal study will of course last for several months at least. The only kind of results available are the comments of the students which are quite favorable and my reaction as the teacher which may well be due to various factors other than the methods described here. I do feel that student activity (in class and with homework) is at a higher level than normal and I would expect the data to indicate that they learned a great deal.

DISCUSSION

None of the projects described in this paper has reached a sufficient state of completion to warrant any definitive conclusions. It is possible, however, to make some provisional remarks and in particular to indicate some directions in which the research will continue.

The experiment with UNIX suggests that this particular way of using computer experiences can be helpful with teaching functions and composition. Statistical analysis of our data and repetition of the experiment should confirm and quantify this affect. The subject matter seems particularly appropriate for advanced high school students so our next project may involve this group. We will attempt to develop a package that can be used by a high school or college teacher as a unit on function definition and composition. It should be useable in either a classroom or learning laboratory situation.

Even a superficial glance at our data raises some interesting questions regarding composition. Three of the seven questions involved giving the subjects two of the functions F, G, H in the formula H = F∘G and asking them to find the third. It turned out that both groups did well (about 75%) in finding H or F but not so well (33% for the control group and 63% for the experimental) in finding G. Also, in the four sub-questions that involved functions unlike any that had been shown to either group, they were much more successful in thinking about arithmetic calculations with triples of integers than about rigid motions of a square. It would be interesting to analyze these observations from the point of view of cognitive development. It may well be that the emphasis at Clarkson on numerical calculation has something to do with this effect.

The project with induction appears in a much different light. In the first place, the classroom discussion of while loops did not seem to have much affect. It may be that explicit, recent activities with the computer experience will work better. It may even be necessary for those computer experiences to involve directly the concept being learned. On the other hand, looking at the three stages, it seems that the while loop image would be helpful only with people at stage I. It is possible, although there is no evidence for it, that the people in stage II got there by thinking about while loops. This would explain why stage II has the largest population but it would not explain why they don't mention this analogy in the interviews.

In any case, the people in stages II and III need something very different to help them learn induction. In fact, the most striking reaction that I have to the results of this project is that if one wanted to teach induction to this group of students any <u>single</u> activity would be largely a waste of time for students in two of the three stages.

In future work on this project the same students will be interviewed on compactness (which was a major topic for them during this semester). My expectation is that there will be a similar decomposition into stages and the students will be easy to assign to a particular stage. At this point there will be an attempt to analyze this genetically and use it to advance our understanding of how undergraduate mathematics students construct the concepts which it is necessary for them to acquire.

As we move into more of the abstract concepts in mathematics, it may become increasingly difficult to find appropriate computer experiences. This is an area that will require some creative activity and also, as we grow in our understanding of how the cognitive development takes place, we may find that experiences which do not relate to computers may be helpful in the same way.

Although there is no data on the SETL course yet available, it can at least be reported that this approach makes for a lively course in which the students are responsive in class and active outside of class. In comparison with similar groups to whom I have tried to teach this material, these students seem to be more prone to speak in terms of sets and less confused by complicated logical statements. They like the material and seem to enjoy working with it. On the other hand, the task of integrating all of this into a coherent course is not trivial. The issue of how far the computer experience can be removed (in time and awareness) from discussion of the concept arises here (as it did with while loops) and is particularly important in designing the specific operation of the course. The students in this course did not do as well with the study of relations, their inverses and their composition. This is more difficult, it came later in the course and although there was ample <u>verbal reference</u> to computer analogies, there were no actual computer <u>experiences</u> directly related to relations.

It is possible to begin to perceive an emerging overall approach to teaching concepts in undergraduate mathematics. Broadly speaking, the projects described here contribute to the study of this approach as, respectively, an effective package for teaching one topic, a study of the genetic decomposition of specific concepts and the integration of these ideas in a single course.

Here is how I see the overall approach at this, provisional, stage. All of the concepts in undergraduate mathematics should be analyzed and their genetic decomposition determined. The curriculum should be rearranged not in terms of courses but in terms of the interrelationships amongst the structures of which the various concepts are composed. Appropriate methods, including the use of computers as described here, should be developed to induce the acquisition of each structure. Students should be constantly tested and interviewed to determine which concepts they already have, which structures they should work on next and which alternative methods are most appropriate for them. Individual courses and classes of students should be loose and changing, putting together temporarily those individuals working on the same thing with the same method.

There are many difficulties with the development and implementation of such an approach. The most obvious is the amount of research required before one could even begin to think of using it on a large scale. The development would be "circular" in that the testing and interviewing of students would serve not only to assimilate them to the approach but also to accomodate the approach to our growing experience with students learning these concepts. The perpetrators would reflect on what was going on and continually revise the methodology on higher planes. Hopefully there would be periods of equilibrated tranquility during which time one could think about evaluating what had happened and disseminating the results.

Another difficulty is that, however important, concept acquisition is not the only thing that is required in studying undergraduate mathematics. Problem solving, techniques, modelling and applications are all essential and may well require entirely different approaches which would have to be incorporated.

In spite of all the difficulties, the experiences that I have had so far convince me that the ideas behind this approach are sound and that implementation is not only feasible but will lead to important, measurable results. At the very least I have demonstrated in this paper that it is possible to develop and use the approach piecemeal so that the total system described above can be instituted gradually. I propose to continue this and I invite others to join me. Eventually, there may come an opportunity to attempt an implementation of the entire approach in one place. I would hope that all of this will make a contribution to teaching, to learning and to the understanding of both.

ACKNOWLEDGEMENTS

The UNIX experiment is being done jointly with T. Ayres, G. Davis and P. Lewin. Also, P. Lewin is collaborating on the induction project. I would like to thank G.T. Klein for many useful discussions on all of these projects and for assistance in revising this paper. The interview were transcribed by S. Pregger and the paper was typed by C. Martin.

REFERENCES

1. Abelson, H., Computation in the Undergraduate Curriculum, Int. J. Math. Educ. Sir. Tech. 7 (1976) 127-131.

2. Abelson, H. and DiSessa, A., Turtle Geometry: Computation as a Medium for Exploring Mathematics, Cambridge MIT Press (1980).

3. Beth. E.W. and Piaget, J., Mathematical Epistemology and Psychology, Gordon and Breach (1966).

4. Iverson, K. APL in Exposition, APL Press (1976).

5. Kaput, J.J., Representation Systems and Mathematics (preprint).

6. Moise, E.E., Mathematics, Computation and Psychic Intelligence, in Computers in Mathematics Education, NCTM (1984) 35-42.

7. Piaget, J., The Child's Conception of Number, Percy, Lund, Humphries And Co., (1941).

8. Piaget, J., Piaget's Theory, in Manual of Child Psychology by Musser Carmichael, Wiley (1970).

9. Piaget, J., Intellectual Evolution from Adolescence to Adulthood, Human Development 15 (1972) 1-12.

10. Polya, G., How to Solve it, Doubleday-Anchor (1954).

11. Schoenfeld, A. H., Teaching Mathematical Problem Solving Skills, Department of Mathematics, Hamilton College, Clinton, NY (1979).

A NEW DEGREE IN MATHEMATICAL STUDIES

Ramesh Kapadia,
Polytechnic of the South Bank,
LONDON.
SE1 OAA.

This paper addresses the first of the questions posed by the orientation paper for the ICMI Symposium on The Influence of Computers and Informatics on Mathematics and its Teaching. Details of a new degree in mathematical studies, motivated by the introduction of computers and informatics are presented.

After giving background information, the vocational and academic aims of the course are discussed. An outline course structure and curriculum is then presented, followed by detailed syllabuses for a few selected units. The course was developed by a small planning committee at the Polytechnic of the South Bank; it has been formally approved and the first intake of thiry students commenced in September 1984.

1. Background Details

In England children complete a general course of study at school at 16 years when they take a number of subjects at Ordinary Level. They spend the next two years specialising in three subjects to be examined at Advanced Level prior to commencing an undergraduate degree, which takes three or four years. Students studying for a mathematics degree must take mathematics upto Advanced Level; the other two subjects are usually Physics and Chemistry, but this pattern is now changing with many students taking subjects from the arts side while a few take Computer Science.

A student may study for a degree at a University or a Polytechnic. Universities are autonomous centrally funded institutions, awarding their own degrees. Polytechnics were created to develop more vocational courses, which are validated by the Council for National Academic Awards (CNAA). Polytechnics are funded on a more local basis and specifically develop courses to satisfy local needs. An important feature of Polytechnic degree courses is that they are sandwich courses to include one year of industrial experience within the four years of study. Thus the degree in mathematical studies at the Polytechnic of the South Bank consists of two years of full-time study, followed by a year in industry and the final year in the college. This vocational element is an imporant feature of the course and was one of the main reasons that it was decided that the degree in mathematical studies had to take account of the influence of computers and informatics.

2. Vocational Aims

The pattern of demand for skilled personnel in the areas of information technology and management support services is becoming stabilised. The overall aim of the course is, to equip graduates with the educational background and professional skills that will enable them to begin careers in information systems, management support services, software production and resource management.

Although the level of demand for staff in those areas will certainly increase, specific requirements are likely to change as technology moves forward and techniques and practices are improved. The professional skills that would have served a graduate well enough in, Data Processing, no longer provide an adequate foundation for professional employment in the later 1980's. Therefore, the course needs to reflect the changes that have taken place and that are likely to take place in the immediate future. It is also important to reflect the need to introduce mathematical rigour to the teaching of computing subjects, the orientation of applied mathematics to computer based applictions and the importance of software engineering and knowledge processing to the development of information systems.

Although new directions in information technology are represented, there are elements present that are not so susceptible to changes as the 'harder' technical aspects of computer hardware and software. These elements include organisational theory, the role of information and resources management as well as established mathematical and statistical methodologies. Material of that kind remains central to the processing of training graduates for their future roles in commerce and industry. For example, although the technology for information handling may (and does) advance in sophistication, it remains true that the systems analyst or management services professional can only fully exploit that technology if there is a fundamental appreciation of the nature of the information as an organisational resource. Similarly, a knowledge of communications technology needs to be complemented by an understanding of the benefits offered to the organisation that can control and utilise distributed information resources. There are numerous parallel examples that could have been quoted.

To summarise, the course aims to produce graduates who are fully aware of the needs of the modern organisation, are trained in the necessary skills to serve these needs and are equipped to take their skills in the Information Technology and Management Support Services teams of those organisations.

The overall vocational aim of the scheme is to strike an appropriate balance between generalised and specialist knowledge. Students are given a solid background in Management Information Services which enables them to:-

(i) Understand the problems of locating and acquiring appropriate data.

(ii) Formulate the mechanism for processing the data to provide the information required by an organisation.

(iii) Specialise in a particular area of Management Information Services and integrate that specialism within a multidisciplined team.

(iv) Communicate their knowledge to others.

3. Academic Aims

In addition to recognising that mathematical and problem solving techniques are finding wider applications in commerce, the public services and industry, the course attempts to address the fact that a mathematical sciences degree is becoming increasingly recognised as the mark of a rigorous intellectual training and is therefore more acceptable for entry to fields in which logic, organisation, system and judgement matter. The course aims to establish an appreciation of the need for rigour and appropriateness of abstraction through the provision of units in Algebra and Discrete Mathematics. The units have replaced Mathematical Analysis as the vehicle for these concepts which we regard still have the hall-mark of the mathematical scientist.

The course aims to establish a sound theoretical and practical foundation in the broad area of mathematical sciences. The first two years provides a comprehensive introduction to the areas likely to be useful in a commercial, governmental or industrial environment. The student is therefore equipped for his/her year in vocational training.

The final year introduces a degree of choice which enables the student to pursue his/her particular interests to a greater depth and to achieve some degree of specialism. The course units currently on offer aim to provide a wide range of choice that stretches right across the perceived needs of the management services from a theoretical understanding of elements of computer science to advanced data analysis.

In particular, students on the mathematical studies route will:-

(i) Understand the basic theoretical and abstract aspects of mathematics.

(ii) Have a sound knowledge and command of the basic skills in the main subject areas.

(iii) Be able to apply these skills to real problems.

(iv) Have an appreciation of the computational aspects of the main subject areas and have a working knowledge of at least one high level language.

(v) Understand the interrelation between the organisation and the mathematical sciences.

(vi) Have the necessary background of systematic problem solving, coupled with theorem proving skills in order to be able to adapt to future changes in the field.

4. Course Structure and Curriculum

To achieve the academic and vocational aims of the scheme it is recognised that the structure must facilitate an appropriate balance of general and specific areas of study. Further it must provide the necessary integration between subjects to form a coherent programme of study so that the student is able to place the subject into the context of the course. To this end the scheme is based on a series of units each with defined objectives. Care has been taken to ensure that the units in each year of the course form a sensible mix and do not appear to the student as a disparate collection of topics.

Each unit represents 33 hours of class contact allowing, where appropriate, flexibility of teaching method. In addition each unit is allocated extra hours which may be tutorial, practical, seminar, case studies or revision class etc. Examples of the additional time allocation are given in the syllabuses to illustrate the approach in the scheme to move from traditional teaching methods to student based learning.

Part I

The first two years provide the supporting studies of the year in industry and subsequently to the specialisms of the final year. In each year the student will take a total of eleven units commencing from an introduction to the basic concepts through to the study of rigourous disciplines. Discrete Structures is seen as the foundation for the study of Programming Methodology and formal approaches to System Design. The units of Algebra are seen as the vehicle for developing the necessary level of abstraction and rigour.
At the end of the second year it is expected that the student will be equipped with the applicative skills and basic knowledge necessary to take full advantage of the vocational year and to rapidly contribute productively to the placement in industry. In preparation for this the core unit Project Implementation is designed to give the student

the opportunity to work within a multidisciplinary team on a substantial piece of practical work and to develop effective communication skills necessary for such work.

The course units are listed below:-

Year One

Mathematical Methods I and II
Business Studies and Economics
Complementary Studies
Programming I and II
Discrete Structures I and II
Statistics I and II
Algebra I

Year Two

Operational Research I and II
Business Systems
Complementary Studies
Project Implementation
Introduction to Economic Modelling
Mathematical Methods III
Algebra II and III
Statistics III
Statistical and Economic Application

Part II

The final year of the course gives the student the opportunity to study, in depth, particular specialisms which are selected in view of the previous years in industry and with advice from academic staff. To achieve the depth necessary for final year honours some of the units have been linked to provide an extended study. Students will be expected to study a total of six course units comprising of a compulsory linked pair, a selected linked pair and two other units. Currently the following units are offered:-

Formulation in Mathematical Modelling
Mathematical Models for Industry and Commerce.
Stochastic Processes I and II
Mathematical and Numerical Analysis I and II
Econometrics I and II
Data Analysis I and II
Mathematics of Computer Science I and II

Logic and Computability
Applied Combinatorics and Graph Theory
Discrete Event Simulation
Probability and Statistics
Mathematical Programming

The compulsory units are designed to emphasise the prospective career profile of the graduates emerging from the scheme. The linked units Formulation in Mathematical Modelling and Mathematical Models for Commerce and Industry provide for the problem solving expertise in terms of the collection and analysis of 'dirty data' as required by the management services function of the modern organisation.

All students will undertake a major piece of work in the final year which is considered to be a culmination of the study for the scheme. They are required to submit a written report (approximately 5000 - 8000 words) of their project which is expected to be an in depth study of a particular area of interest.

UNIT TITLE: DISCRETE MATHEMATICS I

YEAR AND TERM: First Year. Terms 1 and 2.

AIMS: To introduce many of the essential ideas and techniques of Discrete Mathematics and provide the working vocabulary of basic concepts for Computer and Information Science.

OUTLINE SYLLABUS: Basic ideas of set theory.

Functions and relations.
Algorithmns and proof techniques.
Elementary number theory.
Introduction to graph theory.
Boolean algebra and propositional calculus.

TEACHING METHOD: Lectures (33 hrs), Tutorials (22 hrs).

The presentation will be strongly biased towards applications and practical problems.

ASSESSMENT: Examination (40). Coursework (10).

READING LIST:

LIPSCITZ, S.,	Discrete Mathematics.
PRATHER, R. E.,	Discrete Mathematics for Computer Science. Houghton Miffin, 1976.
FISHER, J. L.,	Application Oriented Algebra. Dun-Donnelly. 1977.
BIRKHOFF, G., and BARTEE, T. C.,	Applied Modern Algebra. McGraw Hill. 1970.

UNIT TITLE: DISCRETE MATHEMATICS II

YEAR AND TERM: Year 1, Terms 2 and 3.

AIMS: To introduce the basic concepts and structures of Graph Theory, Boolean Algebra and Logic.

OUTLINE SYLLABUS: Introduction to mathematical structure, morphisms and algebraic structure.

Graph Theory

Isomorphism. Connectivity; strong, unilateral, weak connectivity. Planarity. Euler tours. Hamilton cycles. Travelling salesman problem. Matchings. Matchings and bipartite graphs. Applications.

Boolean Algebra and Lattices

Lattices and posets. Semi-lattices. Sublattices. Direct products. Distributive, modular, geometric, Boolean lattices. Morphisms and ideals. Finite Boolean Algebras.

Logic

Informal propositional calculus. Truth functions and truth tables. Normal forms. Rules of inference and proofs. Informal predicate calculus: predicates and quantifiers. Introduction to Logic and Languages.

TEACHING METHOD: Lecture (33 hrs), Tutorial (22 hrs).

See Discrete Mathematics I.

ASSESSMENT: Examination (50).

READING LIST: See Discrete Mathematics I.

UNIT TITLE: ALGEBRA I

YEAR AND TERM: First year. Term 1.

AIMS: The three Algebra units are intended to lay a solid foundation of Modern Algebra in a rigorous and coherent way. Algebra I is intended to introduce the number systems and the algebraic structures and to initiate an appreciation of the axiomatic method. Algebra II and III study particular important structures, namely vector spaces and groups respectively.

OUTLINE SYLLABUS: Survey of number systems N, Z, Q, R, C. Discussion of the properties and introduction to the basic algebraic structures: groups, rings, integral domains, fields.

Polynomials. Division algorithm. Remainder theorem. Statement of the fundamental theorem of algebra.

TEACHING METHOD: Lecture (33 hrs), Tutorial (16.5 hrs).

The subject will be taught with many examples and illustrations to minimise any difficulties in comprehension of the abstract approach. The abstraction is desirable to allow a coherent and consecutive development of the subject. The level of abstraction will gradually rise from Algebra I to Algebra III.

ASSESSMENT: Examination (50).

READING LIST:

BIRKHOFF, G., and MACLANE, S.,	A Survey of Modern Algebra. 4th Edition. Macmillan, 1980.
LEDERMANN, W.,	Introduction to the Theory of Finite Groups. Oliver and Boyd. Revised Edition. 1981.
ARCHBOLD, J. W.,	Algebra. 2nd Edition. Pitman. 1981.

UNIT TITLE: ALGEBRA II

YEAR AND TERM: Second Year. Term 1.

AIMS: See Algebra I.

OUTLINE SYLLABUS: Linear Vector Spaces.

Linear quations.
Matrices and matrix algebra. Vectors. Vector spaces, subspaces, linear dependence. Basis and dimension. Linear transformations. Rank and nullity. Matrices and linear operators. Change of basis. Similarity. Eigenvalues and eigenvectors. Diagonalisation. Characteristic and minimum polynomials. Discussion of quadratic forms and real symmetric matrices.

TEACHING METHOD: Lecture (33 hrs), Tutorial (16.5 hrs).

See Algebra I.

ASSESSMENT: Examination (50).

READING LIST: See Algebra I.

LANG, S.,	Linear Algebra. 2nd Edition. Addison-Wesley. 1966.

UNIT TITLE: ALGEBRA III

YEAR AND TERM: Second Year. Terms 2 and 3.

AIMS: See Algebra I.

OUTLINE SYLLABUS: Isometries. Permutation groups. Cayley's Theorem. Cosets, Lagrange's Theorem. Isomorphism. Homomorphism. Normal subgroups. Factor subgroups. Isomorphism Theorem. Automorphisms. Direct products. Survey of rings and fields. Simple introduction to universal algebra.

TEACHING METHOD: Lecture (33 hrs), Tutorial (16.5 hrs).

See Algebra I.

ASSESSMENT: Examination (50).

READING LIST: See Algebra I.

UNIT TITLE: MATHEMATICS OF COMPUTER SCIENCE I

YEAR AND TERM: Final Year. Term 1.

AIMS: The aim of this unit is to consolidate the knowledge gained in the first two years in applied modern algebra, and examine in detail those areas that are applicable to computing and computer science.

OUTLINE SYLLABUS: Review of group theory. Semigroups, monoids, groups. Morphisms. Finite state machines. Morphisms, machines as recognisers. Kleene's theorem. Consideration of machine design and construction. Introduction to formal languages.

TEACHING METHOD: Lecture (33 hrs), Tutorial (33 hrs).

The subject matter will be presented from a theoretical viewpoint in order to introduce the rigorous theoretical frameworks within which ideas about computer science can be expressed.

ASSESSMENT: Examination (50).

READING LIST:

GERSTING, J. L.,	Mathematical Structures for Computer Science. Freeman & Co., 1982.
STONE, H. S.,	Discrete Mathematical Structures and Their Applications. Science Research Assoc., 1983.
FISHER, J. L.,	Application Oriented Algebra. IEP. 1977.

UNIT TITLE: MATHEMATICS OF COMPUTER SCIENCE II

YEAR AND TERM: Final Year. Term 2.

AIMS: This unit follows naturally from Mathematics of Computer Science I and has similar aims.

OUTLINE SYLLABUS: The following is a suggested list of topics not all of which will be covered at any one time. The list is not meant to be exhaustive.

1. Coding Theory: cryptography, matrix encoding. Scambled codes. Hamming matrix. Binary group codes. Equivalent codes.

2. Languages: BNF. grammars. Context free grammars. Language recognisers.

3. Polya theory of enumeration. Historical introduction. Burnside's theorem. Inventories. Applications to switching functions.

4. Linear finite state machines.

5. Finite fields. Structure and existence of finite fields. Latin squares. Fields of fractions. Polynomial codes. BCH codes.

6. Universal algebra.

TEACHING METHOD: Lecture(33 hrs), Tutorial (11 hrs).

As in Mathematics of Computer Science I, each of these topics will be presented theoretically.

READING LIST: As in Mathematics of Computer Science I plus published papers.

UNIT TITLE: LOGIC AND COMPUTABILITY

YEAR AND TERM: Final Year.

AIMS: To provide an introductory course to Mathematical Logic; to show the relevance of the subject Mathematics and Computer Science; and to introduce the notion of effective computability.

OUTLINE SYLLABUS: Formal statement calculus. The formal system L. The adequency theorem for L. Formal predicate calculus. The formal system K_L. Equivalence. Substitution. Prenex form. The adequacy theorem. Models. Mathematical systems; first order systems with equality. Group theory. First order arithmetic. Consistency and models. Algorithms and computability. Turing machines. Recursion. word-problems and sub-groups. Church's thesis. The halting problem. Undecidiability.

TEACHING METHODS: Lecture (33 hrs), Tutorial (16.5 hrs).

Whenever possible the material will be presented in an introductory manner and in areas where algebra and logic impinge on one another, for example in the consideration of word problems for algebraic systems such as semi-groups. Group theory will be used to show how mathematical systems arise as extensions.

ASSESSMENT: Examination (50).

READING LIST:

MENDELSON, E., Introduction to Mathematical Logic. Van Nostrand. 1964.

BOOLOS, G. S., and JEFFREY, R. C. J., Computability and Logic. Cambridge. 1974.

STOLL, R. R., Sets, Logic, and Axiomatic Theories. Freeman. 1975.

ROGERS, H., Theory of Recursive Functions and Effective Computability. McGraw Hill. 1968.

LEMMON, E. J., Beginning Logic. Nelson. 1965.

The effect of computers on the teaching of mathematics students

D.L.Salinger

School of Mathematics, Leeds University, Leeds LS2 9JT.

Students of mathematics will seek employment in a large number of sectors. In most of these, they will be expected to be able to use computing packages of one kind or another. Some will be in the business of creating such packages. Those concerned with research in maths, science and engineering will need to understand how such packages work and what their limitations are. More specifically, some will need to understand the limitations of computational accuracy, and the applicability and effectiveness of algorithms for solving problems.

On the other hand, certain routine skills for which mathematicians have in the past been required will come to be handled by machine. Packages which perform the routine computations in statistical analysis are widely available (SAS, SPSS, MINITAB, GENSTAT). Symbolic integration is available (MACSYMA and REDUCE), but is not yet so established. Numerical solution of differential and difference equations has opened up a huge field of mathematics but has rendered some analytical techniques redundant.

What implications do these observations have? I shall set some out below and indicate how far our practice at Leeds conforms to them.

1. *Teaching programming to maths students*

This is a prerequisite for both practical and theoretical aspects of computing. Some mathematics departments teach BASIC on the grounds of its simplicity: it is important for a mathematics student to get quickly to a stage where she or he can use the machine for calculations. However many versions of BASIC are deficient in structural complexity. We have chosen to teach PASCAL on a mainframe computer (with good terminal facilities - VDUs and a fast response time allowing inter-active computing) and have achieved most of the objectives of the course.

Experience here and elsewhere has shown that the initial stages of teaching students to use a computer require a great deal of manpower. It is important to have enough staff present during practical sessions so that problems can be quickly dealt with. Otherwise many students will give up.

2. *Theoretical computing*

What courses should be incorporated in the curriculum of a mathematics student to provide a theoretical basis for the use of computers? Here's a suggestion. In the first year a course teaching how to compute in a suitable high-level language, followed by a course on computability including a simple mathematical model of computers (Unlimited Register Machines or Turing Machines), recursion, unsolvability of the Halting Problem, computational complexity. A later course could contain more recursive function theory, the Normal Form Theorem, an abstract form of Godel's incompleteness theorem, as well as more complexity theory. Further courses appropriate for mathematicians could deal with automata theory, denotational and operational semantics, the theory of correctness of programs, feasibility and intractibility. (Some of these courses are already available as options within our mathematics degree scheme.)

3. *Differential equations*

One of the areas where most fundamental change has taken place is in the solution of differential equations. One can question how much of the traditional theory should still be taught. Differential equations of which no analytic solution is available can be solved numerically (though one can get a 'solution' which is not a solution of the original equation.) These developments seem to reinforce the need for greater theoretical underpinning of the numerical methods (convergence of numerical process, closeness of solution to that of the given differential equation). There is also a need for a firm basis of a qualitative or geometric understanding of the type of solutions possible and the way in which they vary with the varying of parameters or initial conditions.

We run a course on numerical solutions of partial differential equtions in which the stability, consistency and convergence of numerical procedures are examined by analytic techniques. It is expected that, within a few years, students will have enough confidence in programming to be able to combine the analytic techniques with numerical experimentation and thus get a real feel for the philosophy inherent in these sophisticated methods, with the consequent ability to judge whether a numerical solution bears any comparison with the actual solution required in a particular physical situation.

4. *Use of packages in statistics teaching*

This has changed the nature of statistics teaching in courses at Leeds, both to specialist mathematicians and to other students. We use a number of commercially available packages, principally MINI-TAB and SAS. Students can carry out statistical analyses on data they either type in or get the package to generate. Part of the teaching is to indicate the limitations of the packages and the dangers in not fully understanding the calculations which they can carry out. [1]

5. *Use of microcomputers for illustration*

Micros are useful for classroom demonstration. Diagrams which previously were the province of the Open University or of commercially produced film are now available to the lecturer with a minimum of effort on his or her part. Complicated surfaces, Fourier series, solutions of differential equations can all be illustrated in a way that helps students to get intuitive ideas of abstract constructs. Ideally, students would be able to use computers to produce the diagrams themselves.

6. *Move to more project-based work*

Assessing work on computers does not lend itself to formal examination. It seems reasonable to assess skills by something the student produces on the machine. Most usefully, this can form part of a project in some other area of mathematics. In a recent final year project, one of our students used a microcomputer for both numerical and graphical investigation into the optics of hydrocyclones. In doing this, he was much closer to the style of work in applied mathematical research than students are in traditionally taught courses.

7. *Computer based learning project*

A team at Leeds has produced a multiple-choice question system whcih runs under VAX/VMS. We have used it to produce questions for students to practice techniques of integration. We are also using it to assess statistics practical work. Both of these uses are running for the first time this year. We have not yet drawn any conclusions as to the efficacy of the system, though it has been used successfully in other departments. [3,4].

References

[1] E.J.Redfern, *Computers, Practicals and Packages in Teaching Statistics*, Department of Statistics Technical Report, University of Leeds.

[2] K.Tait *The building of a computer-based teaching system* Comput. Educ. 8 (1984) 15-19

[3] K.Tait and I.E.Hughes *Some experiences in using a computer-based learning system as an aid to self-teaching and self-assessment* Comput. Educ. 8 (1984) 271-278.

The Progress of Computers and Mathematical Education

in JAPAN

Haruo Murakami

Faculty of Eng., and
The Graduate School of
Science and Technology

Masato Hata

The Graduate School of
Science and Technology

Kobe University

1. Introduction

The progress of computers has been remarkable. Ever since the emergence of the first computer, the use of computers has long been restricted mostly to fast numerical computations. Today, the extensive use of computers for non-numeric operations has begun in a variety of applications. Such non-numeric operations in the field of mathematics include computer algebra allowing symbolic differentiation, symbolic integration, factorization and expansion, etc. The rapid progress of computers has been brought about by the technological innovation of micro semiconductor devices comprising computers. The astonishing speed of the innovation, and thus of computers, will soon realize a small and cheap computer algebra system as small as the present electronic calculators or hand held computers, but with mathematical capabilities as powerful as those of average first year college students at least for the above symbolic mathematical operations. The anticipation for technology in the future is an inevitable consequence of such technological progress as the appearance of electronic calculators, micro-processors, personal computers, and memories which has been witnessed in the last decade.

The extensive use of such powerful computer algebra systems is anticipated to inevitably influence mathematical education. For instance, the extensive use of electronic calculators has already influenced one part of mathematical education. If an electronic calculator is used, big numbers do not have to be calculated by hand. Thus, it is now well known that the teaching of logarithms has changed due to the emergence of electronic calculators. Namely, a great part of the course on logarithms used to be spent teaching how to calculate big numbers using logarithms. When multiplying, for example, what uased to be taught was how to use a table of logarithms, how to get an index and a mantissa for addition, and how to get the answer using the table of logarithms. In the present curriculum, this subject is not included. In addition, slide rules are scarcely used now.

As seen from this example, technological innovation can easily influence mathematical education. Therefore, it is very important to consider the following questions regarding the future of mathematical education.

What influence will computers with the above capabilities have on mathematical education ?

How will mathematical education be changed by computers ? Or, how will it have to be changed ?

This paper attempts to discuss the above questions and lead to some basic answers to those questions.

2. The state-of-the-art of computers today and their future

Before discussing the questions concerning computers and mathematical education, it is essential to establish the basis of discussion. Therefore, this chapter is devoted to reviewing the state-of-the-art of computers today in view of mathematical education. It also attempts to explore the future of computers in the same view.

2.1 The state-of-the-art of computers today

Today's computers can be divided into the following three classes according to their size, and thus their capabilities.

i) mainframe computers
ii) minicomputers
iii) personal computers

It is noted that rapid recent technological innovations tend to blur the boundary of the classes by extending the capabilities of computers in each class.

Mainframe computers are usually used as central machines in computer centers. They are shared by a number of users as time sharing systems. Minicomputers are usually shared by a smaller number of users, e.g., ten to fifty. On the other hand, personal computers are cheap enough for an individual to buy. Computers in this class are mainly for individual use. It is noted that a new class of computers to be categorized between ii) and iii) has recently emerged. The computers in this class are called super personal computers or work stations, and they can support several users at a time.

Let us now review the state-of-the-art of computer systems associated with mathematics. Computer processing associated with mathematics includes
i) numerical computations,
ii) non-numerical computations,
iii) computer graphics.

The first class of processing can be performed by any of the classes of computers given above and their performance is comparatively well known. Thus, this class is not reviewed in detail in this paper.

The second class of processing includes computer algebra. According to [1], computer algebra is defined as a part of computer science which designs, analyzes, implements and applies algebraic algorithms. This is a rather broad definition. A computer algebra system is a computer system in which algebraic algorithms are implemented. The system can also be used for formal algebraic manipulations.

It is reported that to date about 60 computer algebra systems have been developed throughout the world [2]. Most of them operate in computer systems larger than minicomputers. Such computer algebra systems include MACSYMA, REDUCE, SAC-2, SCRATCHPAD, SYMBAL, etc. Their capacity to manipulate algebraic algorithms is great [2]. In personal computer systems, on the other hand, a few computer algebra systems have been developed. Among them, muMATH is well known*.

The capabilities of these computer algebra systems include the following operations [3].

1) rearrangement of terms and symbols
2) finding common terms
3) simplification of terms, e.g., $X+2X=3X$, $X+0=X$, $X/1=X$, etc.
4) substitution of variables by numbers or other expressions
5) symbolic differentiation
6) expansion of polynomials
7) simplification of rational functions
8) finding the G.C.M. of polynomials
9) calculation of matrices and determinants
10) factorization of multivariable polynomials
11) symbolic integration
12) calculation of limits, e.g.,

$$\lim_{X\to 0}(X+1)/\sin X^2 = X^{-2} + X^{-1} \text{ etc.}$$

13) summation of (infinite) series, e.g.,

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}, \quad \sum_{i=1}^{\infty} \frac{1}{i^2} = \zeta(2) = \frac{\pi^2}{6}$$

14) solving some differential equations symbolically
15) solving some integral equations symbolically
16) handling some special functions
17) Laplace transform
18) series expansion or expansion by continuous fractions of functions
19) graphical images of functions

In addition to the above operations, some of the systems feature a facility which allows a user to make his own program to combine the basic system facilities for extensive operations. For instance, REDUCE supports a PASCAL-like language, RLISP [4], for such a purpose. These computer algebra systems have already been used extensively in physics and engineering as computational tools. At the same time, they can be used in mathematical education as they are now.

Computer graphics are used widely for CAD (Computer Aided Design) in the field of engineering. Such systems allow a fine display of two or three dimensional graphic images and editing of

---

* It is reported that REDUCE has recently been implemented in a super personal computer using CP/M-68K as its operating system [6].

the images for design. In addition. simpler and cruder graphical images can be displayed by recent personal computers.

2.2 The future of computers

The major factors which determine the capability of computers are operational speed and size of memories available. In this section, the technological trend of components for small computers is reviewed in order to consider their future. The development of small computers is anticipated to have a vital influence on mathematical education, because such computers will probably allow students to manipulate algebraic expressions with the ease of calculation of today's electronic calculators.

Fig. 2.1 shows a historical review of the number of transistors integrated in a chip. It shows that the number of transistors in ICs (Integrated Circuits) and LSIs (Large Scale Integrated Circuits) grew twice a year during the early stages and at present is still growing twice every two years. The consequence of this growth is a reduction in price, at an annual rate of 50%. as shown in Fig. 2.2. On the other hand, the operational speed of microprocessors has been increased exponentially as shown in Fig. 2.3. And also, the integration of microprocessors has grown as shown in Fig. 2.4. Therefore, Fig. 2.3 and Fig. 2.4 show that from 1972 to 1980 microcomputers were improved 100 times in their integration and 100 to 1000 times in their speed. From these figures. it may be concluded that the performance of microcomputers was improved about a million to ten thousand times in one decade.

As is obvious from these figures, one characteristic of the astonishing growth of computers is that the growth has been accompanied by lower prices as well as higher performance. There is, of course, a saturation phenomenon for every technological innovation, so that the future cannot be predicted by a simple linear extrapolation of the past. Nonetheless, even after compensating for the possible saturation for this innovation, it is still likely that the innovation will soon realize a computer with a size smaller than today's personal computers but with capabilities as powerful as today's minicomputers. If this prediction comes true, an individual will be able to have on his/her desk such a computer algebra system as is operating in today's minicomputers.

In addition, a lot of research in many countries is now directed towards the development of fifth generation computers. This type of computer is anticipated to provide superior capabilities to manipulate symbolic data. Therefore, if such a computer is realized. the above prediction to have a small and powerful computer algebra system will be more likely.

From all the evidence. it is now concluded that a powerful and cheap small computer algebra system will probably be produced in the near future.

3. The basic purpose of mathematical education

In this chapter, let us consider the basic purpose of mathematical education as this forms the basis of the entire discussion of this paper.

The basic purpose of mathematical education may be divided into two objectives, i.e.,

i) acquisition of mathematical knowledge and computational skills,
ii) acquisition of the capacity for mathematical (logical) thought.

It is noted here that the two objectives are not acquired independently but probably obtained through the entire process of mathematical education. The classification made above is based on what can be obtained after mathematical education.

The first objective includes:

i) the mathematical knowledge and computational skills which are often used in daily life and thus essential for everyone.
ii) the mathematical knowledge and computational skills which are necessary to pursue higher education and thus essential only for those concerned.

The characteristic of this objective is that the contribution of an instructional topic to the acquisition of a knowledge or a skill is generally clear. For instance. a topic on quadratic equation directly contributes to the knowledge of the equation itself and to the computational skill required to solve it.

For the second objective, on the other hand, the contribution of a topic is not as clear as in the first objective. Rather it may be stated that a topic does not contribute directly to a particular goal but that the second objective is naturally fulfilled through the experiences of solving many exercises while learning mathematics systematically.

As is now clear, the purpose of mathematical education itself is multifold. Therefore. the differences between the purposes must be rigorously taken into account when considering the influence of computers or the methodology of introducing computers.

In addition to the differences between the purposes. students' ages and their educational environments are also varied. Namely, the multiformity includes

i) elementary education.
ii) secondary education.
iii) higher education.

Moreover, iii) can be divided into science and engineering courses (math. major and non math. major), and liberal arts.

Therefore, the multiformity of education has a tight relation to the influence of computers or the methodology of introducing computers. In practice, there are a variety of problems that have to be carefully considered, taking account of each educational environment. However, further discussion on this problem is not pursued in this paper.

4. The influence of computers on mathematical education

The possible influence that computers will exert on mathematical education will be so multifold itself that it is not easy to predict the influence perfectly now. Many experiments and theories must be accumulated before an answer can be found. As an initial approach, therefore, this section is devoted to pointing out the fundamental changes which computers will introduce in mathematical education and considering how to cope with them.

The changes which computers will bring in mathematical education can be divided into

i) changes in the methodology of math. education,
ii) changes in the topics taught in math. education.

These two categories are not independent but related to each other. For instance, if a topic is changed, the corresponding methodology must be changed. However, the two categories are different in whether they directly influence the topic.

Changes in the methodology of math. education

Many CAI (Computer Assisted Instruction) systems have already been tested in actual educational environments. In particular, a representative class of CAI systems, the drill and practice mode, is now extensively used. The results show that CAI systems are especially effective in improving students' ability to do formal calculations and in helping students to understand a new concept or topic by the use of graphical images.

This type of application of computers will be greatly increased in mathematical education. If computers are used as CAI systems more extensively in this way, the methodology of mathematical education will inevitably be changed. Namely, the conventional methodology whereby a teacher teaches everything by him/herself will be replaced by a new methodology whereby the teacher can selectively use computers for a particular topic or a situation in which computers are very effective. Therefore, it will be necessary to establish a new methodology of using computers most effectively in mathematical education in order to teach what has been taught without computers. That is to say, considerations must be made as to how computers will be introduced, for which topics, under in which sort of situation, and what effects can be obtained, etc.

It is noted that most of the CAI systems which have been developed so far are designed for the first objective of mathematical education which was given in section 3, i.e., acquisition of mathematical knowledge and computational skills. On the other hand, few CAI systems have been developed yet for the second objective, i.e., acquisition of the capacity for mathematical (logical) thought. The last fact arises because the methodology to develop the second objective is not explicitly established. Therefore, it is concluded that we can not let computers replace a greater part of what a human teacher has taught until the methodology for the second objective is explicitly established and a CAI system based on it is developed.

Changes in the topics taught in mathematical education

What to teach is determined by the demands of society. As computers are used more extensively and become more important, the demands of society change. Therefore, there may be an increase in: 1) the demand that computer oriented mathematics should play a greater part in mathematical education. Such computer oriented mathematics includes discrete mathematics, algorithms, etc. It is noted that discrete mathematics is defined here in a rather broad sense to include sets, graph theory, algebraic structure, boolean algebra, data structure, etc. The teaching of algorithms includes teaching what sort of idea is useful for algorithmic operation for use in computers.

Furthermore, as the capabilities of computers increase, what can be done by computers increases. Thus, a question arises: 2) can the part dealing with topics that computers can do, be reduced or omitted ? 1) and 2) are extremely important as they have a direct effect on the curriculum of mathematical education. Therefore, these two subjects are considered in detail below.

1) Should the teaching of computer oriented mathematics be increased ?

The mathematics used in computers is based on discrete and finite numbers. It is different from such mathematics as differentiation or integration which are generally taught in senior high school or the first year in college. The latter type of mathematics is based on continuity and infinite numbers. Therefore, discrete mathematics has been taught in higher education in computer related fields. In order for computers to advance further, a larger number of scientists and engineers who have mastered such computer oriented mathematics will be needed immediately. Therefore, it is obvious that the part on computer oriented mathematics will have to be increased in computer related fields.

On the other hand, as computers are used more extensively, opportunities for non-computer-specialists to use computers will greatly increase. Then, a question arises: should the teaching of computer oriented mathematics also be increased for those people ? In order to answer the question, it is necessary to consider how computers will develop in the future.

One of the most difficult problems that computer science has been facing is the low productivity of computer software. There is even a prediction that all the people on the earth will have to become computer programmers in the near future to meet the demands for software if computers increase in number at the present rate. Thus, it is firstly necessary to increase the number of software engineers to increase the productivity. With regard to this, there is the idea that computer oriented mathematics should be taught to people in non-computer-majors so as to make them programmers. This idea is not only effective but necessary to cope with the shortage of software engineers in the short term. However, is it still effective and necessary in the long term ?

One of the major reasons for the low productivity of computer software is believed to be the difference in the way of thinking of humans and computers. It is often said that computer are still in their infancy. Therefore, a lot of research has now been directed toward the development of a new type of computer which can be programmed much more easily. In addition, research to make software into parts, like the electrical parts used in assembling a radio, has been making progress.

In the long term, the successful results of such research is anticipated to greatly improve software productivity. At the same time, this implies that computers can be used or programmed by those who do not know computers very well. In essence, this sort of goal must be reached for the future of computers.

Therefore, it is concluded that, in the long term, a great part of computer oriented mathematics will not have to be included in the general mathematics curriculum as long as the above research is successfully continued to improve computers.

2) Can the part devoted to topics that computers can do, be reduced or omitted ?

As considered in section 2, the progress of computers and their capacity to do computer algebra in particular, is ashtonishing. Therefore, there arises the question: how can computer algebra systems be introduced into mathematical education ? For instance, most of the computer algebra systems can easily calculate differentiation symbolically. Then, can we let computers do the calculation and not teach differentiation at all ? Or, on the other hand, shall we not let students use computers when differentiation is taught ?

To answer these questions, it is necessary to consider the questions in the light of the basic objectives in mathematical education which were discussed in section 3. At first glance, it seems that computers can replace computational skills. For instance, why not let the computer calculate differentiation all the time if differentiation is necessary ? Why should students spend so much time practising tedious calculations ? Why not finish teaching differentiation altogether by simply teaching the definition ? The progress of computers is so fast that a small computer algebra system will soon be produced as seen in section 2.

However, the above idea contains a crucial problem. For instance, can a student understand differentiation well simply by learning the definition and how to use the calculating machine ? Can he/she really understand it without making efforts to solve many exercises by hand ? Obviously, the answer is negative. The reason is the same when students are not allowed to use electronic calculators when learning addition, subtraction, multiplication, division, etc.

Then, another question arises: isn't there any way in which computer algebra systems can be effective in mathematical education ? For example, when learning indefinite integration, is it essential to master complex expansion into partial fractions or tricky transformation of variables in order to learn indefinite integration ? As is obvious, such operations are not so essential to learn integration. It is noticed that when learning a topic there are two things which differ in nature: those which are quite essential for the topic, and those which are not essential but necessary as tools to understand the topic. The expansion and transformation introduced above belong to the latter. Therefore, in the case of the latter, there is a way in which computers can be effectively used, since these are not topics which have to be learnt now but things which have already been learnt. Consequently, it is now clear that when a student is learning, there are essential things which cannot be replaced by computers and inessential things which may be replaced by computers.

Therefore, when considering the introduction of computer algebra systems to mathematical education in the light of the educational environment as considered in section 3, it is primarily necessary to make the boundary of the above two things clear. Moreover it is necessary to examine the curriculum for possible changes, taking account of computer algebra systems. Finally, the development of a methodology for the effective use of computer algebra in mathematical education and an understanding of the effects of this methodology are necessary. These points will be increasingly required as computers progress. And it should be the responsibility of modern mathematicians and mathematical educators to consider these points for the future.

In the next section, a new way of teaching by using a computer mathematics system is presented for discussion.

5. A new way of teaching mathematics

--- Introduction of computer mathematics systems ----

This section presents a new type of mathematical educational model incorporating computer mathematics systems and discusses its advantages. It is noted that the discussion in this section is based on the assumption that cheap and small computer mathematics systems will be produced as shown in section 2. Namely, the discussion assumes a situation in the near future in which every student has a computer math. system on his/her desk and can use it with the ease of present electronic calculators.

5.1 Method of teaching

Let us now assume that the same topics as are taught at present are to be taught throughout mathematical education. In particular, let us consider a situation where a topic A is to be taught. Of course, it is assumed that the computer systems on students' desks can calculate what is going to be taught.

The teacher teaches the basic concepts and then the applications of the topic A in almost the same way as before. Hereafter, let us call the former part, 'the basic course' and the latter, 'the application course'.

During the basic course, students are not generally allowed to use the computer system. This is due to the fact that the use of computers for what is being taught will hinder the students from a deeper understanding of the topic A. Namely, students would use the computer system instead of thinking or elaborating a solution by themselves. However, CAI systems, e.g., CAI systems with graphic images, that will help students to understand the basic concepts of the topic should be frequently used under the supervision of the teacher. Compared with the conventional method, more emphasis is placed on teaching the principles or meaning of the topic A. Thus, a longer time must be assigned to this course and students are given more basic exercises. However, more complicated exercises in which the complexity is not so essential to the topic A should be reduced. As a result, a greater part of the topic A is spent on the more basic concepts or principles compared with the conventional one.

Let us now suppose students have to use a knowledge B which has already been studied but is not essential to the topic A, in order to solve a question during the basic course of the topic A. That is, B is an inessential part of the question of A. For instance, such a knowledge B includes a complex expansion of partial fractions when learning indefinite integration, etc. In this case, the new teaching method allows the students to use the computer to solve B even during the basic course of A, if and only if it is certain that B is inessential to A. Namely, students may use the computer to get an answer for a part of the question and apply it to the question of A. By allowing the use of the computer algebra systems for an inessential part of a question even during the basic course, it is possible to draw students' attention to the more essential parts of the new topic being taught. Consequently, a deeper understanding of the topic can be obtained effectively.

Let us next consider the case of the application course of the topic A. After the basic course, the students have understood the basics of the topic A fairly well. The purpose of the application course is to clarify the position of the topic A by applying it to more complex questions and learning the relation of A to other topics, thereby reaching a deeper understanding of A. During this course, students are allowed to use computer systems not only for inessential parts such as B but for the essential part of A to some extent. Before the computer can be used for A, of course, instruction concerning A's facility on the computer must be given to the students. Such instruction includes how to get answers to A and an explanation about the limits, and advantages and disadvantages of the computer algebra systems with respect to A.

A model of problem solving by the new teaching method incorporating computer algebra systems is shown in Fig. 5.1. The model is different from the conventional ones which do not use computers in that computers are used for numerical calculations and algebraic operations (4), and graphic imaging and simulation of the obtained results (5).

This model features the following advantages. That is to say, by allowing computers to do the work for (4) and (5), students can solve problems more quickly and with less effort. If the input to the system is correct, the results from the system are generally free from the mistakes which students might make during tedious calculations by hand. Therefore,

(i) A student can more quickly verify his understanding of the problem, his basic strategy, and his mathematical formulations. Thus, if there is any mistake, he can correct it more quickly and more easily, i.e., he can attempt the problem again much more easily. Due to these advantages, students can focus their attention on more intellectual work, i.e., problem understanding, planning basic strategy, verifying the obtained results, etc. In addition, since students can verify their ideas much more quickly by examining the results, they are encouraged to study further. This leads to an increase in students' motivations to study.

(ii) Students can try several strategies for comparison. This sort of learning leads to a development in students' proficiency in obtaining an optimum stragegy. If the calculations in (4) of Fig. 5.1 are complicated, the learning takes too much time and effort without computers, so it is impractical.

(iii) Students can solve more problems in a limited time. Thus, they can see the topic being taught from a larger number of viewpoints. This leads to a deeper understanding of the topic.

(iv) This new method does not hinder students from developing the second objective of mathematical education, i.e., acquisition of mathematical thought. Namely, what is replaced by computer algebra systems has, in essence, little relation to the development of the objective.

There is a point which must be noted when applying this new teaching method. Namely, since the computer systems return wrong answers to incorrect inputs, it is extremely important to instruct students not to believe the answers from computers too much. Therefore, greater efforts must be made to develop proficiency, in order to be able to verify the validity of the obtained results and select the right answer from a number of outputs from the computer.

In summary, the advantage of the new teaching method using computer algebra systems is that the method can shift the focal point of mathematical education to more essential point, such as more emphasis on problem understanding, elaborating basic strategies and mathematical formulations and verification of obtained results. Accordingly, a greater amount of more essential materials must be included in the mathematical curriculum.

In order to make full use of all the advantages of this new method, there are several points which the computer algebra systems to be used must feature.

(i) The final output from the computer system is not always the most suitable answer for the students' use. Therefore, the systems must allow students to see the important intermediate results.

(ii) The above feature leads to a computer algebra system with the following two operating modes.

(1) Calculator mode: returns only final results. Students use the system just as a computational tool.

(2) CAI mode: provides not only final results but intermediate results, explanations of the rules used to reach the final results, etc. This mode is used when students want to understand the function of the system which is usually treated as a black box, or when they want to use the intermediate results.

(iii) A graphics system which allows results to be displayed from the algebra system or to be simulated must be effectively connected with a computer algebra system. Thus, students can use not only the algebra system but also the graphics system, so that they can tackle problems more easily.

(iv) The system must allow students a numerical calculation system as well as the algebra system. The two systems must be integrated effectively, so that students can perform numerical analysis of an expression obtained by the algebra system.

(v) The system must allow hand-writing input in addition to the ordinary key-input, in order to make the system easier to use.

5.2 Feasibility of exploratory mathematics

Finally, the feasibility of exploratory mathematics is examined. Exploratory mathematics is a heuristic educational method which allows students to experimentally or inductively discover rules or theorems by themselves using computer mathematics systems. In this way, rules or formulas are not taught top-down, but bottom-up. Namely, students use the computer algebra system to find rules and make hypothses. Then, the students attempt to prove their hypotheses. A model for exploratory mathematics is shown in Fig. 5.2.

As an example, let us consider a case where the binomial theorem is to be taught. Before showing the expansion formula of (1+X) , students use the computer algebra system and expand the expression for n=0,1,2,3,..... Observing the obtained Pascal's triangle, the students can find the rule.

In addition, it may be expected that they find associated formulas at the same time, e.g.,

$${}_{n}C_{r} = {}_{n-1}C_{r} + {}_{n-1}C_{r-1}.$$

This educational method allows students to find such rules and verify them.

Although almost the same method could be performed without using computer algebra systems, it would take too much time and effort and thus, would be impractical. However, the use of computer algebra systems will make it practical and effective.

This method can give students pleasure when they discover something, so that it promotes their motivation to study. Furthermore, the heuristic ability is extremely important not only for mathematics but also for any sort of scientific research. It should be stressed here that such an ability can be developed by exploratory mathematics. In addtion, this method can be used in the educational method stated in section 5.1. That is, during the basic course, this heuristic method can be included to teach specific topics.

Finally, there are topics in which exploratory mathematics is especially effective. Therefore, it is necessary to select such topics suitable for this method.

6. Conclusions

This paper considered the influence of the progress of computers on mathematical education. What was shown is as follows:

1) Small and inexpensive computer systems capable of algebraic operations will soon be produced. Thus, mathematical education will be greatly influenced by the emergence of such systems.

2) Inevitably, there will be two types of changes brought about by computers; changes in the methodology and changes in the topics taught.

In the case of the former, the extensive use of CAI systems will inevitably be influential.

As for the latter, this paper considered the necessity of computer oriented mathematics and the educational strategies for the material which can be handled by computers.

This paper showed:
i) Computer oriented mathematics must be increased in the short term.
ii) An increase of computer oriented mathematics in the general mathematics curriculum will not be necessary in the long term due to the advance of computer technology.

For the educational strategies, an educational model which extensively uses computer algebra systems was presented, and the advantages were considered.

3) The advantages of the new educational model are:

i) Students do not have to spend as much time on tedious calculations.
ii) Instead, they can concentrate on more essential and intellectual matters, i.e., problem understanding, elaborating basic strategies and mathematical formulations, and verifying the obtained results.

4) Exploratory mathematics can be utilized in the actual educational environment by the use of computer algebra systems. This method is especially effective in developing the ability of heuristics which is very important for all scientific work.

5) A revision of the curriculum will be necessary to incorporate computers into mathematical education.

Like it or not, the extensive use of computers is bringing about a variety of changes into our society and our daily lives. It is remarkable to note that computers can be an effective tool in attaining the ultimate goal of education, if the goal can be stated as to make a man who thinks by himself, who studies by himself, and who, having set himself questions, can think of the way to solve them. In this sense, the establishment of a most effective way of using computers in mathematical education is urgently needed.

References

[1] R. Loos, "Introduction," B. Buchberger et al. Ed., Computer Algebra: Symbolic and Algebraic Computation, Springer Verlag, pp1-10, 1982.

[2] J. Huizen and J. Calmet, "Computer Algebra Systems," B. Buchberger et al. Ed., Computer Algebra: Symbolic and Algebraic Computation, Springer Verlag, pp221-243, 1982.

[3] Y. Kaneda, "What is Computer Algebra ?" Mathematical Science, No. 8, pp5-12, 1983 (in Japanese).

[4] A.Toshima, "RLISP," My Computer, No. 15, pp100-111, 1984 (in Japanese).

[5] T. Sasaki, "Development of Electronics," Encyclopedia of Electronics, pp6-28, Nikkan Kogyo News Inc., 1983 (in Japanese).

[6] M. Ikeda and T. Yamamoto, "Computer Algebra and LISP," My Computer, No.15, pp52-69, 1984 (in Japanese).

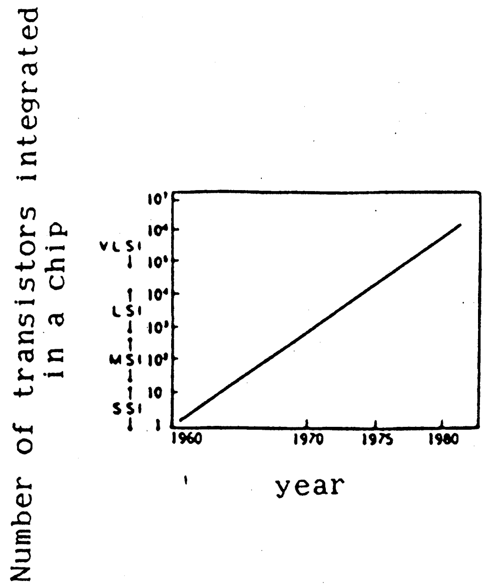


Fig. 2.1 Progress of the number of transistors integrated [5].

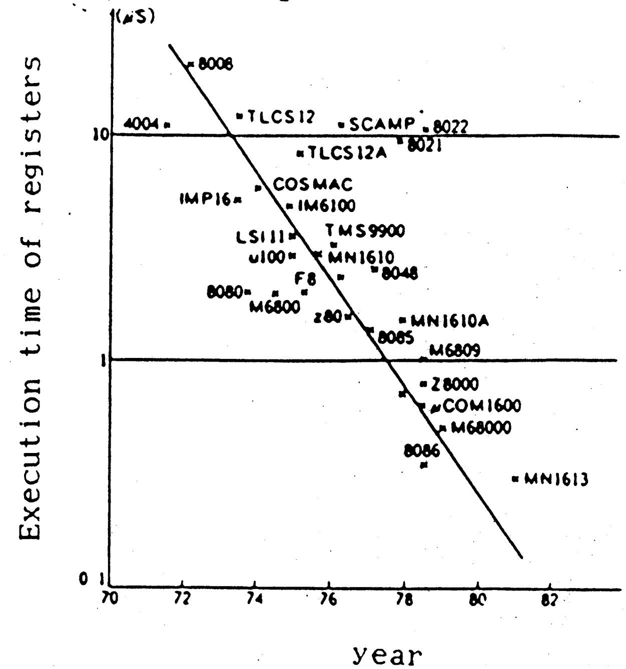


Fig. 2.3 Speed improvement of microprocessors [5].

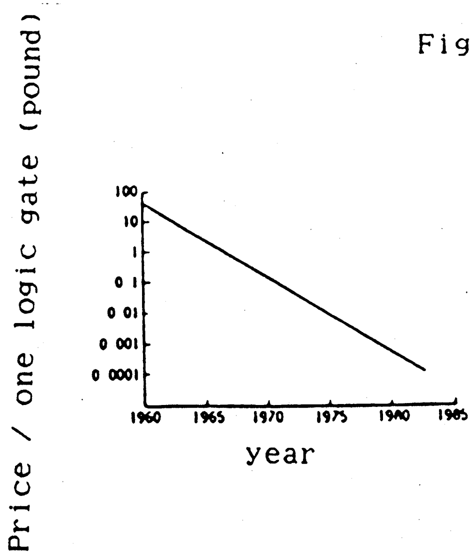


Fig. 2.2 Reduction of cost of LSIs [5].

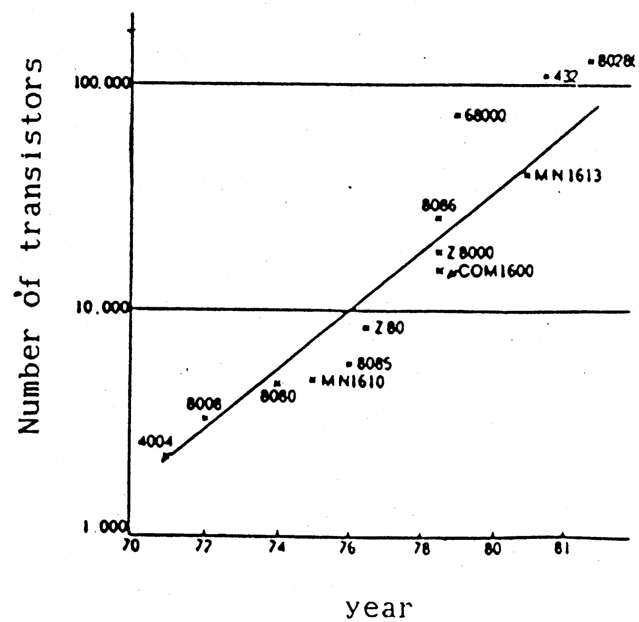


Fig. 2.4 Improvement of integration for microprocessors [5].

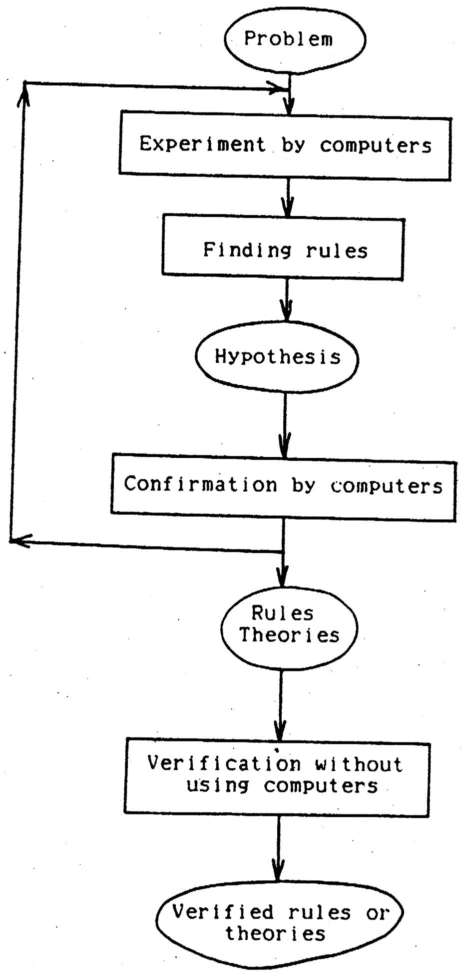


Fig. 5.2 A model of exploratory Mathematics.

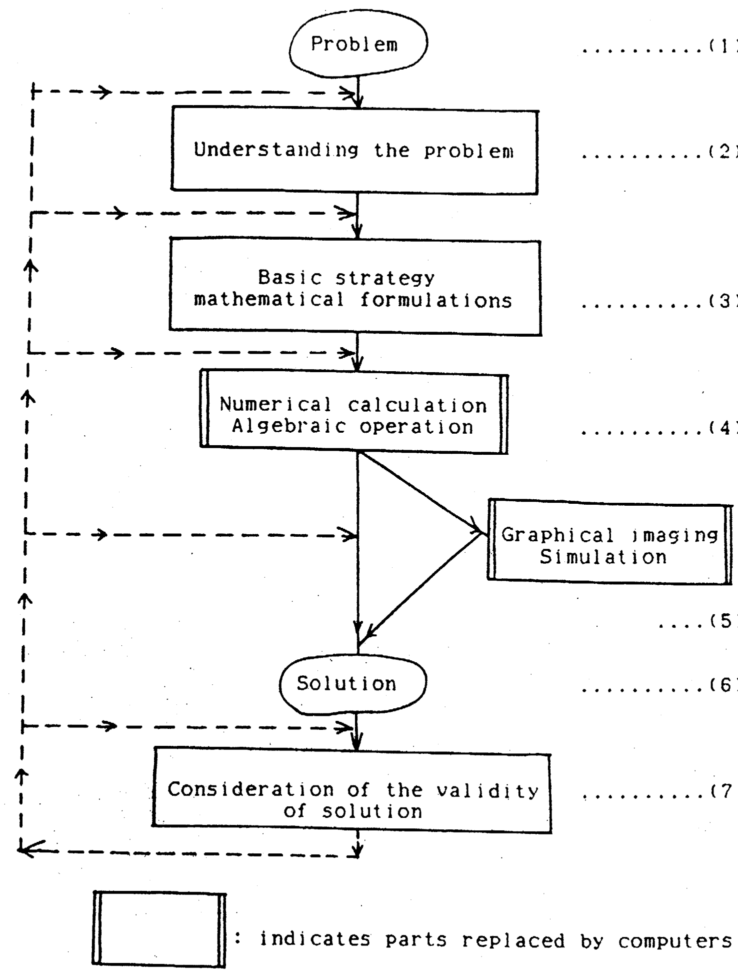


Fig. 5.1 A model of problem solving by the new teaching method incorporating computer algebra systems.
(As indicated by dashed lines, it is possible to return from one point to any other.)

The University of Bath Syllabuses

J. H. Davenport,
Computing Group,
School of Mathematics,
University of Bath,
Bath BA2 7AY, England

1 – Introduction

The purpose of this document is to explain how one particular university, the University of Bath, approaches the problems raised by the International Commission on Mathematical Instruction (*L'Enseignement Mathematique* **30** (1984) pp. 159–172) on the influence of computers on mathematical education. These approaches are certainly not perfect, and they are unlikely to be transplantable to other universities or countries in anything like their current form. Nevertheless, they do form an approach which has been agreed between the varying groups within the School of Mathematics, and which seems to find favour with students and employers. This report focusses on the training of computer scientists, and the training of mathematicians in computer science: the School trains pure and applied mathematicians and statisticians as well, but only the computing elements are described here.

The author has taken advice from many of his colleagues, both in general and on this particular document. Nevertheless, it is a personal view, and not an "official" policy. Any errors of fact and emphasis are the author's responsibility. The author concentrates on his own teaching (the logic component of A5, C14 and E8), not only because it is the area which he understands best, but also because, as the most "mathematical" member of the computing group, he is closest to the problems of relating mathematics and computer science. The author has omitted various special cases, transitional provisions, and regulations allowing people to take courses in years other than the usual, in the interests of simplicity. Courses marked A and B are taken in the first and second semesters (i.e. the first year) respectively, C and D in the third and fourth semesters (second year) and E and F in the fifth and sixth semesters (third year, or fourth for the sandwich students).

The School of Mathematics admits about 80 undergraduates per year, of a relatively high standard (14.1 'A' level points out of a maximum of 15, for those familiar with the British 'A' level system). These take either a three-year course, or a four-year "thick sandwich" course with the third year being spent in industry (a phrase that is interpreted widely, to include research establishments and the civil service, as well as part of the year being spent on teacher training for those intending to go into school teaching). Depending on the options studied during the last two years, students may either take the general degree *Mathematical Studies*, or be qualified to take one of the more specialised degrees in *Mathematics*, *Mathematics and Computing* or *Statistics*. Distribution between the degrees varies from year to year, but no one degree is predominant. The University does not have a purely computing course, and is unlikely to establish one in the immediate future (see the conclusions).

2 – The First Year

The first year is common to all students, and consists of ten compulsory courses, together with optional courses in Education, Economics and Politics. The compulsory courses are of 13 weeks duration (26 lectures + problem classes and tutorials) except where stated, and, apart from A5, form a fairly conventional mathematics course.

Note that the students will have done one or two mathematics 'A' levels, and will therefore have been exposed to a substantial amount of mathematics at school – typically between one-third and two-thirds of the total school time spent on academic subjects in the last two years. This means that they will have done some linear algebra, a significant amount of calculus, probably including some simple differential equations, and may have been exposed to concepts like Group Theory, Probability etc. No prior computing knowledge is assumed.

<u>A1 & B1 Algebra.</u> Sets. Functions. Relations including equivalence and order. Binary Operations. Algebraic properties of *R*, *C*, *Q* and *Z*. Order properties including induction and completeness; countability. Axioms of a group. Examples including symmetries and permutations. Subgroups. Cyclic Groups. Concept of isomorphism. Lagrange's Theorem. Axioms and examples of a field, including some finite fields. Axioms and examples of a vector space. Subspace. Linear dependence and independence, bases, dimension. Linear transformations. Dimension Theorem. Matrix representation of a linear transformation and links with matrix algebra. Rank of a linear transformation.

Systems of linear equations. (22 weeks combined).

A2 & B2 Analysis. Review of properties of R including induction and the Completeness Axiom. Convergence and divergence of real sequences. Limit Theorems. Monotonic sequences. Subsequences. Cauchy sequences. Limit supremum/infimum. Bolzano-Weierstrass Theorem. Convergence and divergence of real series. Comparison, ratio, root, integral and alternating tests. Conditional and absolute convergence. Rearrangement of series. Introduction to power series and the elementary functions. Definition of pointwise continuity, elementary algebra of continuous functions. Properties of functions continuous on closed intervals. General theory of limits of functions. Elementary properties of differentiable functions. Mean Value Theorem and l'Hopital's rules. Taylor's Theorem. Inverse functions: existence, continuity and differentiability. Sequences of functions: uniform convergence; differentiability of power series. Taylor series. (26 weeks combined).

A3 & B3 Mathematical Methods. "The aim of this course is to develop methods of solution for problems arising in a wide range of applications of mathematics". The syllabus is lengthy, but of importance for the computing element of the course are items on:

a) Difference equations. Formation of difference equations in approximate solution of differential equations. Methods of solution of simple difference equations, using analogous methods to those used for differential equations.
b) Little o and big O notation.
(26 weeks combined).

A4 Vectors and Matrices. "This course presents a 'methods' approach to vector spaces over the reals with particular attention to 2- and 3- dimensional vectors". (13 weeks).

B4 Applied Mathematics. This course includes Newtonian mechanics, emphasized as a very successful and widely applicable branch of applied mathematics. (13 weeks).

A5 Computer Programming. See the next section. (17 weeks).

B5 Probability and Statistics. "The aim of this course is to introduce the basic concepts of probability and statistics". (13 weeks).

3 - A5: the first year computing.

This course, which is the only computing course that all students have to take, assumes no prior knowledge of computing. It is largely a relatively conventional introduction to the local computer system (PDP/11 Unix, about to become GEC 63/40 Unix) and a programming language (FORTRAN 77). This latter point requires some comment - my colleagues in Computer Science departments often ask why we do not teach Pascal (or MODULA 2 or ...). The answer is, of course, that we are not a separate Computer Science department, and that what we teach has to be aimed at the needs of all undergraduates, not just those specialising in computing. In particular, it has to help those who take up industrial placements in the sandwich year. Here it is worth noting that all sandwich students use a computer in their placement, and about two-thirds use FORTRAN (as a main part of the programming, if not exclusively). Indeed, since the typical non-computing graduate will work as a statistician or applied mathematician, FORTRAN is the language they are most likely to meet after graduation. The assessment on this course is 60% by examination and 40% by programs written during the course.

One slightly unusual feature is that this course includes four lectures (taught by the author) on "propositional logic and its application to programming", and this is the only part of the first-year syllabus in which logic is formally met. Clearly, it is not possible to do a great deal in four lectures, but it does seem to be useful. In general, the emphasis is on formal methods, but the proofs of these methods (where given) are informal. The application is to computers and computer programming in general. The application to computers explains the use of *monotone*, which has the logic-designer's meaning of an expression without negation, rather than the logician's meaning. The detailed scheme of lectures is:

1) Monotone Logical Expressions. Introduction of & (and) and | (or). Syntax of expressions (very brief since students are already familiar with the syntax of FORTRAN). Associativity and commutativity of these rules. Both distributive laws - explain the contrast with "ordinary algebra", i.e. rings. Define Conjunctive and Disjunctive Normal Forms. Informal proof, via truth

tables, that these are canonical forms for monotone expressions.

2) Non-monotone Logical Expressions. Introduction of ¯ (not). Remark that ¯ is self-inverse, and explain the distribution over & and |. This point seems particularly important in English: many students, and even experienced programmers, translate the informal statement "if x is not equal to 2 or 3" as $(x\neq2)|(x\neq3)$, whereas the intended meaning is $(x\neq2)\&(x\neq3)$.
Definition of => and <=> in terms of the primitives, and a brief discussion of the difference between => and naive "implication". Explanation of contrapositives and proof by *reductio ad absurdum*.

3) Non-Monotone Logical Expressions (II). Definition of Conjunctive and Disjunctive Normal Forms for non-monotone expressions. Examples showing that they are not canonical in general. State, and justify via truth tables, that an expression is a tautology if, and only if, its CNF is void, and that it is a contradiction if, and only if, its DNF is void. Theorem that two expressions a and b are equivalent if, and only if, a<=>b is a tautology. Application to a binary adder - verification that two designs are equivalent.

4) Predicate Calculus. Syntax of the predicate calculus - propositions with variables. Quantifiers *A* and *E* (for purely typographic reasons!). Distinction between free and bound variables - relationship to local and non-local variables of programming. Explanations of what distributions are, and are not, axioms of the Predicate Calculus - justification with "real life" examples. Point out that *A* and *E* cannot be interchanged - reference to uniform continuity.

The examples for this group of lectures consist of a mixture of formal logic, programming, and "real world" (e.g. bureaucratic rules). There is no book recommended as such, although students are advised to re-read *Alice Through the Looking Glass*.

4 - The Second Year

The second year sees some specialisation in the students, usually in the form of deciding a topic to drop. There are four compulsory courses, which extend the mathematical education, and a variety of courses which are necessary for proceeding towards one of the various specialised degrees, either by themselves or in the form "two out of the following ...". The four compulsory courses (each of 13 weeks) are

C1 - Linear Algebra. Linear transformations and matrices. Eigenvalues. Bilinear and quadratic forms. Inner Products.

C2 - Real Analysis. Uniformity. The Riemann Integral. Functions of Several Variables.

D1 - Complex Analysis. Regular functions **C** - **C**. Cauchy-Riemann equations. Cauchy's Theorem for a disc. Cauchy's integral formulae. The theorems of Taylor, Laurent and Liouville. Zeros and singularities - behaviour at infinity. The residue theorem and contour integration.

D2 - Mathematical Methods. Formal treatment of Fourier series. The Fourier transform. Laplace transform. Sturm-Liouville theory. Laplace's equation. Legendre polynomials. The wave equation in one space dimension.

In order to specialise in computing, it is necessary to take courses C14 (Algorithms and Data Structures - see the next section) and D14 (Introduction to Computer Science), and two out of the following four: C5, C8, D8, D11. C5 and D11 are Numerical Analysis courses, relying on the programming taught in A5, and on the mathematics (e.g. of eigenvalues) included in the compulsory mathematics courses. It is worth pointing out that these courses are taught by members of the Mathematics Group, and are taken by those who do not intend to be computer specialists as well as by those who do. D8 is a course on micro-processors and computer architecture, and tends to be limited to computer specialists. C8 is on Computability Theory - "While being of theoretical interest to students of computing, this course is basically pure mathematics" - and covers areas such as finite state machines, Turing machines, recursive functions, Church's thesis and undecidability. Most students taking it also take C14, and an attempt is made to link the two (bridging the gap between the Mathematics Group, teaching C8, and the Computing Group, teaching C14).

There is also a course on Group Theory (recommended to many) and several courses in statistics are often taken by people intending to specialise in computing. Since a student normally does 10 or possibly 11 courses in the second year, almost all programmes include at least one "non-computing" course, even for the most computing-oriented student, and similarly nearly all students take at least one computing course, whatever their final orientation.

5 - The Second Year Computing Core

The core computing in the second year consists of C14 and D14, though those interested in non-numerical computing will take C8 and D8 as well (see above). C14 is taken by about two-thirds of the students, of whom rather less than half will ultimately become computing specialists (in terms of their university studies - who knows what will happen later). D14 is taken by about half the students. It has C14 as a prerequisite, and is largely concerned with applying the methods of C14 to problems such as operating systems and computing languages, as well as talking about the organisation of virtual memory, interrupts and similar details.

C14 attempts to take a fairly mathematical approach to the design and analysis of computer algorithms and data structures. However, it is not a purely theoretical course: the assessment is 60% by examination and 40% by programs written during the course. We give below the detailed syllabus, indicating in square brackets the elements of mathematics that are used during the course. The author hopes that this will provide one answer to the question "What do computer scientists really want?". Of course, not all the mathematics has been developed before: items marked with * are taught during the course, rather than quoted from previous courses. There is no one standard book: students are referred to various parts of Knuth *The Art of Computer Programming* or Aho, Hopcroft & Ullman *The Design and Analysis of Computer Algorithms*.

The course is taught using C as a programming language. While the students learnt FORTRAN in the first year, it is really not adequate as a vehicle to express more complicated data structures, and such ideas as recursive algorithms. The question of what language to use instead is not clear. The author sees little point in teaching Pascal as a *second* programming language, especially since one of the aims of the course is to practise students in modularisation and dividing programs into separate units, which is not supported by standard Pascal. MODULA 2 might be attractive, but it is not available on the computing system, and anyway might stress a PDP 11/44 too much (the transfer to the GEC may change this). Such considerations also rule out ALGOL-68. In the end, C was chosen more because of its availability and the tools that go with it than for any positive intellectual reason. With the current rise in popularity of Unix, it also increases the students' employability. This reflects the general fact that computing teaching tends to be more constrained by such problems than mathematics teaching.

<u>The language C</u>

int and *long*: number representations.

float and *double*: number representations.

logical operations. [Boolean algebra: Propositional Calculus (from A5)].

scope and declarations: program structure.

arrays and records [the concept that $x \neq \{x\}$].

pointers.

recursion [induction].

informal justification of recursive factorial functions.

<u>Data Types</u>

Algebra of stacks [Axioms for an algebra: A1/B1].

Stacks in arrays.

Algebra of queues.

Linked Data Structures.

Trees, tree traversal and treesort [induction].

Hashing [modular arithmetic. A vague feel for the probability issues involved]

<u>Algorithms</u>

Choice of algorithms [little o and big O]

Introduction to divide-and-conquer: Classical $O(n^2)$ multiplication and Karatsuba $O(n^{\log_2 3})$ multiplication [algebra: manipulation of logarithms to different bases].

Sorting: $n-1$ interchanges suffice [permutation groups: cycles].

Need $O(n\log n)$ comparisons [Stirling's formula*].

Bubblesort and variants [$\Sigma i = O(n^2)$].

Quicksort - $O(n\log n)$ on average [f concave implies

$$\sum_{1}^{n} f(i) \leq \int_{1/2}^{n+1/2} f(x)\,dx \quad *].$$

Median in $O(n)$ [recurrence relations, recurrence inequalities, e.g. $T(n) \leq n+T(n/7)+T(5n/7)$ *].

Guaranteed $O(n\log n)$ quicksort.

Tape sorting - properties of hardware.

three-tape polyphase sort [Fibonacci numbers - $F_n = O(\phi^n)$].

Internal sorting phase - snowplough algorithm.

Graphs

Data structures for graphs and directed graphs [graphs; directed graphs*].

Eulerian circuits [Euler's Theorem*].

Shortest paths - inkblot algorithm.

Brief mention of NP-hard and NP-complete.

6 - The Final Year

In the final year, most students take eight courses, of which at least five must be in their speciality (Mathematics, Computing or Statistics) if they wish a specialised degree. Note that Numerical Analysis and Number Theory (see later) count both as Mathematics and as Computing. The computing has core courses in systems programming and operating systems, and options in graphics (relying on the linear algebra learnt earlier), programming languages and formal programming techniques (relying on the logic etc.), LISP and the λ-calculus, Number Theory and Numerical Analysis. We give in the next section a detailed description of the Number Theory course.

In general, there are well-recognised groups of courses that students take, but it is not unusual for computing students to take a statistics course, or for a non-computing student to take, say, an operating systems or graphics course. The School very much approves of this flexibility, and it is a pity that in many institutions a rigid barrier is raised between those primarily interested in computing and those whose primary interest lies elsewhere. Of course, it does increase the teaching and equipment load.

7 - E8: Number Theory and Applications

The purpose of this course is to give final year students some understanding of number theory, with special emphasis on computational applications, e.g. in cryptography. This course is somewhat hampered by the fact that it precedes the "rings and fields" course, and that the students have, in fact, a fairly weak knowledge of algebra. At the moment, the second year group theory course is not a prerequisite, but that will change.

1) Numbers, primes, every number is the product of primes [HW T. 1]. There are infinitely many primes [HW T. 4]. Uniqueness of prime decomposition in $\boldsymbol{N}$ [HW T. 2], non-uniqueness in $4\boldsymbol{N}+1$ and $Z(\sqrt{-5})$.
2) Z/pZ is a field; congruences. More general congruences, Chinese Remainder Theorem [HW T. 121], application to arithmetic.
3) Fermat's Theorem [HW T. 70]. Wilson's Theorem [HW T. 80]. Lagrange's Theorem: a polynomial of degree n has at most n roots in a field [HW T. 107]. Existence of a primitive element [HW T. 108]. Logarithms base p. Applications to Diffie-Hellman cryptography.
4) Fermat's Theorem as a primality test? Carmichael numbers. Rabin's probabilistic test [K. p. 379]. General remarks on probabilistic algorithms.
5) Pollard's Monte Carlo Algorithm. [K. p. 370] General discussion of factoring - state Schnorr's bound. Use for RSA public-key encryption systems [K. pp. 386-389].
6) Linear Congruential random number generators: primitive elements and maximal periods [K. pp. 16-20]. Prime moduli and 2^n.
7) Continued Fractions, induction rule [HW T. 149], convergents. Representation of rationals, connection with Euclid's algorithm. Reconstruction of rational numbers from reals [HW T. 164], from modular representations [HW T. 184]. Rounding rational numbers. Continued Fractions for irrationals - periodic iff quadratic [HW T. 176, 177]. Remarks on approximation to rationals by irrationals; the example of π - 22/7 and 355/113. Approximation to astronomical constants: the calendar.
8) Definition of transcendental numbers. Liouville's Theorem [HW T. 191]. Transcendence of e [HW T. 204]

In the above, we use HW to stand for Hardy & Wright, *An Introduction to the Theory of Numbers*, and K to stand for Knuth, *The Art of Computer Programming* Vol II, 2nd. ed. The recommended students' book is H. Davenport, *The Higher Arithmetic*.

8 - Conclusions

We have tried to present one university's actual curriculum, which straddles the areas of mathematics and computer science. This curriculum, having evolved for several years in response to changing staff and student interests, and being inevitably constrained by resources, timetabling, and the requirement that sandwich students be attractive to industry after two years, is not perfect. Perhaps, though, that makes it a more useful model than an "ideal" curriculum set up by some committee.

As in any evolving system, some gaps and idiosyncracies are evident, e.g. linear algebra seems to be covered from several different points of view, but this is not necessarily bad. While a fair amount of the mathematics underpinning C14 has been covered in the previous mathematics courses, a substantial amount has not. For example, it is odd that Stirling's formula is only encountered by those who do a computing option. The general treatment of abstract algebra is somewhat weak, but it is hoped that future developments based around the Number Theory course will alleviate this.

It would be possible (given additional staff - a non-trivial constraint in terms of university funding and industrial pay scales) to increase the computing component of the course. Based on the experience of other universities, this would probably lead to a sounder degree than would the opening of a purely computing degree, and would at the very least provide a suitable half-way house to such a move. But does the world really need a large supply of computer scientists, especially those not trained in mathematics? Our experience has been that employers are delighted with the *Mathematics and Computing* degree we provide, and some computer manufacturers have said that they prefer it to purely computing degrees from other places (though this is, of course, a biased point of view, and people tend to flattery).

It is appropriate to end with a comment from a colleague: "The commonest comment made to me is that the employer is:

a) astounded at the students' ignorance of computing;

b) astounded at how quick the students are to learn".

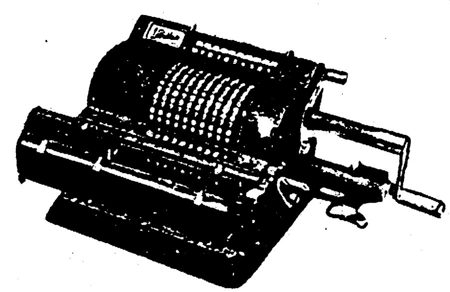
This may not be a bad position in which to be.

THE M.E.I. SCHOOLS PROJECT: AN INTEGRATED APPROACH TO THE TEACHING OF MATHEMATICS AT SENIOR SECONDARY LEVEL.

by Douglas Butler, Oundle School, England.

The Mathematics in Education and Industry Project was founded in 1965 by a number of schools in England who wished to present secondary mathematics that was more relevant to the needs of further education and industry. Through contact with the 'Schools and Industry Committee' of the Mathematical Association, a numerical approach to estimation, calculus and statistical analysis has been encouraged, and an enlightened attitude to the use of calculators and computers in school mathematics has been feature of these syllabuses from the early days.

Electronic calculators are now in common use, and their impact on the teaching and assessment of school mathematics is almost complete. On the other hand, the microcomputer is not yet established: there are still relatively few mathematics teachers making regular use of micros in the classroom, and this paper attempts to outline some of the benefits of an integrated approach using the new technology as a teaching aid at senior secondary level.



Hand calculating machines such as the one illustrated were used by candidates in M.E.I. Applied Mathematics A level examinations from the start. This enabled a numerical approach to integration and differential equations to be included before the introduction of electronic calculators and micros.

Teachers will not generally take to using new aids unless they have them to hand, and the conclusions of this Symposium and the suggestions in this paper, and others, cannot be taken seriously until it is the short term aim of all schools to install one micro in each mathematics classroom. To this end, we are all at the mercy of Governments to provide the funding and manufacturers to provide inexpensive hardware, perhaps of a portable nature. Moving occassional mathematics classes to a laboratory of computers is not the answer: the mathematics must dominate, not the machine, though it is highly desirable that students should have access to computers themselves so that they can learn to use them as a resource in their private study.

By way of illustration, this paper selects 8 topics from the M.E.I. A level Syllabus in Mathematics (1), where the micro has already been shown to have influenced and enhanced the learning process. With each topic is a selection of questions from recent examination papers (2), together with screens from the appropriate teaching software that has been written for the M.E.I. Project (3).

---

1. GRAPHS and FUNCTIONS: the micro can serve as a visual extension to the calculator, enabling its incidental use to illustrate standard presentations. For example, polar graph plotting is now so simple that students can be introduced to a wide range of curves (even the whole family of conics) in a short space of time. Knowledge so easily learnt, however, is easily forgotten, and it will always remain the teacher's responsibility to ensure that basic principles are securely taught.

2. TRANSFORMATIONS and MATRICES: the students' mathematical experience can be greatly enhanced through the carefully selected use of computer demonstrations. As with all of these topics, it is desirable that an accompanying programme of directed private study is offered to the students outside class time on school or home computers.

3. NUMERICAL METHODS: although students have been able to become moderately familiar with iterative processes through their calculators, the micro offers the chance to explore many more examples, with a visual simulation (and a 'zoom' feature) offering a new insight into the meaning of convergence and divergence.

4. - 6. PROBABILITY and STATISTICS: the teaching of most of this subject has been transformed - simulations can now bring the theory of probability to life, and the central limit theorem in particular can now be demonstrated convincingly using a wide variety of population distributions and sample sizes. Despite micros being unavailable in examinations, students can approach examination questions with a firmer grasp of the essential theory gained from interative study at the computer.

M.E.I. ADVANCED LEVEL MATHEMATICS AND FURTHER MATHEMATICS (from June 1984)

<table>
<tr><th>Paper</th><th>Instructions
(All papers 3 hours)</th><th>Pure Syllabus</th><th colspan="3">Applied Syllabus</th></tr>
<tr><td rowspan="3">MATHS A</td><td rowspan="3">Paper 1 (Pure)
Section A: 8 short compulsory questions
Section B: 5 longer questions (choose 3)

Paper 2 (Applied)
Section A: 3 short compulsory questions
Section B: 7 longer questions
Section C: 7 longer questions (choose 5)</td><td rowspan="3">Paper 1
Algebra
Geometry and Trigonometry
Calculus
Numerical Methods</td><td colspan="3">Paper 2</td></tr>
<tr><td>Section A
Probability

Differential Equations</td><td>Section B
Further Probability

Statistics</td><td>Section C
Applied Calculus

Mechanics</td></tr>
<tr><td></td><td colspan="2">(Alternatives)</td></tr>
<tr><td>MATHS S</td><td>Section A: 8 harder questions on A Level
Section B: 2 questions on each of the 5 topics (choose 6)
Project: candidates may submit an optional Mathematics Project.</td><td>Topics
1. Calculus
2. Numerical Methods</td><td colspan="2">Topics
3. Probability & Statistics</td><td>4. Mechanics
5. Engineering Statics*
(*experimental Topic)</td></tr>
<tr><td rowspan="2">FURTHER MATHS A</td><td rowspan="2">Paper 1 (Pure)
10 questions (choose 7)

Paper 2 (Applied)
Section A: 9 questions
Section B: 9 questions (choose 7)</td><td rowspan="2">Paper 1
Algebra
Geometry and Trigonometry
Calculus
Numerical Methods</td><td colspan="3">Paper 2</td></tr>
<tr><td colspan="2">Section A
Probability

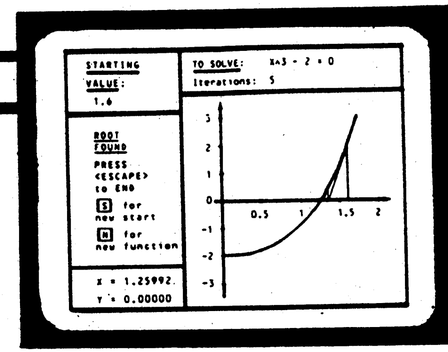
Statistics</td><td>Section B
Applied Calculus

Mechanics
(Alternatives)</td></tr>
<tr><td>FURTHER MATHS S</td><td>Section A: 8 harder questions on F.M. A
Section B: 3 questions on each of the 4 topics (choose 6)
Project: candidates may submit an optional Mathematics Project.</td><td>Topics
1. Algebra
2. Calculus and Numerical Methods</td><td colspan="2">Topics
3. Probability & Statistics</td><td>4. Mechanics</td></tr>
</table>

The overall scheme for the M.E.I. A levels in 'Mathematics' [taken by over 2000 candidates in England and Wales] and 'Further Mathematics' [taken by over 500]. The more able candidates, who take the 'Special' papers ("S") can offer a mathematics project, and this has over the years inspired original work of a high standard, often involving computer simulation.

A PAGE FROM THE DOCUMENTATION FOR THE M.E.I. TEACHING PROGRAMS

L Newton-Raphson Method-for any f(X)=0



This Program operates in a similar way to 'Solution of X = F(X)', except that it uses the Newton-Raphson method to solve F(X) = 0. Many roots converge very rapidly, and there is less chance to use the ZOOM facility effectively.

OPERATION

1. Choose the graphics MODE (if in doubt, MODE 1)
2. Type in the function, F(X), and its derivative, F'(X), in standard BASIC. (If your differentiation lets you down, F'(X) will be rejected!)
3. Enter XMIN,XMAX and YMIN,YMAX (rescaling is possible later)
4. While the graph of F(X) is showing:
   - (a) Enter a STARTING VALUE for X, and press the SPACE BAR for successive approximations, as with X = F(X).
   - (b) Corresponding values of X and Y are displayed. When Y = 0, "ROOT FOUND" is displayed. If divergence occurs, "DIVERGENT" is displayed.
5. Once successive approximations have been started, the following single keys may be pressed:

   I - to 'ZOOM IN' (suitable for convergence)
   O - to 'ZOOM OUT' (suitable for divergence)

   R - to RESCALE and RERUN, same F(X)
   S - for a NEW STARTING VALUE
   N - to start again, new F(X)

   <esc> to return to the MENU.

RED FUNCTION KEYS - (for instant Program demonstrations)

$\boxed{f_0}$ -

F(X) =	X∧3 - 2	<ret>
F'(X) =	3*X∧2	<ret>
XMIN,XMAX	-1,2	<ret>
YMIN,YMAX	-3,6	

Suggested Starting Value:
X = 2
Press SPACE BAR to continue.

$\boxed{f_1}$ -

F(X) =	X∧3 - 10*X - 10	<ret>
F'(X) =	3*X∧2 - 10	<ret>
XMIN,XMAX	-5,5	<ret>
YMIN,YMAX	-20,20	

Suggested Starting Values:
X = -2, -1.8, -1.7, 1

The documentation for any mathematical software can make a vital contribution to its successful use. The M.E.I. Programs were written to help introduce teachers and students to the benefits of the micro as an interactive aid to learning. The associated documentation gives operational guidelines for each of the 18 Programs, and this includes worked examples and demonstrations stored in the BBC Micro's 'red function keys'.

Programs such as these provide a dynamic new medium for classroom presentations, but the micro needs to be treated as an incidental resource; it should ideally be available all the time, but used only when required.

SELECTIONS FROM THE M.E.I. A LEVEL IN MATHEMATICS

1. PURE MATHEMATICS: GRAPHS and FUNCTIONS

SYLLABUS and SAMPLE EXAMINATION QUESTIONS

Elementary two dimensional co-ordinate geometry. Equation of the straight line and the circle. Distance and section formulae, gradient, parallel and perpendicular lines. Length of a perpendicular from a point to a line.

Candidates should know that the curves $\frac{x^2}{a^2} \pm \frac{y^2}{b^2} = 1$, $y^2 = 4ax$, $xy = c^2$ are called ellipse, parabola, hyperbola (as appropriate), but no formal properties will be assumed.

Polar co-ordinates in two dimensions.

Simple curve sketching. e.g. $r=a(1+\cos\theta)$ $r=a\cos 2\theta$ according to the convention $r \geqslant 0$.

Manipulation and solution of simple inequalities.

e.g. $(x-1)(2-x) > 0$, $x^2-2x+2>0$, $\frac{1}{x-1} < 2$, $\frac{3x}{x^2-1} > 2$, $|2x+3| < 16$.

Graphical interpretation of inequalities in two variables.

Treatment and sketching of graphs of the form $y=f(x)$, $y^2=f(x)$ where $f(x)$ is a simple rational function, and simple modulus functions of the form $y=|f(x)|$, $y=f(|x|)$. Treatment and sketching of curves with simple parametric forms. Their tangents and normals.

The graphical effect of simple transformations on the curve $y=f(x)$. The relationship between the graph of a function and of its inverse.

Recognition of odd and even functions and the corresponding properties of their graphs.

Graphical representation of simple functions including such functions as $\arcsin x$ and $e^{-x}\cos x$.

Maclaurin series. Approximate evaluation of a function.

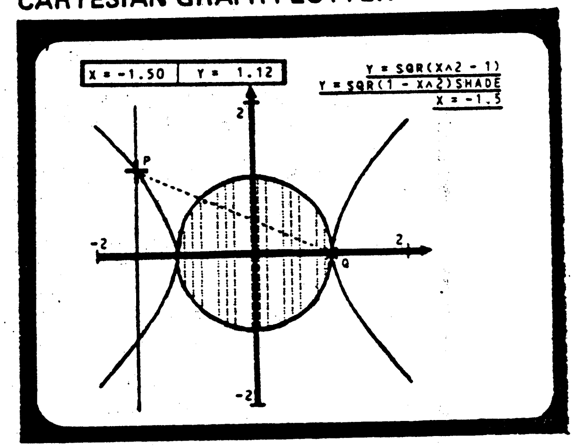
Candidates will be expected to recognise the series for e^x, $\sin x$, $\cos x$ and $\ln(1+x)$.

- Find the range of values of x for which
$$\frac{1}{1-x} \leqslant \frac{1}{1+x}.$$

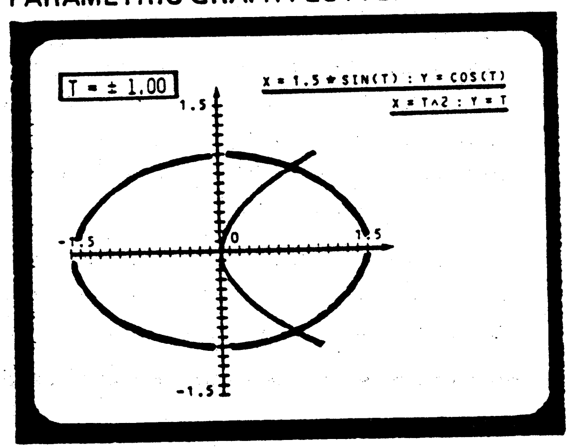
- In separate diagrams, sketch the curves whose equations are $r\cos\theta = 1$ and $r = \cos 2\theta$, where r and θ are plane polar coordinates with $r \geqslant 0$.

ASSOCIATED TEACHING SOFTWARE

CARTESIAN GRAPH PLOTTER



PARAMETRIC GRAPH PLOTTER



Find the coordinates of the stationary points of the curve $y = \dfrac{x^3}{x^2 - 1}$, distinguishing between maxima, minima and points of inflexion.

Sketch the curve, showing clearly the positions of all asymptotes. Deduce the range of values of a for which the equation $x^3 = a(x^2 - 1)$ has only one real root.

A curve is given by the parametric equations

$$x = 1 + e^t, \quad y = 2t - \ln t.$$

Find $\dfrac{dy}{dx}$ and $\dfrac{d^2y}{dx^2}$ in terms of t.

Find also the equation of the normal to the curve at the point where $t = 1$.

Given the complex number $z = 1 + \cos\theta + j\sin\theta$, where $0 \leqslant \theta \leqslant \pi/2$, show by using an Argand diagram, or otherwise, that

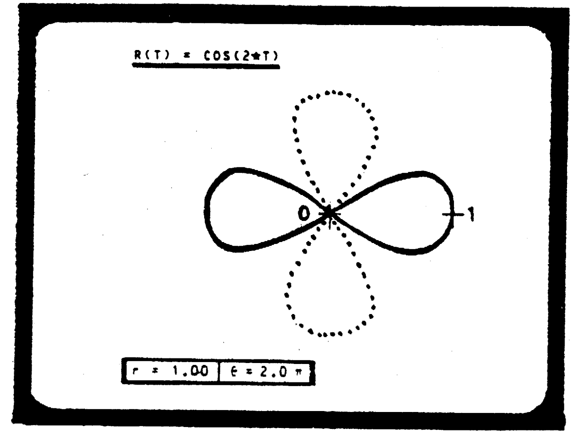
$$\arg z = \frac{\theta}{2} \quad \text{and} \quad |z| = 2\cos\left(\frac{\theta}{2}\right).$$

Find $\arg(2z)$, $|z^*|$ and $|z - 1|$.

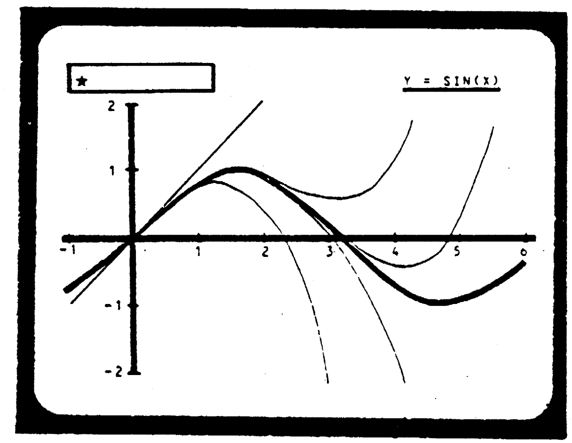
Sketch the locus of z in an Argand diagram as θ varies from 0 to $\pi/2$ inclusive.

Find the first two derivatives of $f(x) = \dfrac{e^{-x}}{1+x}$. Hence, using Maclaurin's method, find the first three terms in the expansion of $f(x)$ in ascending powers of x.

POLAR GRAPH PLOTTER



MACLAURIN EXPANSIONS



2. PURE MATHEMATICS: TRANSFORMATIONS and MATRICES

SYLLABUS and SAMPLE EXAMINATION QUESTIONS

Evaluation of determinants of order two and three. Matrix addition and multiplication. Linear transformations in a plane, their associated matrices and their manipulation. The significance of a zero determinant. Meaning of transpose and inverse.	The product rule for inverses. Evaluation of inverses of 2X2 and 3X3 matrices. The solution of. the matrix equation Ax=b in the following cases: (i) A is 2X2 or 3X3 non-singular (ii) A is 2X2 or 3X3 singular (iii) A is 2X3. Geometrical interpretation of the solution.

- The matrix
$$\mathbf{M} = \begin{pmatrix} 2 & 1 & -1 \\ -1 & \lambda & 2 \\ 1 & 1 & 0 \end{pmatrix}.$$

For what values of λ does **M** have an inverse? In the case where $\lambda = 3$, find the inverse $\mathbf{M}^{-1}$ and hence, or otherwise, find the column vector **x** such that

$$\mathbf{Mx} = \begin{pmatrix} 2 \\ 1 \\ -1 \end{pmatrix}.$$

- (i) Determine the matrix $\mathbf{M}_1$ such that $\mathbf{M}_1\begin{pmatrix} p \\ q \end{pmatrix} = \begin{pmatrix} q \\ -p \end{pmatrix}$ for all values of p and q, and state the nature of the transformation represented by $\mathbf{M}_1$.

  (ii) Find the matrix $\mathbf{M}_2$ which represents a reflection in the line $y = x$.

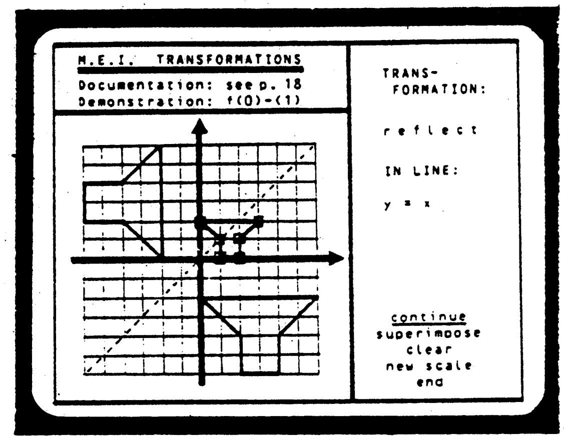
  (iii) Find the images of the straight line $y = 3x - 1$ under

  (*a*) the transformation represented by $\mathbf{M}_1$ and

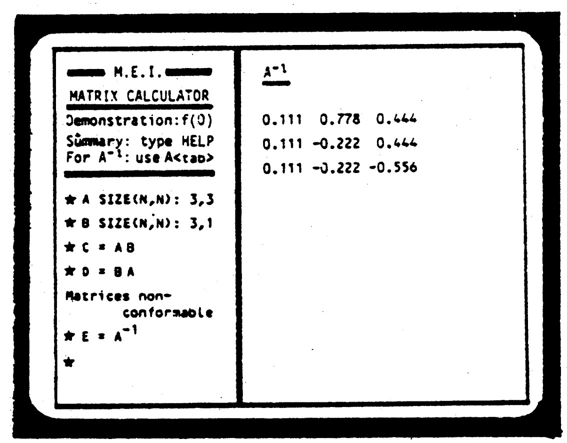
  (*b*) the transformation represented by $\mathbf{M}_2$.

ASSOCIATED TEACHING SOFTWARE

TRANSFORMATIONS



MATRIX CALCULATOR



3. PURE MATHEMATICS: NUMERICAL METHODS

SYLLABUS and SAMPLE EXAMINATION QUESTIONS

Candidates should appreciate the importance of the estimation of errors in any numerical calculation. Interpretation of, or correction of errors in, a simple flow diagram. Estimation of maximum error in the result of a short calculation for given bounds of data.	Simple examples of iterative methods; solution of $x=F(x)$ by $x_{n+1}=F(x_n)$, solution of $f(x)=0$ by the Newton-Raphson process. Numerical integration, by trapezium rule or by Simpson's rule. Reduction of laws to a linear form and their graphical interpretation. The use of linear interpolation

● Show that the cubic equation $x^3+2x-11 = 0$ has only one real root and further that the root lies between $x = 1$ and $x = 2$.

Two possible iterative schemes for finding the root are

(i) $x_{n+1} = (11-x_n^3)/2$ and (ii) $x_{n+1} = (11-2x_n)^{\frac{1}{3}}$.

Show that only one of these schemes converges from an initial estimate of $x = 2$ and hence find the root correct to 3 d.p., justifying the accuracy of your answer.

● Experimental values of a continuous function $f(x)$ were:

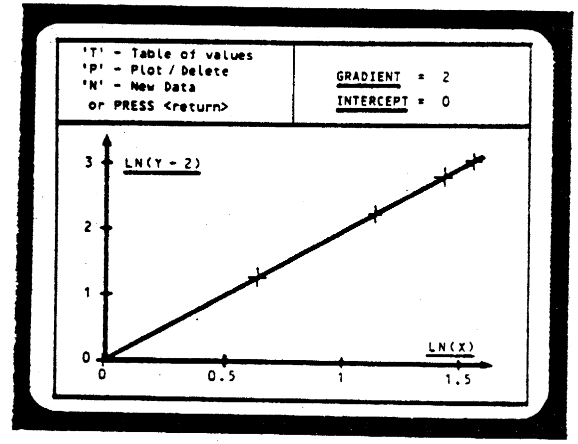
x	0	0.2	0.4	0.7	0.8	1.0	1.2
$f(x)$	1.285	1.114	0.944	0.706	0.634	0.500	0.384

Use linear interpolation to estimate

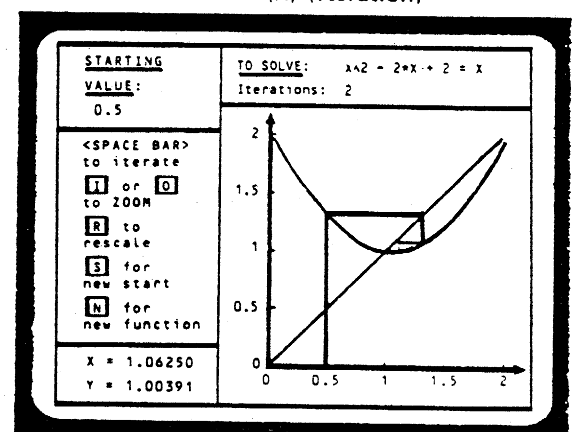
(i) $f(0.6)$; (ii) a value of x for which $f(x) = 1$.

Estimate $\int_0^{1.2} f(x)\,dx$ using Simpson's rule with seven ordinates.

VARIATION & EXPERIMENTAL DATA



SOLUTION OF x = F(x) (Iteration)



4. APPLIED MATHEMATICS: PROBABILITY and STATISTICS

SYLLABUS and SAMPLE EXAMINATION QUESTIONS

Data presentation: Efficient procedure for classification and visual presentation of data and the computation of central and dispersion statistics. Informal discussion of the planning of a simple experiment and of exhibiting and processing the observations. Histogram, cumulative frequency curve. Mean, median, mode, range, interquartile range, mean deviation, standard deviation. The mean, variance and mode of a binomial distribution.	Situations leading to a Poisson distribution and calculations of probabilities, mean, variance and mode. The Poisson approximation to the binomial distribution. The normal distribution: (i) recognition of its p.d.f. (ii) as an approximation to the binomial and Poisson distributions.

● Two fair dice are thrown 72 times. Calculate to 3 decimal places the probability p that the total score is 12 on two or fewer of the 72 throws. Also obtain two approximations to p using

(*a*) the Normal approximation to the Binomial, and

(*b*) the Poisson approximation to the Binomial.

Comment on the different accuracies of the two approximations.

● The numbers of customers entering a shop in forty consecutive periods of one minute are given below.

3	0	0	1	0	1	0	1	2	0	2	1	0	1	2	2	1	0	1	1
0	3	4	1	2	3	1	0	0	2	1	0	3	1	2	0	2	0	3	1

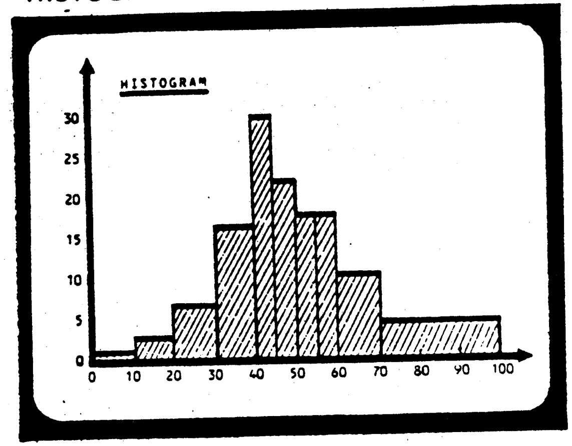
Draw up a frequency table, and illustrate it by means of a bar chart.

Calculate values for the mean and variance of the number of customers entering the shop in a one minute period. Fit a Poisson distribution to the data, and comment briefly on the degree of agreement between the calculated and observed frequencies.

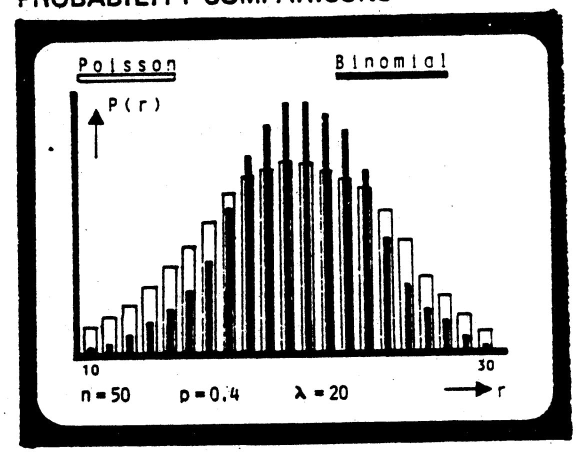
Estimate the probability that no customers enter the shop in a given two minute period.

ASSOCIATED TEACHING SOFTWARE

HISTOGRAM/CUMULATIVE FREQUENCY



PROBABILITY COMPARISONS



5. APPLIED MATHEMATICS: LINEAR REGRESSION ANALYSIS

SYLLABUS and SAMPLE EXAMINATION QUESTION

Bi-variate samples: Interpretation of scatter diagrams. Calculation and use of (i) covariance (ii) product-moment correlation coefficient.	Notation: cov(X,Y), r_{XY}
Linear regression: Calculation of regression lines. Their use for prediction and estimation.	Calculation of residuals will not be required.

Drums of hair shampoo are kept in storage for some time before being rebottled for retail sale. During storage, evaporation of part of the water content takes place and an examination of several drums gave the following results:

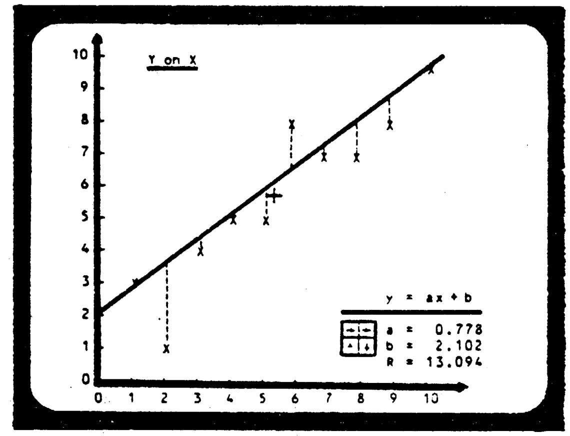
Drum number	1	2	3	4	5	6
Storage time (weeks)	2	5	7	9	12	13
Evaporation loss (ml)	38	57	65	73	84	91

Calculate the product-moment correlation coefficient, and hence comment on the appropriateness of using a linear regression model for predicting evaporation loss.

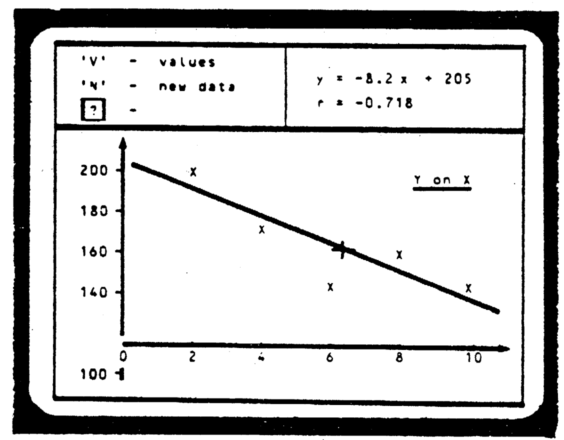
Find the line of regression of evaporation loss on storage time, and estimate the evaporation loss for a drum kept in storage for eleven weeks.

Why would you not expect to get good estimates from the line of regression for evaporation loss when the storage time is very long? In such a case, suggest, with reasons, whether the line of regression would under-estimate or overestimate the evaporation loss.

LINEAR REGRESSION (1)



LINEAR REGRESSION (2)



6. APPLIED MATHEMATICS: SAMPLING THEORY

SYLLABUS and SAMPLE EXAMINATION QUESTIONS

Random sampling from a population: Definition of a random sample. Estimation of the mean and variance of the population, from a random sample. Distribution of the mean of a sufficiently large sample. Standard error of the mean.	Confidence intervals for the mean using the normal distribution. Hypothesis Testing: for a single mean, using the normal distribution.

● (*a*) For quality control purposes, every twentieth refrigerator to come off the assembly line in a factory is tested. What criticism can be made of this method of sampling? Outline briefly how a random sample might be obtained in this case.

(*b*) In a large city the distribution of incomes per family has a standard deviation of £5200. For a random sample of 400 families, what is the probability that the sample mean income per family is within £500 of the actual mean income per family? Given that the sample mean income was, in fact, £8300, calculate a 95 % confidence interval for the actual mean income per family.

● The weights of steaks sold by a supermarket are distributed Normally with mean μ and standard deviation 0.02 lb. A quality control inspector tests the hypothesis that $\mu = 1$ lb at the 5 % level of significance. He takes a random sample of 5 steaks whose weights (in lb) are:

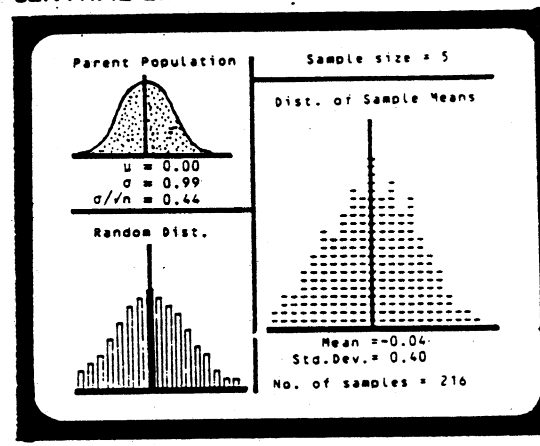
0.977, 1.014, 0.989, 0.972, 0.968.

His null hypothesis is that $\mu = 1$ lb, and he performs a two-tailed test. State his alternative hypothesis and perform the test.

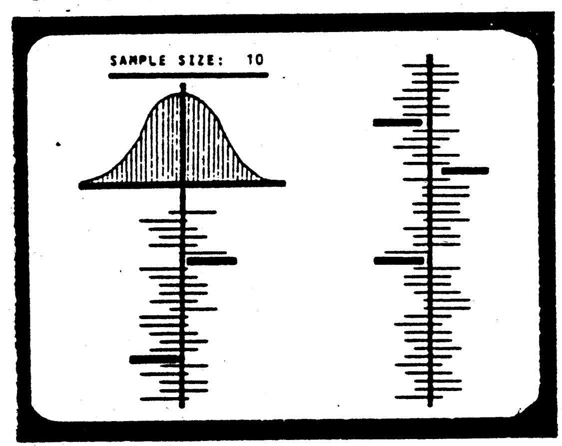
Another inspector is employed to check that customers are not (on average) sold underweight steaks. If he had conducted a one-tailed test using the same random sample, the same level of significance and the same null hypothesis, what would have been his alternative hypothesis, and his conclusion?

ASSOCIATED TEACHING SOFTWARE

CENTRAL LIMIT THEOREM



SAMPLING: CONFIDENCE INTERVALS



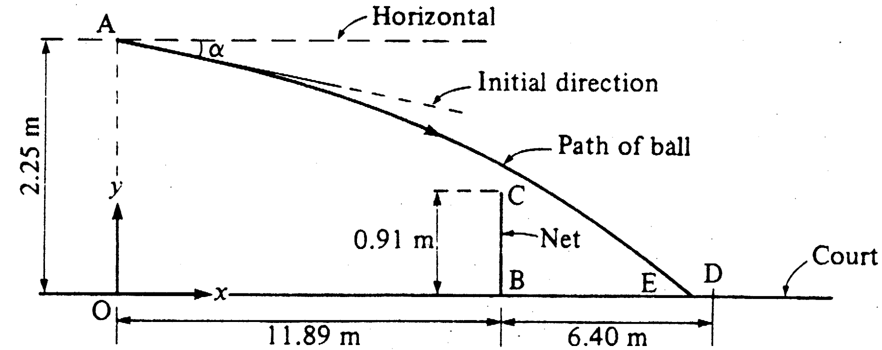
7. APPLIED MATHEMATICS: APPLIED CALCULUS and MECHANICS

SAMPLE EXAMINATION QUESTION

In several A level schemes Mechanics is offered as an alternative to Probability & Statistics, as it is with M.E.I.. However, in mosts countries Mechanics is taught outside the Mathematics syllabus (usually with Physics), and so it receives only a short mention here.

Clearly a computer simulation can be a valuable teaching aid for many topics in Mechanics.

For example, to prepare students for this question on projectiles, a much larger number of situations can be studied through simulation than was ever possible without the calculating and visual power of the computer.



● The sketch (not drawn to scale) shows part of the path AE of a tennis ball after service. The ball is served from A, where A is 2.25 m vertically above O, its initial speed is 150 km h^{-1}, and the direction of projection is at an angle α below the horizontal. Air resistance can be neglected and $g = 9.81\,\text{m s}^{-2}$. Show that when the ball has moved x m horizontally, its height above the ground is y m, where

$$y = 2.25 - x\tan\alpha - 0.002825x^2\sec^2\alpha,$$

approximately (assuming that the ball has not hit the net or the ground).

Verify that the ball clears the net BC, where OB = 11.89 m and BC = 0.91 m, when $\alpha = 4°$ but not when $\alpha = 5°$.

Determine whether the service is 'legal' when $\alpha = 4°$, i.e. determine whether BE is less than BD, where BD = 6.40 m and E is the point where the ball first bounces.

8. FURTHER NUMERICAL METHODS - Syllabus and Sample Examination Question

(a) The approximations $\left(\frac{dy}{dx}\right)_0 \approx \frac{1}{2h}(y_1 - y_{-1})$ and $\left(\frac{d^2y}{dx^2}\right)_0 \approx \frac{1}{h^2}(y_1 - 2y_0 + y_{-1})$ where y_{-1}, y_0, y_1 are the ordinates at $x-h, x, x+h$. Applications to approximate step-by-step solution of $y'=f(x,y)$ and $y''=f(x,y)$ over a short range with suitable starting conditions.
(b) The properties of polynomial differences and their use as a checking or calculating procedure.
(c) The candidate will be expected to have a knowledge of the following:
The estimation of errors.
How numerical integration can be carried out to any required degree of accuracy.
The effect of rounding and truncation errors.

● Show that if h is small and $y(x)$ is suitably continuous, the value of $y''(x)$ at $x = a$ is given approximately by

$$y''(a) \approx \frac{1}{h^2}\left\{y(a+h) - 2y(a) + y(a-h)\right\}.$$

Given that $y(x)$ satisfies the differential equation

$$y'' - y = -2e^{-x},$$

with conditions $y(0) = 0$, $y'(0) = 1$, and using the notation $y_r = y(x_r)$, $x_{r+1} = x_r + h$, show that $y_{r+1} \approx (h^2+2)y_r - y_{r-1} - 2h^2e^{-x_r}$, $(r \geqslant 0)$. If $x_0 = 0$, find an expression for y_{-1} in terms of y_1 and h **without** using this recurrence relation. Hence obtain an estimate for $y(0.5)$, taking $h = 0.1$.

The numerical methods syllabus continues for more advanced students, including some error analysis and the numerical solution of differential equations. In the early years, candidates sitting this examination had a special room on account of the noise of the hand calculating machines.

The influence of computing on the teaching of mathematics at senior secondary level is thus shown to be dramatic in a number of topic areas, if used wisely, and we can expect this influence to become more widespread as more hardware is installed in schools. In the longer term, as symbolic algebra and calculus systems become established, perhaps in hand held devices, it is to be hoped that school syllabuses will use the extra freedom to explore the modelling process in a greater variety of applications rather than extend to further topics.

REFERENCES

(1) The complete syllabuses for M.E.I. A level 'Mathematics' and 'Further Mathematics' - published by The Oxford and Cambridge Schools Examination Board, 10 Trumpington Street, Cambridge, Great Britain.

(2) The examination questions from recent M.E.I. A level papers in 'Mathematics' are reproduced by permission of the Oxford and Cambridge Board.

(2) "Programs for Mathematical Computing" (for the BBC Micro), by Philip Couzens, and the Documentation, by Philip Couzens and Douglas Butler - 2nd Edition, February 1985 - published by The M.E.I. Schools Project, 41a West Street, Oundle, Peterborough, Great Britain.

Douglas Butler is Head of Mathematics at Oundle School, Peterborough, and Chairman of the M.E.I. Schools Project.

Philip Couzens is Head of Mathematical Computing at Oundle School.

rom
Irish Mathematical Society
Newsletter No.7, March 1983.

COMPUTER SCIENCE AND THE MATHEMATICS CURRICULUM

Anthony Karel Seda

"As soon as an Analytical Engine exists, it will necessarily guide the future course of the science. Whenever any result is sought by its aid, the question will then arise - By what course of calculation can these results be arrived at by the machine in the shortest time?"

Charles Babbage, 1864.

§1. Introduction

In any discussion of computer science and its relationship with mathematics, from an educational viewpoint, certain obvious questions come to the fore:

(1) What is the role of mathematics in computer science education?

(2) What is the role of computer science in mathematics education?

(3) What is, or has been, the response of mathematicians to computer science in relation to the mathematics curriculum?

There are two viewpoints, at least, from which these questions can be contemplated. One is that of the computer scientist engaged in teaching/research in a third level institution peering over the ramparts at the mathematicians. The other, which is ours, is that of the mathematician similarly engaged in teaching/research and similarly peering at the computer scientists. Having thus declared my vantage point, and for reasons of space, I wish to concentrate here on Question 1, and only to touch on Questions 2 and 3. Specifically, I wish to bring to the attention of readers of the *Newsletter* the discussion contained in the articles [4] and [5] of Professor Anthony Ralston. Ralston's conclusion is,

"It is time to consider (i.e., try)
an alternative to the standard
undergraduate mathematics curriculum
which would give discrete analysis an
equivalent role to that now played
by calculus in the first two years
of the undergraduate curriculum".

In §3 I have listed the topics which Ralston proposes in order to achieve his aim. Actually, [4] is a detailed version (83 pages) of [5], and [5] will suffice to support the main thread of the argument here.

In the quotation above, Babbage is of course talking about algorithms, and algorithms in the words of Knuth [3], are "... really the central core of the subject (computer science), the common denominator which underlies and unifies the different branches". Indeed, Knuth has, just prior to writing this, chosen to describe computer science as "the study of algorithms". Now, as confirmed by Knuth, the study of algorithms is very mathematical and it is worth stating this fact in order to dispose of the short, negative reply to Question 1 which just might be proposed from the other vantage point! Further confirmation of this fact, i.e. of the mathematical nature of computer science, can be gained by consulting the list of topics in Section 68 of the 1980 Subject Classification of Mathematical Reviews, or by actually reading some recent reviews in this section; see also [1].

2. Some History and Some Educational Philosophy

Whilst our main discussion centres on Question 1, it will not be out of place to devote a few words to Questions 2 and 3.

One might wonder why it is today that there is a division between computer scientists and mathematicians, and that there is not more sympathy shown by each for the other's subject. After all, computer science grew out of mathematics and in its early days, some twenty five-thirty years ago, it was necessarily closely bound to mathematics. However, today, digital computers vastly predominate over analogue computers and digital computers are essentially discrete. What, though, is being taught in most mathematics departments? I suspect that it is largely either continuous mathematics, such as analysis, or relatively abstract mathematics, to the great exclusion of discrete mathematics. Certainly this is true in U.C.C., but may be less so in non-university departments. Indeed, Ralston [4] argues that in American universities the present-day structure of the mathematics curriculum (mainly calculus/linear algebra - at least in the first two years) has come about for reasons more to do with history and inertia (human) than with a judicious choice of topics to meet the educational requirements of those students other than majors in physical science and engineering.

As far as Question 3 is concerned, there are at least three discernible responses:

(a) Ignore the problem - maybe it will go away.

(b) Continue teaching traditional material but illuminate it with examples/projects worked on the computer.

(c) Meet the problem head-on and design/update courses to more nearly meet the needs of those students studying computer science.

Response (a) needs no comment: (b) is outside the scope and limits of this note but surely has a lot of merit, see [2] and its references for some experiments, and also elsewhere in this *Newsletter*; (c) is the main topic of this discussion, see §3.

Before leaving this section, there is another aspect worth noting. Mathematics courses are widely held to be educational, irrespective of their content, for purposes of training the mind. Can the same be said of computer science? This touches on Question 2, because the solution Ralston has in mind for (c) is best framed in terms of a mathematical sciences degree programme and, naturally, the educational value of such a programme, over and above its content, has to be considered. To quote G.E. Forsythe, see [3], "The most valuable acquisitions in a scientific or technical education are the general-purpose mental tools which remain serviceable for a lifetime. I rate natural language and mathematics as the most important of these tools, and computer science as a third". Some of Knuth's own views on this can also be found in [3].

3. Ralston's Proposals for the Mathematics Curriculum

I want, now, to list the topics which Ralston believes could form a suitable basis for the discrete component in a better balanced curriculum for mathematics students, computer science students and others. The headings below are taken from [4] and [5] and the topics from [4].

i) Algorithms and their Analysis. Topics: the notion of an algorithm: notation for expressing algorithms: basic analysis of algorithms.

ii) Introductory Mathematical Logic. Topics: the notion of mathematical proof; the propositional calculus; Boolean algebra: the notation of the predicate calculus; introduction to the verification of algorithms.

iii) Limits and Summation. Topics: the notion of infinite processes: ideas of convergence and limits: limits of discrete functions: summation.

iv) Mathematical Induction. Topics: the principles of induction: examples of induction proofs.

v) The Discrete Number System. Topics: real numbers and finite number systems; definition and laws of the discrete number system; number bases other than 10.

vi) Basic Combinatorial Analysis. Topics: the binomial theorem and Stirling numbers; permutations and combinations; simple combinatorial algorithms.

vii) Difference Equations and Generating Functions. Topics: recurrence relations; linear difference equations and their solution; generating functions.

viii) Discrete Probability. Topics: basic laws; discrete probability distributions; random number generation; queueing theory; probability and algorithm analysis.

ix) Graphs and Trees. Topics: basic definitions and theorems of graph theory; path and colouring problems; tree enumeration and binary trees.

x) Basic Recursion and Automata Theory. Topics: basic definitions; recursive algorithms; recursive functions; regular sets and expressions; finite state machines; languages and grammars; Turing machines.

In connection with this list, the following points should be noted:

(A) These topics are only suggestions. Moreover, it is assumed by Ralston that they will be presented in some combination with abstract algebra, linear algebra, analysis etc., for in [4] it is observed that "... there are numerous areas of computer science where calculus plays an important role..." Moreover, a better balanced curriculum is being argued for, but not a complete reversal in favour of discrete mathematics.

(B) These topics are, with the possible exception of some in viii), mathematics subjects and as such are best taught by mathematicians.

(C) Due to the differences between the educational systems here and in America, certain additions and subtractions might need to be made were these proposals to be adapted to fit into our context (Probably extra more advanced material such as more computability theory or computational complexity could be added for, say, honours students).

(D) These proposals are at least worthy of consideration, for Professor Ralston has wide experience in both computer science and mathematics and backs up his suggestions with an exhaustive study.

More questions are asked here than are answered. For example, consideration needs to be given to the feasibility of such topics for various types of student, ranging from students of management through to honours mathematics students. But space permits no more comment, and for answers to such questions the reader must either consult [4] and [5] or, if Ralston [5] page 484 is correct, undertake experiment for himself or herself.

Educational problems are not usually very well defined; they are likely to be controversial and to raise temperatures. Indeed it may be that Ralston's criticism does not apply here and that all is well. If not, and this article creates some discussion or starts people thinking about the problems raised here, then it will have achieved its purpose. We hardly need reminding in 1983 that computer science is a major undergraduate subject. But what has perhaps not been widely recognised yet is the fact that the next generation of students will be taught computer science in secondary schools by those currently studying it at third-level. Future incoming students may therefore elect to study computer science "because it is familiar" just as many do now, I suspect, in the case of mathematics.

References

[1]. E.W. Dijkstra, Programming as a Discipline of Mathematical Nature, Amer. Math. Monthly, 81 (1974), 608-612.

[2]. S.P. Gordon, A Discrete Approach to Computer Oriented Calculus, Amer. Math. Monthly, 86 (1979), 386-391.

[3]. D.E. Knuth, Computer Science and its Relation to Mathematics, Amer. Math. Monthly, 81 (1974), 323-343.

[4]. A. Ralston, Computer Science, Mathematics and the Undergraduate Curricula in Both, Technical Report 161, Department of Computer Science, SUNY at Buffalo, 1980.

[5]. ———— Computer Science, Mathematics and the Undergraduate Curricula in Both, Amer. Math. Monthly, 88 (1981), 472-485.

Department of Mathematics,
University College,
Cork.

COMPUTER AWARE CURRICULA: IDEAS AND REALISATION

Hugh Burkhardt
Shell Centre for Mathematical Education
University of Nottingham, England

0. Introduction

This paper offers a series of comments on the three main areas set out in the background paper (1), numbered correspondingly; it focuses on the processes of curriculum change.

0.1 Curriculum Planning - Ambition and Realism

Before getting down to the task of throwing ideas and comments into the pool which this meeting provides, there are some general points to be made about the nature of the exercise. It is speculative - a conference for conjectures; as in mathematics itself such activity is creative and important, but the outcomes should be seen as entirely provisional. We can have no reliable idea how far any suggestions we put forward will prove feasible in <u>any</u>, let alone <u>every</u> educational system. Even if they are implemented reasonably faithfully, the full curriculum reality of what occurs will contain many surprising side effects; more likely, the translation from an idea to a small scale pilot experiment with exceptional teachers and facilities, and then to large scale reality will involve critical distortions of the aims of the exercise which may call in question it's value.

This is not at all a negative viewpoint; I think it better to be sharply aware of these dangers from the beginning so that they can be minimised in planning the development process.

In case there are any who believe that I exaggerate the dangers, let me draw attention to a few examples so everyone can see what I have in mind:

> The splendid Bourbaki enterprise was launched to establish a firmer foundation for mathematical education in <u>school</u> (2); few now see that as among the positive contributions it has made, while many are concerned at the over-emphasis on formalism that has widely emerged from the movement (15 year old children who can accurately define an equivalence class, but cannot name an example of one, and so on).
>
> SMALLTALK was devised by the Xerox Learning Research Group largely to produce a medium, the DYNABOOK, that would be "as natural to a child as pencil and paper" (3); what has emerged is perhaps the most sophisticated graphics orientated data management system so far - an important achievement, but a very different thing. SMALLTALK has not, at any rate, done any harm to the school curriculum, and its offspring, such as the Mackintosh microcomputer, may yet contribute.
>
> The same cannot necessarily be said for much that until recently went under the name of "Computer Studies" - large amounts of fact learnt and one or two trivial programs produced with infinite tedium on distant facilities - a remarkable way to illustrate the miraculous speed of a wonderful medium.
>
> My final example must be the reform movement of 25 years ago in mathematical education - "new math", "modern mathematics" and so on. Comparison of the initial aims agreed at conferences such as this, the pilot schemes in a few exceptional schools, and the classroom reality of today show the contrasts vividly. For example, in England the applications of mathematics occupied a central place in the original design; in most of the major courses that emerged they are not to be found. Equally, new mathematical concepts were introduced but often with none of the pay off that motivated their inclusion - because the serious examples originally envisaged proved too difficult for most students, and were replaced with trivial ones.

What are we to do about this? This is not the place for a serious discussion of methodologies of research and curriculum development (4). Very briefly, there is no proven successful answer but some seem to be less susceptible to such corruption than others. I believe that the essence is an empirical approach - find out what actually happens to your draft ideas in practice, in circumstances sufficiently representative of what you are aiming for, and then revise the materials repeatedly until they work in the way intended. We have found that structured classroom observation is a key ingredient in this approach (5).

0.2 **The Moving Targets**

One other unusual factor makes curriculum development involving advanced technology more difficult than usual. It is the mismatch of time scales between technical change (¼ one year) and curriculum change (¼ ten years). The curriculum designer can not assume a specific level of technological provision and sophistication in schools - it will vary widely both in time and from place to place.

This is important. If each student has a "micro", curriculum possibilities open up which are not there with one micro per class; these possibilities depend on the sophistication of the micro - one line of display, a few lines, many lines, graphics, access to data - each step is highly significant. Equally it is already clear that low levels of provision and sophistication still have enormous educational potential. Is technical restraint a virtue, or does it impede progress?

Again there is no proven strategy for this new situation. It is likely that flexibility will pay off - perhaps a modular approach within a broadly defined framework of aims, allowing everyone to operate at their best level. The materials too should be flexible, working well in different styles and modes of classroom or "private" use.

The other targets that should be thought about are the teachers and the students. I shall say little about the latter because they will not be forgotten. It is the teachers that will face the greatest difficulties; changing well established ways of working is extremely difficult, particularly when teaching style is involved - as it must be. As in any other highly skilled occupation, levels of performance of mathematics teachers vary enormously; what works for the exceptional few will not usually be accessible to a broader target group. The situation is very different in the secondary (16-19) and tertiary (18-22) centres. In many countries the greater independence of the tertiary teacher, who has more control over curriculum and assessment, means that it is easier to make experimental reforms but harder to implement them on a larger scale; the stricter curriculum constraints on the secondary teacher place heavy responsibilities on the innovator who aims at large scale change.

It is not enough to talk of in-service training as a solution to such difficulties; it has to be shown that it will be effective. Studies and history suggest that changes of syllabus content have been achieved but that, except for a small minority of teachers, changing the pattern of classroom learning activities has not so far proved possible.

This, however, is widely regarded as the central challenge of mathematical education. Everywhere the curriculum is dominated by (6)

teacher explanation   illustrative examples   imitative exercises;

This leads to more rapid apparent student progress, but the skills acquired are not usable on non-routine problems or in the world outside the classroom. To achieve the flexible competence of understanding that this requires, the pattern of classroom activities has to be widened to include some which give more initiative to the student. It is encouraging that the micro has shown great promise in this regard.

All change is threatening. Technology appears to reduce this threat, partly because it produces an obviously new situation and thus does not imply criticism of the teachers existing modes of operation. This more than compensates for the extra barrier of learning to use the equipment - provided it is reliable.

1 **Changes in Mathematics**

I shall not say much under this heading, because it has received a lot of attention; what I say will relate fairly directly to curriculum questions. The main areas of change in mathematics are outlined in the background paper for this meeting. Many of the issues are much more general than the technological background that brought them to the focus of our attention. This is often so, and is equally important in the curriculum and classroom dynamics domains.

Decisions on how far any change penetrates at any level will only emerge from experiment. Those of us who began 20 years ago to use symbolic manipulation programs first to check, and then to do "heavy algebra" are sensitive to the balance of cost-effectiveness in acquiring personal skills, and how this balance changes with time. 30 years ago, one Nobel Laureate in Physics said that the most useful thing he ever did was to learn the multiplication tables up to 24; we should doubt the value of that today, but we have always tended to underrate the knowledge base of able mathematicians.

There are interesting questions for research here. Their relevance to mathematical education will be slight unless that is their focus. Forefront developments in mathematics or any other subject do not often impinge on the taught curriculum - "modern mathematics" in schools was almost entirely 19th century, and the same is broadly true of undergraduate courses. Recent developments will have to justify a curriculum slot against stiff competition, as well as entrenched opposition; we need the evidence to support their inclusion in curriculum terms. Ideas for such studies would be a useful outcome of this meeting.

It seems to me that the central challenge of any new medium is to acquire enough skill with, and understanding of it, so that it becomes a powerful tool - and not a nett-absorber of effort and attention. Otherwise, one is replacing mathematics by computer studies - another possibility but not our goal here. The ambition to provide a resource as natural to the mathematician, and to the mathematics student, as pencil and paper, remains a good one. It will not be easy. We shall be able to provide procedures for the student to follow, as at present, which may well bring some further insight - the danger is that we shall be content just to provide more of this kind of fairly passive, imitative learning.

Thus, if the new medium is treated seriously, it will probably bring better understanding but take more time. It should bring about a reduction in total syllabus content. For example, the interrelation between numerical, graphical and analytic methods of handling a mathematical situation, their respective strengths and weaknesses, is not easy to master but is essential both for understanding and for action. The normal path of curriculum development, for example the movement to introduce more "discrete mathematics", is likely to lead to the opposite effect. The present course on calculus will not prove dispensable and history suggests that the tendency will be to arrive at a compromise with greater total content than at present; this inevitably leads to an even greater emphasis on imitation. The alternative of earlier specialisation avoids such hard choices by transferring them to the student.

In one area there seems likely to be clear gain. The new central role of algorithms, including their design rather than simply their execution, is a rich field for developing both technical and higher level skills. Algorithms are, I believe, inherently less abstract that the implicit relationships (such as equations to solve) that dominate the mathematics curriculum. The work of David Johnson and others at Minnesota in the 1960's and 1970'sshowed that programming could provide a semi-concrete bridge to abstract thinking that enabled many more children to achieve some fluency in school algebra. It is likely that similar gains can be established in the 16-22 age range. It may be useful to take a broader, less formal view of algorithms, with emphasis on graphical processes. Perhaps even human processes, such as negotiations of criteria in solving a problem, may usefully be brought within the algorithm framework.

Finally, another word of warning - because of the imitative nature of the curriculum, it is easy to get a quite false picture of the student as mathematician. A mathematician has command of a range of concepts and techniques (or knows where and how to get such command) and uses them autonomously to express and manipulate ideas and relationships to get answers and understanding. There is clear evidence that, on such criteria, students are several years atleast behind their performance on imitative exercises. The calculator is a useful resource because students can use arithmetic for a range of purposes; in contrast it has been shown (7) for example, that even very bright 17 year old students do not use algebra at all as an automonous mode of expression, though they have had 5 years of success in manipulating it; the benefits of a machine that will manipulate in a language they do not speak are elusive, and maybe illusory.

2. Curricula

Computers and informatics can influence the mathematics curriculum in at least two different ways. Some new developments in mathematics will displace part of the current content because we come to believe that students should learn about them; I shall not say much more about such content aspects, which attract more attention than the development in the student of the fundamental processes of doing mathematics.

However, it seems to me that exemplary teaching "packages" rather than general ideas on content will be needed both to convince and to enable (8). We have begun to make some progress beyond speculation. Computing options are popular in undergraduates courses in mathematics, at least in Britain, though they are rarely well integrated with the rest of the mathematics curriculum. It will be most interesting to see the results of the 20 experimental US college courses in discrete mathematics funded by the Sloan Foundation, particularly when some of them are developed and trialled by more representative teachers than the initial innovators. It is worth keeping in mind the typical text book for college calculus courses which stands as an exemplar and a warning of what lies in wait at the end of the road of routine development.

In other cases, there is such clear opportunity for the computer to play a role (an introductory course on differential equations is one obvious example) that it seems scandalous that courses have been taught without - until we appreciate the difficulties of curriculum change. There are many such developments of current courses to be pursued, and surely collaboration, or at least communication, could help.

At least as important as new content are the insights and opportunities that computers provide in helping us tackle more effectively some of the key problems in the mathematics curriculum; these are centred on mathematical processes, particularly related to the development of higher level skills. There is already some evidence that these possibilities are rich and various; it is equally clear that we are only at the beginning of discovering what they are.

Many of them need not have involved the computer. For example, it happens that mastery is often expected in programming (you go on until it works) but rarely in other parts of the mathematical curriculum. Yet the mastery of a technique is essential if it is to be used in problem solving, pure or applied. Similarly, debugging skills are recognised as an essential element of computing. Research suggests that they are equally important in the mastery of mathematical techniques - effective mathematicians "debug" their half-remembered algorithms. The "diagnostic teaching" approach is designed to build on this.

In looking for such opportunities, it may help perspective to take a brief look at the curriculum:

The Mathematics Curriculum consists of a pattern of **learning activities** in which the teacher helps the pupils to acquire **knowledge** and skills rooted in a developing structure of **concepts.** Such conceptual understanding is known to be important for the accurate recall of knowledge and retention of skills. There is a need for skills at various levels:

technical - carrying out well defined procedures
tactical - choosing them for use for a familiar purpose
strategic - choosing the line of attack on a problem
control - deploying general strategies independent of the particular problem

and all these must be backed by knowledge. To be useful in serious problem solving, the component skills need to be available at mastery level. Technical and tactical skills often relate to **particular mathematical topics, as do some strategies.**

The shortcomings are well recognised, as are the consequent opportunities for curriculum change. Only a small subset of the desirable learning activities happen in most classrooms. Curriculum targets are presently unrealistic for most students - extensive practice is used to close the gap, only temporarily because the conceptual structure is not robust and skills are not retained. Technical skills are emphasised at the expense of more strategic ones, and of applications.

Learning Activities If children are to become functionally effective at mathematics at any level, then need a wider range of learning activities than is commonly found. The Cockcroft Report (9) expresses it thus

> **243 Mathematics teaching at all levels should include opportunities for**
>
> * exposition by the teacher
> * discussion between teacher and pupils and between pupils themselves;
> * appropriate practical work;
> * consolidation and practice of fundamental skills and routines;
> * problem solving, including the application of mathematics to everyday situations;
> * investigational work.
>
> In setting out this list we are aware that we are not saying anything which has not already been said many times and over many years. The list which we have given has appeared, by implication if not explicitly, in official reports, DES publications, HMI discussion papers and the journals and publications of the professional mathematical associations. Yet we are aware that although there are some classrooms in which the teaching includes, as a matter of course, all the elements which we have listed, there are still many in which the mathematics teaching does not include even a majority of these elements."

The absence of the activities underlined (by us) is not surprising - they are **more demanding** on the teacher than the prevailing exposition plus exercise in three senses:

mathematically - because a variety of different tracks through the problem or argument must be handled,

pedagogically - because diagnosis and 'minimum correction' are needed, different for each child,

personally - because the teacher will at times be sailing unchartered waters, and will not know all the answers.

The microcomputer has been shown (10) to be a powerful support to teachers in **widening their style range** to support more open activities; the design of programs to this end is an important field. The teaching skills involved then seem to transfer, at least to some extent to other teaching. **It may well be at this stage, this is the most valuable single area for development** - it is of course, a form of INSET as well.

The background paper rightly emphasises the curriculum opportunities for exploration, for "experimental mathematics", that the computer provides. However, we have a lot of evidence and some understanding of how difficult such activities are for the teacher to handle in the classroom.

Exploratory investigation as a key element in the curriculum has been a major objective in English mathematical education for at least 30 years - the Association of Teachers of Mathematics was founded largely to promote it. Despite strenuous efforts it has not happened except in a tiny minority (much less than 1 percent) of classrooms. The Cockcroft report puts this delicately in the paragraph quoted above, but the evidence from systematic surveys is starkly clear. Though the computer can provide support to teachers in this regard, the development of an investigative element in the curriculum can succeed only if it confronts the difficulty such activities present, particularly for teachers.

Equally, the challenge to explore must be at a level matched to the student - if the aim is to "discover" in an hour or so some important mathematical achievement that took a genius half-a-life-time to create, the exploration will have to be so closely guided as to be essentially a fake; on the other hand, interesting, though less global problems do exist at every level which the student can tackle on his own resources. For example, programming projects, at school and university have shown the possibilities and the difficulties for the teacher; a creative and systematic program of detailed empirical development will be essential if exploration is not to degenerate in most classrooms into that closely guided "discovery learning", which is really an alternative style of explanation.

We already have evidence (11,12) that the potential of the microcomputer for helping teachers to enhance student learning presents a tremendous opportunity for curriculum enhancement. The effects on the dynamics of the classroom can be profound, but they are often subtle; for this reason there is a great deal still to do before we have even a broad understanding of what can happen in the various modes of computer use of the kind listed in the background paper.

I shall illustrate the sort of thing that may be expected by describing one application that has been developed and studied in some detail, and which has proved particularly rich - the use by the teacher of a single micro in the classroom programmed to be a "teaching assistant". I do so for various reasons - it is less familiar to most people, it brings out some general points about the overwhelming importance of the <u>people</u>, teacher and pupils, and of the dynamics of their interaction, and it is particularly relevant to schools as we know them because it seeks to enhance the performance of a teacher working with a group of children in the classroom in the normal way. It also only requires one micro computer per class rather than one per child.

This mode of use, first emphasised by Rosemary Fraser, has been shown to have remarkable effects in leading typical teachers in a quite unforced and natural way to broaden their teaching style to include the "open" elements that are essential for teaching problem solving. Since this is a crucial aim that we have been trying to achieve for at least thirty years with little or no effect, this is a valuable result. It is worth explaining briefly why these effects come about (10). First, the micro is viewed by the students as an independent "personality". It takes over for a time a substantial part of the teacher's normal "load" of **explaining, managing,** and task setting. These are key roles played by every mathematics teacher. The micro takes them over in such a way that the teacher is led into less directive roles, including crucial discussion with the children on how they are tackling the problem, providing guidance only of a general strategic kind - counselling if you like.

It is equally important to recognise that there will be disappointments - or at least frustrations.

Apart from programming itself, perhaps the first big idea for using computers in mathematical education was in teaching technical skills, particularly arithmetic. The approach followed the behaviourist teaching machine model. This has proved a much harder problem than was expected. It is still unsolved. It seems that the computer can be effective in teaching facts and straightforward techniques to people who have little difficulty with them; so, of course, are other methods. However, despite great efforts by some extremely talented people, it has not so far proved possible to write programs which are successful in diagnosing and remediating students' errors in technical skills that they find difficult.

In other cases, the size of the potential "target group" is unclear. The activities of that small proportion of enthusiastic "computer nuts" display a motivation and the deployment of a range of strategic and technical skills that are rarely matched in the normal curriculum. (Could we ever visualise a mathematical "hacker" causing ingenious chaos in the school or college mathematics department?) How far can such rich learning activities be stimulated and exploited in all children? We do not know, but the proportion of children who are spontaneously using their home computers in this sort of way does not seem to be large. Again experiment is needed; we are hoping to set up such a study of "100-micro schools".

In tertiary education, it is common for the teacher to play a much narrower range of roles - explaining and task setting, with little else. The enrichment of the range of learning activities through the alteration of the classroom dynamics which the computer makes possible may not be welcome here; it makes a much more serious departure from standard lecture format than in schools (some at least) and, again, will certainly slow down the rush from one topic to the next which ensures a syllabus content of "high standard", whatever the level of independent student performance. This is particularly serious in advanced undergraduate pure mathematics courses, where enough 1 hour advanced problems often seem hard to find.

The questions I have raised require a great deal of work, integrating research techniques with curriculum development, before we have even a basic understanding of the classroom potential that we see. Experience suggests we shall find other possibilities of at least as much promise.

In order to realise the potential of any of these possibilities they will need to be systematically developed in detail with representative samples of teachers and students, using structured detailed data from the classroom.

12 Hugh Burkhardt "The Microcomputer: Miracle or Menace in Mathematical Education" in Proceedings of ICME5, Ed Marjorie Carss (Birkhauser, 1985).

REFERENCES

1 The Influence of Computers and Informatics on Mathematics and its Teaching. ICMI, L'Enseignement Mathematique 30, 159 (1984).

2 Andre Weyl, in Proceedings of ICM, Helsinki 1978.

3 Adele Goldberg, "Informatics and Mathematics in the Secondary School", 1977 IFIP Conference, Ed D Johnson and D Tinsley (North Holland, 1978).

4 See, for example "How might we move the curriculum" Hugh Burkhardt, 1982 BSPLM Oxford Conference (Shell Centre 1982).

5 Design and Development of Programs as Teaching Material, Hugh Burkhardt and Rosemary Fraser et al (Council for Educational Technology, 1982).

6 See, for example, "Aspects of Secondary Education", Report of the HMI Secondary Survey, (HMSO, 1977).

7 Vern Treilibs, Hugh Burkhardt and Brian Low, "Formulation Processes in Mathematical Modelling (Shell Centre, 1981).

8 See, for example, "Problems with Patterns and Numbers", a module of the Testing Strategic Skills programme (Joint Matriculation Board, Manchester M16 6EU, 1984).

9 Mathematics Counts, Report of the Cockcroft Committee of Inquiry (HMSO, 1982).

10 Rosemary Fraser, Hugh Burkhardt, Jon Coupland, David Pimm, Richard Phillips and Jim Ridgway. "Learning Activities and Classroom Roles" (Shell Centre, 1983).

11 Hugh Burkhardt "How Can Micros Help in Schools: Research Evidence" (Shell Centre, 1984).

USING COMPUTERS WITH UNDERGRADUATE MATHEMATICS STUDENTS IN COLLEGE ALGEBRA, ELEMENTARY CALCULUS, AND TEACHER-TRAINING COURSES

M.J.Winter
Michigan State University
East Lansing, Michigan

As the title of this paper indicates, my work has been with three almost disjoint groups of students. These students represent increasing levels of mathematical ability and sophistication. The different levels of these students require very different ways of using computers in their instruction.

College Algebra

The first group of students is the largest: those who are not going into science or mathematics, but who must take two or three terms of non-calculus, college mathematics. usually college algebra, trigonometry, and/or "finite mathematics." These students are required to study mathematics both to acquire skills necessary to their academic programs, and to develop some form of problem-solving ability or mathematical thought.

This group is not shrinking, nor are its members arriving better prepared. Their approach to mathematics has been (and continues to be) atomic--each section or topic is unrelated to any other. These students, especially, lack the geometric and algebraic background that was common to students of fifteen years ago.

It has become more and more difficult to give these students the kind of "good algebra course" that their departments assume will raise their mental processes as well as provide them with skills. A "good algebra course", most faculty agree, is one for which students work lots of word problems. "The more problems you work, the better you'll work them" summarizes the conventional wisdom. Students who are actively working problems will find their thinking changed.

This active involvement of the learner must be maintained and increased as we modify the presentation and content of college algebra. Graphical demonstrations will provide a base on which we can build. There also must be interaction between the students and the mathematics for the base to be solid.

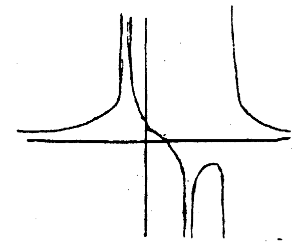
At the beginning of a college algebra course, I believe it to be inappropriate for students to use a symbolic language such as muMath. Learning manipulations and concepts should preceed learning such a language. However, arithmetic calculation and graphing are already within students' experiences; computers can be used to perform these operations. The numbers in fundamental examples and problems students work by hand, should be kept simple, so as not to distract. Once a student can graph

$$f(x) = \frac{(x-1)}{(x-2)*(x-3)^2}$$

by hand, then s/he can use a computer to graph

$$f(x) = \frac{2^x - 1}{x(\cos x - 1)}$$

The student can be assigned the task of finding a function whose graph looks like this:



The experimentation required to produce such a graph deepens a student's understanding of rational functions.

Some other topics for which computers could be successfully used to introduce concepts and build a background are:

Composite Functions The idea that f, g and fog are all functions is easily conveyed by showing their graphs. The fact that fog and gof are different functions is better taught by comparing their graphs, than by calculating a single point at which they differ.

Shifts and Scale Changes

Compare the graph of the equation $y = f(x)$ with the graphs resulting when x is replaced by ax, by x-b, by ax-b. Replace y by cy, y-d, cy-d. Software which allows the user to specify f either by an equation, by a set of equations, or by its graph (using a joystick) would be especially useful. Repeated examples can make the point that lines go into lines. After students have seen the nature of the transformed graphs, they can be asked to predict (by hand) the results of linear transformations. They can be asked to find the transformation that effected a given change of graphs.

A graphing program which allowed one to change from regular axes to semi-log and log log scales would be another useful tool for learning about the behavior of functions the students will see in their later coursework.

Sequences Recursive sequences, especially, appear in many other fields. With computers available our students can study these sequences as well as sequences arising from population dynamics and other applications. Graphically, they can learn about stability, oscillation, and carrying capacity. They can study the effects of varying parameters.

Although students learning about matrices should multiply them by hand, students working problems involving matrix operations should be able to have products, inverses, and calculations for the simplex algorithm performed mechanically. Languages which produce the solution sets to dependent systems should also be introduced early. (once the concepts have been mastered). The amount of time spent on tedious and error-prone calculations, far exceeds what is involved in learning how to enter a system and understand the output. With the help of such languages, students can be required to completely solve problems, not just set them up.

These are some of the ways we can better teach college algebra, and teach a better college algebra course. Students will understand and be able to use more of what we teach them. The word problems must remain, however. They should not be made to match a computer template. Setting up these problems is, and will remain, as essential part of achieving mathematical maturity.

Beginning Calculus

The second group are the beginning calculus students. I'll mention briefly an innovation that has been tried at MSU. Early in the first course we have taken a day to define a "slope function" for a given function f(x). The first two points of the slope function were found using a calculator, and plottted by hand; a computer program was used to complete the calculations and draw the graph. (Figure 2). The students then used the computer to graph the slope functions for different f(x); they copied the graphs, made predictions, and answered questions. This hour appeared to provide a better understanding of derivative when it was formally introduced. It had another benefit: one of the problems was to observe that the slope function of 2^x was similar to the function's graph, but below it; the slope function of 3^x was similar to 3^x but above it. A search for b such that b^x was its own slope function led to a "tangible" defintion of e.

An experiment we intend to try is to introduce quadrature at the beginning of the course.(Figure 3). After computing areas under a curve using the trapezoidal rule n times, the power of the Fundamental Theorem might be more meaningful.

Prospective Secondary Teachers

The final group of students I would like to consider are the prospective teachers of secondary mathematics.

One of the strongest reasons for using computers in school mathematics is that they will enable students to experiment and create mathematics. If this is to be effectively done, teachers must be trained to guide exploration. They must be able to pose a problem specific enough for all their students to make some headway, yet open-ended enough to be challenging.

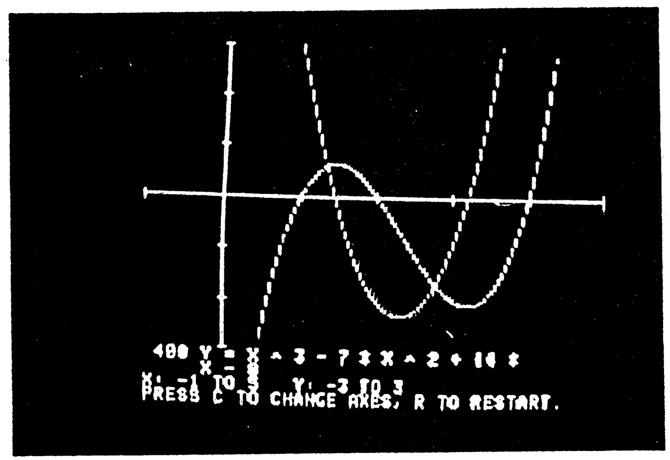
To pose problems, they must possess a larger view of mathematics than that given by the traditional college courses. (Pythagorean triplets, for example, are not just sides of triangles: they are generated by rational solutions of certain polymonials, they have a close relationship with triples from 60' and 120' triangles.) Because their students will do more and different math than before, teachers will need a broader background.

At MSU we have instituted a mathematics course, "Using Microcomputers in Teaching and Learning Mathematics". This course is, among other things, a vehicle to teach topics such as difference equations, continued fractions, variation of parameters, Monte Carlo techniques, expected value, predator-prey equations, number-theory in different bases, and so on. The course itself is taught in an explore-conjecture format. Between class sessions, students spend time in the micro-lab, generating data, writing programs to solve problems, as well as trying to organize their results and draw conclusions.(A two-week four-hour a day version of the course has been successfully used with experienced teachers.)

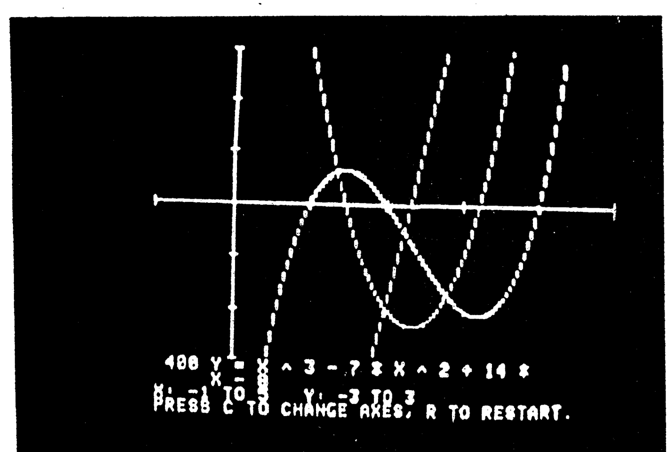
Teachers will also need to know what difficulties students will encounter, for example, why SQR(36)-6 <>0, why 3 raised to the 4th power may not be 81, why the roots of (x-1)*(x-2)*...*(x-10) are impossible to find exactly. They need to be ready to suggest alternate techniques. They need some sort of numerical analysis course.

The teaching of geometry, in particular, will change as software to manipulate geometric figures becomes more common. In the Geometric Supposer, for example, students can make any ruler and compass construction; the construction can be repeated on any triangle. Similar software for transformational geometry cannot be long in coming. Teachers will need to know much more geometry in order to guide explorations.

Finally, they must be prepared to prove conjectures, and to teach students to distinguish between proof and verification.



SLOPE FUNCTION
$y = x^3 - 7x^2 + 14x - 8$



AREA UNDER SINE CURVE
FROM 0 TO π

USING TRAPEZOIDAL RULE

1	1.14897317E-09
2	1.57079633
3	1.81379936
4	1.8961189
5	1.9337656
6	1.95409723
7	1.96631668
8	1.9742316
9	1.97965081
10	1.98352354

AREA UNDER Y = X*X
FROM 0 TO 1

USING TRAPEZOIDAL RULE

1	.5
2	.375
3	.351851852
4	.34375
5	.34
6	.337962963
7	.336734694
8	.3359375
9	.335390947
10	.335

AREA UNDER Y = X*X*X
FRROM 0 TO 1

USING TRAPEZOIDAL RULE

1	.5
2	.3125
3	.277777778
4	.265625
5	.26
6	.256944444
7	.255102041
8	.25390625
9	.25308642
10	.2525

Figure 3

MICROCOMPUTER PROGRAMMING FOR HIGH-ABILITY MATHEMATICS STUDENTS

Clark Kimberling, University of Evansville

Abstract: This article refers to the author's ongoing series in The Mathematics Teacher as a focus for two issues: program-writing needs of high-ability students and an extracurricular approach to meeting those needs.

One of the concluding paragraphs in Lynn Arthur Steen's excellent article, "Living With a New Mathematics Species," submitted to this Symposium, is the following:

> Mathematics is, after all, the science of order and pattern, not just a mechanism for grinding out formulas. Students discovering mathematics gain insight into the discovery of pattern, and slowly build confidence in their own ability to understand mathematics. Formerly, only students of sufficient genius to forge ahead on their own could have the experience of discovery. Now with computers as an aid, the majority of students can experience for themselves the joy of discovery. Only when the computer is used as an instrument of discovery will it truly aid the learning of mathematics.

What practical steps can teachers take to promote students' discovery-learning of mathematics? This question was in the minds of the editorial panel of The Mathematics Teacher when they decided to launch a new section called "Microcomputer-assisted Discoveries."

Certainly, one step that teachers can take is to collect discovery-oriented programs for students to use. In addition to that, teachers can collect ideas for programs for students to write. Accordingly, a main purpose of the Microcomputer-assisted Discoveries section (renamed Microcomputer-assisted Mathematics with the January 1985 issue) is to provide teachers with an ongoing collection of such programs.

The Microcomputer-assisted Mathematics section of The Mathematics Teacher is open to anyone who wishes to contribute. Prospective contributors may consult a recent issue for information regarding manuscript preparation and submission.

Following is a list of all the titles of articles that have appeared in the section, one each month (September through May), beginning in September 1983:

1. Primes
2. Euclidean Algorithm and Continued Fractions
3. Number Bases
4. Random Numbers
5. Probability Machine
6. Generate Your Own Random Numbers
7. Calculating Palindromic Sums by Computer
8. Conics
9. Lines
10. Central Limit Theorem
11. Mean, Standard Deviation, and Stopping the Stars
12. Circles and Star Polygons
13. Roots: Half-Interval Search
14. Graph Many Functions, Part 1
15. Graph Many Functions, Part 2
16. Graph Many Functions, Part 3

Titles to be published during 1985-1987 include Integrals, Powers of Complex Numbers, Newton's Method, From Simple Interest to e, Quadratic, Lagrange Polynomials, Permutations and Combinations, Factoring and Unfactoring, and Complex Roots: the Bairstow-Hitchcock Method.

Each of these articles contains listings of programs and suggestions for student-writing of similar programs or student use of the programs as given. However, the articles do not address two important issues, and it is the purpose of the remainder of this article to do so.

ISSUE 1: Program-writing by high-ability students

That there exists a wide range of abilities among students is a fact that requires no discussion. However, the needs of abler students are wastefully neglected in settings where the level of instruction and computer use are determined by other needs. The authors of An Agenda for Action (NCTM, 1980) recognize this problem pointedly:

> The student most neglected, in terms of realizing full potential, is the gifted student of mathematics. Outstanding mathematical ability is a precious societal resource, sorely needed to maintain leadership in a technological world.

Many gifted students are able BASIC language programmers before the age of fifteen. After that, they take mathematics courses. In all of these, they should be writing original programs that illustrate, extend, and "prove" various topics in those courses.

Unfortunately, in many cases, these students are not expected to write such programs until they enter college. Even then, those students who are in the top 20% in mathematical ability should be writing more programs in their mathematics courses than are presently expected.

Why write original programs? We have already mentioned their value in discovery-learning, but let us also recognize another particularly important and timely benefit to the student. That is, the development of problem-solving skills. As many writers have been pointing out in recent materials, writing a program is definitely problem-solving.

Norris (1981), for example, discusses program-writing as problem-solving in support of a proposed "drastic restructuring of the traditional high school curriculum" that would "delete plane geometry as a required course in the traditional academic sequence and replace it with a year-long course in computer-programming."

Some writers seem to assume that whatever they recommend must involve changing a curriculum. It has become clear, however, that curricula are slow to change. Moreover, the writing of original algorithmic programs is best suited to high-ability students. In schools where appropriate tracking for these students is not available, there exists a choice between neglect of the problem, on one hand, and extracurricular activities, on the other.

ISSUE 2: Extracurricular program-writing

Actually, extracurricular activities are highly appropriate, inasmuch as the needs of the gifted are often individualistic and characterized by a pronounced desire to do things in one's own way.

It may be helpful to consider the development of a specific topic that could best be pursued by a high-ability student working with minimal guidance (supplied mostly by printed material instead of the teacher, who already has more than enough to do within the curriculum).

Suppose the student is setting out to program the Euclidean algorithm. Perhaps (s)he is taking a course, and the textbook has the usual page-or-so treatment of this algorithm. The teacher has hinted that it would be nice to have a program that would perform this algorithm, so that members of the class could experiment with it. Our hypothetical student has taken the teacher's hint as a sufficient reason to try to translate the textbook treatment into a BASIC program. After awhile, the student has made a translation comparable to this:

```
10  INPUT "INPUT A,B = "; A,B
20  R = A - B*INT(A/B
30  PRINT A,B,R
40  IF R = 0 THEN 10
50  A = B
60  B = R
70  GOTO 20
```

Having just written the program, this student understands not only how the algorithm works, but also, the role played by each of the symbols used. (S)he is therefore better prepared to use this program for experimentation than if the program had been "canned".

Suppose this student's gift includes curiosity. (Einstein and others have written that curiosity is the main urge underlying scientific research.) In an extracurricular environment, our hypothetical student is freer to formulate questions and to follow up on them with program-modifications, pattern-searching, trial-and-error, and other techniques of discovery-learning and the scientific method.

After trying out several choices of A and B, our student may begin to wonder what choices require the greatest number of iterations before the algorithm terminates. Further experimentation will lead to questions like this: "Given N = 20, or 21 or 22, or some other specific number, what choices of A and B, with 1 < A < B < N, require the most iterations?" This line of questioning could easily lead to a discovery of the Fibonacci numbers. Such a discovery, on one's own, often engenders a productive attitude about a particular topic and about the environment in which the discovery was made. Accordingly, the student is likely to be receptive to further study and discoveries. The Fibonacci numbers, for example, lead naturally to continued fractions, difference sequences, and topics in combinatorics and botany.

Heid (1983) wrote, in regard to curricula:

> Assuming the existence of an appropriate curriculum, the gifted will still need to be given the time and guidance to pursue chosen mathematical topics independently and in depth. When independent study is pursued, teachers will need to offer adequate encouragement, expertise, and the judicious and selective application of a policy of noninterference. The gifted student who is so tenderly treated stands the change to grow both in depth of knowledge and in ability to work with freedom from supervision.

In reality, there exists relatively little "tender treatment of the gifted." Furthermore, "freedom from supervision" - a _sine qua non_ for many independent minded students - points directly toward EXTRAcurricular program-writing. By EXTRAcurricular is meant something on the order of library books, along with the kinds of incentive that have for decades led students to check out library books without necessarily being assigned to do so. To support program-writing, there should be a check-out system, not only for books, but also for software, programming manuals, problem collections, contest announcements, and, of course, access to computers with printers.

Here are five steps teachers can take to promote extracurricular mathematics-related program writing:

(1) Maintain a "software bank" of the sort described in the ICMI (1984) discussion document, "The Influence of Computers and Informatics on Mathematics and Its Teaching."

(2) Provide incentives. Elevate the status of program-writing within student peer groups. Post printed results of student-written programs. Enable students to save their best programs in a mathematical software library that is open to other students. Sponsor contests and get some publicity in local media. The main thing is to see that students who deserve recognition get it.

(3) Within mathematics courses, let students earn "extra credit" by writing relevant programs. Encourage program-writing as an essential part of projects for science fairs and comparable events.

(4) Help recruit students for summer computer programs, camps, and university offerings for gifted students.

(5) Encourage mathematical program writing as part of the activity of the school computer club.

A final thought concerning the notion of extracurricular program-writing is that such activity stands to help, not hinder, efforts to get more program-writing into mathematics curricula. Our main attention in this article, however, is on immediate needs that simply will not be met for many students now in school, if not through extracurricular means.

References

Heid, M. Kathleen, "Characteristics and Special Needs of the Gifted Student in Mathematics," _Mathematics Teacher_ 76 (1983) 221-226.

International Commission on Mathematical Instruction (ICMI), "The Influence of Computers and Informatics on Mathematics and Its Teaching," _L'Enseignement Mathematique_ 30 (1984) 161-172.

National Council of Teachers of Mathematics, "An Agenda for Action: Recommendations for School Mathematics in the 1980s," Reston, Virginia, The Council, 1980.

Norris, Donald O., "Let's Put Computers into the Mathematics Curriculum," _Mathematics Teacher_ 74 (1981) 24-26.

Learning Geometry with the Assistance of Logic Programming

Richard ALLEN
Pierrick NICOLAS
Laurent TRILLING

IRISA
Campus de Beaulieu
35042 RENNES CEDEX
FRANCE

Abstract. In the learning and teaching of euclidean geometry at the French collège or U.S. high school level, the two most significant learning hurdles facing the student, and, consequently, the two major pedagogical tasks facing the teacher, are: (1) the accurate construction of a geometric figure using the equivalent of pencil, ruler, and compass, and (2) the construction of an accurate proof of a theorem in geometry. A CAI system is being developed at the Institut de Recherche en Informatique et Systèmes Aléatoires (IRISA) in Rennes, France, which will direct and correct student attempts (1) to construct accurate geometric figures representing the hypotheses of a theorem from plane euclidean geometry, and (2) to establish an accurate proof of the theorem. The system allows both graphical and vocal input. Logic programming provides the theoretical base for the system, with Prolog being the implementation language.

Logic Programming

Logic programming can be viewed as a computational model of first order predicate calculus. This view can be made more precise by recalling that any first order predicate calculus formula can be written in clausal form; i.e., as a collection of $A_0, A_1, \ldots, A_n$ of (implicitly) conjoined clauses where each clause A_i is a collection $B_0, B_1, \ldots, B_m$ of (implicitly) disjoined literals. A literal is either an atomic formula or a negated atomic formula.

The following conventions can be used to express a formula in clausal form. Simply write the clauses down one after the other. Within any given clause, there is a collection of unnegated literals $B, C_1, \ldots$ and a collection of negated literals $K, L, \ldots$. More specifically, a given clause would look like:

$$(B \vee C \vee \ldots) \vee (-K \vee -L \vee \ldots)$$

which is equivalent to;

$$(B \vee C \vee \ldots) \vee -(K \wedge L \wedge \ldots)$$

which is equivalent to;

$$(K \wedge L \wedge \ldots) \rightarrow (B \vee C \vee \ldots)$$

A Horn clause is a clause with at most one unnegated literal; i.e., all formulas of first order predicate calculus of the form:

$$Q \wedge R \wedge \ldots \wedge S \rightarrow P$$

Richard ALLEN is on sabbatical leave during 1984-85 from St. Olaf College, Northfield, MN 55057 USA.

where the $P, Q, R, \ldots, S$ are literals and where true $\rightarrow$ P if there are no negated literals.

QUESTION: Can all theorems in plane euclidean geometry be represented as Horn clauses?

Suppose you have a (mathematical) system where all definitions, axioms, and theorems are represented as Horn clauses. How do you prove (new) theorems? The answer is by using the resolution principle discovered by Robinson; i.e., given two Horn clauses,

$$\text{(a) } Q \wedge \ldots \wedge R \wedge \ldots \wedge S \rightarrow P$$

$$\text{(b) } R_1 \wedge \ldots \wedge R_m \rightarrow R$$

and, given necessary substitutions for any variables, the Horn clause,

$$\text{(c) } Q \wedge \ldots \wedge R_1 \wedge \ldots \wedge R_m \wedge \ldots \wedge S \rightarrow P$$

is a valid consequence of the previous two. Obviously, this principle has the appearance of a rule for rewriting Horn clauses which, if carried out systematically, might be a mechanical process executable by a (computing) machine. From a practical standpoint, if there are several Horn clauses with R on the right-hand side, there are several possible rewrites or resolutions for (a), with (c) being just one example. We can apply resolution steps such as the one described to see whether the theorem we want to prove eventually appears. However, theorem proving with resolution is usually approached in a slightly different manner.

Resolution possesses the property of being refutation complete; i.e., if a set of clauses is inconsistent, then resolution can derive from them the empty clause. Consequently, if we want to show P is a consequence of the clauses $Q, \ldots, R$, we use resolution to derive the empty clause from $Q, \ldots, R, -P$.

Prolog is a programming language that is based on a resolution theorem prover for Horn clauses. A Prolog program is a collection of (Horn) clauses; each clause, in the syntax of Prolog, takes one of the following forms:

(1)	P( ) $\rightarrow$;	fact
(2)	P( ) $\rightarrow$ Q( ) ... R( );	rule
(3)	P( );	question

The P( ),Q( ),R( ),... represent predicates and the parentheses indicate that the predicates might have one or more arguments. Specifically, the arguments may be constants, variables, or functional expressions. In particular, the formula (2) can be read and understood to mean that the left side is implied by the right side.

Example 1. Three clauses are given. One defines the fact that john is the father of mary . The other two clauses define a relationship that can be used to check whether a person is an ancester of another.

```
father(john,mary) → ;
ancester(X,Y) → father(X,Y);
ancester(X,Y) → father(X,Z)ancester(Z,Y);
```

Example 2. Consider the Prolog functional expression for a list s defined by

$$s = \begin{cases} \text{nil} \\ \text{e.sl} \end{cases}$$

where e is a constant, variable, or list and sl is a list. Define the Prolog clauses for concatenating two lists as follows:

```
concat(nil,y,y) → ;
concat(e.x,y,e.z) → concat(x,y,z);
```

Example 3. Tower of Hanoi:

```
move(1,I,J) → write(I)write('tc')write(J)newline;
move(N,I,J) → assign(sub(N,1),K)
              assign(sub(sub(6,I),J),Other)
              move(K,I,Other)
              move(1,I,J)
              move(K,Other,J);
```

If the posts are numbered 1, 2, and 3, then the Other post in the second clause above is obtained from the formula 6-I-J .

Project Design and Implementation

This project began with two secondary mathematics teachers* who were participating in a program of the Centre de Recherche et de Formation de Formateurs en Informatique de Bretagne (C.R.E.F.F.I.B.) at the University of Rennes I in Rennes, France, during the academic year 1983-84. The program allows teachers to come to the Centre for one year to study computer science with the aim of returning to their schools to help integrate computers into the teaching at the institution. A requirement of the participants is the design and implementation of a computer project related to teaching. Le Nestour and Rouxel chose as their project focus problems students encounter as they attempt to construct an accurate proof of a theorem in plane euclidean geometry.

*Mme Anne Le Nestour of Brest
M. René Rouxel of Equeurdreville (Cherbourg)

Teachers of geometry are often faced with students who themselves do not seem to be able to find a correct sequence of logical steps which leads from the hypotheses of a theorem to its conclusion. Yet these same students do understand the logical development of a proof of the same theorem if the teacher or, perhaps, a fellow student, shows it to them. The reaction is often, "Yes, I understand now, but how did you know where to begin and what theorem to use to go from such and such a step in the proof to the next step?" Often a student having such difficulty can be helped in the following way: begin by giving the student the statement of a theorem to prove. Help the student construct, using pencil, ruler, and compass, a geometric figure which clearly demonstrates, first, the hypotheses of the theorem and then its conclusion. Next, provide the student with a very limited number of definitions, axioms, and theorems whose timely application will produce a proof. The idea here, of course, is for the student, at any stage in a proof, to be able to resolve logically that stage to another stage using a limited amount of pertinent geometric information. The teacher thus hopes not to overwhelm the student, to help him avoid the feeling of hopelessness, and yet to guide his search for a proof.

Le Nestour and Rouxel designed and implemented a CAI system in Micro-Prolog (a dialect of Prolog) which allows a teacher (1) to enter a theorem to be proved; (2) to enter any definitions, axioms, and implicit information deemed necessary for a student to carry out the proof; (3) to specify those theorems (usually a limited number) that a student may use to establish an accurate proof of the given theorem. The system proceeds in two phases. In the first phase, the student must exhibit understanding of the statement of the theorem by correctly identifying and entering all hypotheses and the conclusion. In the second phase, the student proceeds to construct a proof in a reverse fashion, proceeding from the conclusion of the theorem to be proved to its hypotheses by forming a series of intermediate steps any one of which follows from the immediately succeeding one as a consequence of one of the theorems explicitly provided in advance by the teacher. Appendix C contains a list of three theorems that a teacher might provide the student to construct a proof of the theorem found in Appendix B. The sequence of steps terminates when the student reaches a step that is immediately implied by the hypotheses of the theorem.

The system is constructed to ask the student to enter in Prolog syntax what is understood to be the hypotheses and conclusion. The system then verifies that the user proposed hypotheses and conclusion are valid. Next, the student begins to organize a proof, starting from the conclusion and working toward the hypotheses. In going from one intermediate step in the proposed proof to the next, the student each time in effect introduces a new problem to resolve. The system verifies that the new problem does in fact imply the previous step through application of the student indicated theorem. The system also demands that the student understand the new problem by indicating any new hypotheses or conclusion that may have been introduced in this step.

The Prolog language was both a positive and a negative factor at this point in the project's history. Prolog enabled direct paraphrase of the statements of theorems in geometry and straightforward implementation of logical inference. A negative factor of significance was that both student and teacher had to communicate with the system in the syntax of Prolog. The teacher had to enter all geometrical information and statements in the Prolog syntax. (Appendix D contains the actual Prolog clauses used by the teacher to enter the statement of the hypotheses and conclusion of the theorem of Appendix B.) The student, as well, had to formulate and enter in the Prolog syntax hypotheses, conclusion, and any other geometrical information demanded during the proof phase.

During the summer of 1984, the software was translated into Prolog II and the hypothesis/conclusion verification part of the system was enhanced with the addition of graphics to enter geometric objects describing the hypotheses and conclusion.* In particular, a geometric construction language (in the form of commands - see Appendix E) was implemented that allowed the user to construct and name objects such as points, line segments, perpendicular lines, etc. The system displayed graphically on a screen the object being constructed as well as adding to its internal data base Prolog predicates (see Appendix F) describing the geometric information. After completion of the figure, the system verified the hypotheses and conclusion using, in part, the information added to the data base.

The graphics input was an obvious pedagogical advantage. Students enjoy drawing things with computers, and drawing these figures eases them into thinking about the proof. Although the particular graphics package used was directly executable from Prolog II, it executed slowly. The graphics input is a more natural way for a mathematics student to communicate geometric information; it also removes the necessity of the student's employing the Prolog syntax in the verification phase of the system. However, the teacher still had to employ Prolog syntax to communicate with the system and the student still had to continue its use in the demonstration phase. (Appendix G provides a sample of student communication with the system during part of the demonstration phase.)

The preceding constitutes a beginning from which the present project has grown. The division into two parts remains: a figure construction phase whose purpose is to verify the hypotheses and conclusion, and a theorem demonstration phase whose purpose is to help the student construct an accurate proof.

The construction phase has been expanded to allow input from a graphics tablet, voice input, and screen display of constructed figures. A graphics editor that will allow communication in a language close to the traditional one of plane euclidean geometry and that will allow the teacher to build for student use complex geometric construction tools out of simpler ones, is being implemented.

Currently, a natural language interface is being planned that will allow the teacher to enter geometric information, including all theorems, without requiring the adoption of any additional syntax, such as that of Prolog. A similar interface will be available for the student to use during the theorem demonstration phase. At such time, all communication will be carried out in a manner similar to that employed in the classroom teaching of geometry; i.e., a combination of geometric constructions and statements of geometry in their traditional format, the notations used in the texts of the French school system. (Coincidentally, natural language understanding is a main application area for Prolog.)

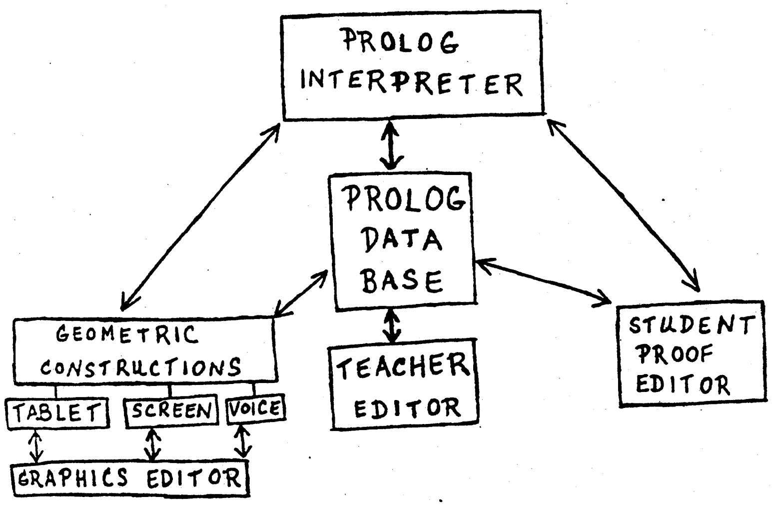
The demonstration phase of the system is being redesigned to allow the student to proceed 'backwards' from conclusion to hypotheses or 'forwards' from hypotheses to conclusion or in a manner that combines both 'directions'. The system will thereby more effectively model the way in which students actually undertake working out proofs. This design, it should be noted, represents a significant break with traditional approaches to CAI. For, whereas traditional approaches normally require the teacher to provide either one 'solution' and/or to anticipate all possible steps the student might take to find it, here instead the teacher is provided with a tool which allows him to specify only the logic of the problem: he need not provide a 'solution' nor anticipate every different valid and invalid attempt to prove a theorem.

Internally, the system is organized as follows. All information exists in the form of Prolog clauses. Any geometric data entered by the teacher is translated into such clauses and these clauses are added to the Prolog data base. The geometric constructions of the students are achieved by graphics procedures and the underlying logical information is extracted, translated into Prolog clauses, and added to the data base of Prolog clauses. As the student constructs a proof, all responses are translated into Prolog clauses. It is the Prolog interpreter, a resolution based theorem prover, that

---

*Realized by Jean-Come Estienney

actually verifies the logical inferences when needed. The interfaces to the graphics tablet and the vocal input card are written in Pascal but are executed from Prolog clauses. The entire system is Prolog based, with occasional calls to nonProlog procedures from certain Prolog clauses. In other words, the unifying intellectual unit is a collection of Horn clauses paraphrased in Prolog syntax.



System Organization

The Future

The present system is seen as a nucleus of an interactive geometry workstation where teachers specify problems on which students work with the system's guidance and aid when necessary. However, from the present starting point, the system could be adapted to the study of nonclassical geometry using nonfinite figures, such as spirals. Another direction that might be pursued is the addition of a mouse and the development of a system for learning Prolog concepts similar to the way the LOGO turtle aids in understanding the concepts of LISP. A third option is the development of a system to be used for observation and experimentation in the framework of deductive learning.

Pedagogical Relevance of Project

The principal human interfaces to the system are the three editors: (1) for the construction of the figures by the student; (2) for the teacher to enter geometric data; and (3) for the student to construct a proof. All three of these are being designed in consultation with the group of secondary mathematics teachers working at IREM in Rennes this academic year on the problems of teaching plane euclidean geometry. The purpose is to provide tools and accesses to them that employ the same mathematical language that is used in the classroom and in textbooks. Obviously, the system will provide help to students on an individual basis. In fact, it is intended to be most helpful to the student who is neither exceptionally bright nor exceptionnaly slow. Computer based systems themselves seem to predispose some students to learning who would not otherwise be interested. Perhaps the intrigue of discovering how to use this system might be the catalyst for learning a little geometry! In a similar way, this system may help a teacher feel a little less intimidated by the computer and by CAI, since he himself must use the system to create the lesson that his students will use. Clearly, then, it is the teacher who maintains control; the system is in no way a competitor to his own expertise.

Traditional approaches to CAI have a great deal of difficulty finding suitable representations for their knowledge bases. The design adopted in this system almost completely resolves the knowledge representation issue. The real problems reside with the design and implementation of the interfaces for acquiring and displaying the knowledge. It is here that the pedagogy of mathematics must play an important role. For example, should the student be given a formal language, a natural language, or a combination of the two, to organize a proof?

Relationship of Project to ICMI Conference Concerns

This project responds to a specific need expressed by a specific group of teachers: writing proofs of theorems from euclidean geometry at the level of quatrième in the French school system. Specific texts used and present classroom teachers were used to determine the curriculum to be taught and the constraints of actual classroom practice needing to be satisfied.

The computer science base of the project is logic programming. It is said that this programming methodology will have an increasing influence on many future computing developments. If this be the case, it may be a fertile place to look for applications to the teaching of mathematics and to the preparation of teachers. The present project grew out of a type of teacher training for mathematics teachers in computer science (where teachers were released for one year) and can be seen as an example of a convenient bridge between two very traditional subjects studied by prospective teachers of mathematics, geometry and predicate calculus. Yet the context is different from traditional mathematics and traditional methods for teaching, and provides a new tool to aid in teaching.

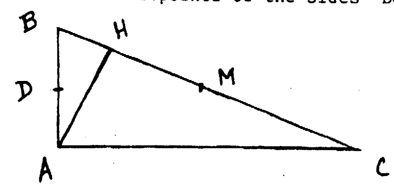
In summary, the project incorporates new technology, the advice of practicing teachers, and some sophisticated mathematics to create a tool designed to help school students master a central component of the traditional mathematics curriculum.

Appendix A

maternelle		2-5 years old
élémentaire	- cours préparatoire - cours élémentaire première année - cours élémentaire deuxième année - cours moyen première année - cours moyen deuxième année	6-10 years old
collège	- sixième	11 years old
	- cinquième	12 years old
	- quatrième	13 years old
	- troisième	14 years old
lycée	- seconde	15 years old
	- première	16 years old
	- terminale	17 years old

Appendix B: Example

Theorem. Let ABC be a right triangle with right angle at A. Let M and D be the midpoints of the sides BC and AB, respectively. Let AH be the height from A to the side BC. Show that the points D and H belong to the circle with diameter AM.



Appendix C: List of Theorems

Theorem 1. If MPN is a right triangle with right angle at P, then P belongs to the circle with diameter MN.

Theorem 2. If two lines are parallel, then any line perpendicular to one of them is perpendicular to the other one.

Theorem 3. In any triangle the line passing through the midpoints of two sides of the triangle is parallel to the third side.

Appendix D

Hypotheses: They are entered by the teacher as a Prolog clause of the form:

ho → list_of_predicates;

Obviously, the predicates must be ones that can describe logically the hypotheses of the theorem. The following predicates are defined for this example:

droit(x,y,z) : the angle xyz is right at y ;

milieu(x,y,z) : x is the midpoint of the segment yz ;

cercle(c,o,x) : c is the circle with center o and passing through the point x ;

apc(c,p) : the point p belongs to the circle c .

Recall that the hypotheses of the theorem in Appendix B are:

M is the midpoint of the hypotenuse of the right triangle ABC where the right angle is at A and AH is the height from A to the hypotenuse.

D is the midpoint of the side AB .

The teacher would then enter the hypotheses as the Prolog clause:

ho → droit(B,A,C)droit(A,H,C)droit(A,H,B)milieu(M,B,C)milieu(D,A,B);

Conclusion: Again, the teacher must enter it as a Prolog clause of the form:

conclgen → list_of_predicates;

The predicates in this list must encapsulate the properties contained in the conclusion of the theorem. The teacher can then enter the conclusion as:

conclgen → apc(c,A)apc(c,M)apc(c,D)apc(c,H);

However, the circle c is not defined in the hypotheses nor is it clear that AM is a diameter. This is an implicit that the teacher must define as a Prolog clause and enter as additional information to the problem's data base:

milieu(o,A,M) → ;

cercle(c,o,M) → milieu(o,A,M);

The Prolog clauses express the fact that c is the circle with diameter AM .

Appendix E

Le langage de construction de la figure géométrique dont dispose l'élève est engendré par les operations de construction suivantes:

N: Nommage d'un point.
Syntaxe: ⟨nom⟩ ⟨coordonnéel⟩ ⟨coodonnée2⟩
Le nom du point doit être nouveau.

D: Droite.
Syntaxe: D ⟨noml⟩ ⟨nom2⟩
Décrit la droite spécifiée par deux points connus en traçant un segment entre eux.

E: Equerre.
Syntaxe: E ⟨nom⟩
Décrit la droite perpendiculaire à la dernière droite décrite en un point connu de celle-ci. Trace un segment de droite entre ce point et la périphérie de l'écran.

P: Projection.
Syntaxe: P ⟨noml⟩ ⟨nom2⟩
Projette orthogonalement le point ⟨noml⟩ qui doit être connu sur la dernière droite décrite à un point qui sera nommé ⟨nom2⟩. Trace le segment entre ces deux points.

M: Milieu.
Syntaxe: M ⟨nom⟩ ⟨noml⟩ ⟨nom2⟩
Nomme le point ⟨nom⟩, milieu de ⟨noml⟩ ⟨nom2⟩.

S: Situation.
Syntaxe: S ⟨nom⟩ ⟨coordonnéel⟩ ⟨coordonnée2⟩
Situe un point sur la dernière droite décrite.

F: Fin.
Syntaxe: F
L'élève pense avoir construit une figure décrivant les hypotheses.

Appendix F

A l'execution de chaque ordre de construction de la figure géométrique, un ensemble de clauses est ajouté. Ces clauses doivent permettre, si la figure est correcte, de verifier les hypotheses. Notre exemple utilise les relations suivantes:

npt(n) : nom de point; n est une caractère majuscule.
coor(n,x,y) : x et y sont les coordonnées du point de nom n .
apd(d,n) : le point de nom n appartient à la droite d'identificateur d .
perp(d,d',p) : les droites d et d' sont perpendiculaires.
mil(m,p,p') : m est le milieu de p p' .

Pour chaque ordre:

N n x y : Nommer n le point de coordonnées x y .
On ajoute: npt(n) coor(n,x,y)

D p p' : Droite passant par p et p' .
On génère un nouvel identificateur: d .
On ajoute: apd(d,p) apd(d,p')

P p p' : Projection de p en p' sur la dernière droite décrite d .
On génère d' , la droite perpendiculaire à d .
On calcule les coordonnées x y de p' .
On ajoute: npt(p') apd(d',p) apd(d',p') apd(d,p')
coor(p',x,y) perp(d,d',p)

E p : Equerre en p sur la dernière droite décrite, d .
On génère d' .
On ajoute: apd(d',p) perp(d,d',p)

S p x y : Situation du point p en x y sur d la dernière droite décrite.
x et y sont rectifiés (par projection sur d) en x' y' .
On ajoute: apd(d,p) bpt(p) coor(p,x',y')

M m p p' : m est milieu de p et p' .
On calcule les coordonnées x y de m .
d est l'identificateur de la droite entre p et p' .
On ajoute: npt(m) apd(d,m) coor(m,x,y) mil(m,p,p')

Appendix G: Déroulement d'une étape de démonstration

Voici ce qu'il faut démontrer dans l'étape 1
apc(c,A) apc(c,H) apc(c,M) apc(c,D)
Pour continuer, taper une lettre et validez
v

Voici la liste des théorèmes utilisés en français et en Prolog

0: Pas de théorème à appliquer

1: Si le triangle M P N est rectangle en P
alors P appartient au cercle de diametre M N

apc(C,P) → milieu(O,M,N) cercle(C,O,M) droit(P,M,N);

2: Si deux droites sont parallèle, toute perpendiculaire à
l'une est perpendiculaire à l'autre

droit(A,B,C) → parallèle(B,A , C,D) droit(B,C,D);

3: Dans un triangle, la droite qui passe par le milieu de
deux côtés, est parallèle au support du troisième côté

parallèle(A,B , C,D) → milieu(A,E,C) milieu(B,E,D)

entrez le numéro du théorème choisi
1

A quel problème vous ramènera l'utilisation du théorème choisi?
Entrez une à une, en Prolog toutes les conditions qui permettraient, en utilisant le théorème, de démontrer la conclusion

Entrez une condition en PROLOG ou fin pour finir
apc(c,A);
Entrez une condition en PROLOG ou fin pour finir
apc(c,M);
Entrez une condition en PROLOG ou fin pour finir
milieu(o,A,M);
Entrez une condition en PROLOG ou fin pour finir
cercle(c,o,A);
Entrez une condition en PROLOG ou fin pour finir
apc(c,D);
Entrez une condition en PROLOG ou fin pour finir
fin;
Analyse correcte

Pensez-vous avoir terminé l'analyse complète du problème?
Répondez par oui ou non;
non;
OK

EXPERIENCES sur les APPORTS de l'INFORMATIQUE

à l'ENSEIGNEMENT DES MATHEMATIQUES

I.R.E.M. DE STRASBOURG

Avant-propos

Le présent document relate les réflexions et les expérimentations sur le terrain, menées par un groupe de travail de l'I. R. E. M.* de Strasbourg entre 1982 et 1984.

Ce groupe, comprenant des enseignants-chercheurs et des professeurs de lycée, s'est constitué en vue de partager les expériences de formateurs en informatique acquises dans les actions de recyclage des professeurs de mathématiques.

Au cours de ces formations sont en effet apparues de nombreuses pistes de réflexions sur les apports possibles de l'informatique à l'enseignement des mathématiques.

Il ne s'agit pas ici de proposer ou d'analyser des didacticiels d'accompagnement d'une leçon de mathématiques, mais plutôt d'examiner :

- dans quelle mesure le traitement informatique d'un problème mathématique peut enrichir la démarche de raisonnement classique,

- en quoi l'activité informatique, et plus spécialement algorithmique, s'apparente à l'activité mathématique qu'elle permet alors d'aborder sous un angle nouveau permettant, entre autres, de contourner les blocages psychologiques que provoque l'enseignement des mathématiques chez certains jeunes.

Les pistes ainsi explorées se devaient d'être expérimentées sur le terrain pour être validées. La deuxième partie de ce document relate les expériences qui ont pu être menées par le groupe

* auprès d'adultes, enseignants, non nécessairement initiés à l'informatique,

* auprès de jeunes :

. de l'école élémentaire,

. des classes de terminale littéraire TA2.

...

* Institut de Recherche sur l'Enseignement des Mathématiques.

1. *Enrichir la démarche mathématique classique*

<u>Parallélisme entre le processus de résolution d'un problème mathématique et la construction d'un algorithme</u>

Les méthodes de construction d'algorithmes et de raisonnement mathématique ont en commun :

- la définition d'un nombre restreint d'outils (<u>symboles</u>, <u>actions élémentaires</u> et <u>procédures soigneusement formalisées</u>),

- une méthode de conduite du raisonnement et de décomposition du problème en sous-problèmes plus simples jusqu'à aboutir aux outils évoqués précédemment.

L'apport complémentaire de l'informatique nous paraît résider dans les points suivants :

- la liste de ces outils est plus facile à établir en informatique qu'en mathématiques,

- la nécessité d'aboutir à un programme traité automatiquement par une machine, oblige de pousser le raisonnement jusqu'à son extrême détail, excluant les "on voit que" et autres C. Q. F. D. visant un interlocuteur intelligent et initié,

- l'exécution du programme prolonge en aval l'activité de raisonnement, par une vérification plus complète qu'il n'est souvent possible en mathématique.

La résolution d'un problème reste, en mathématique, une activité un peu brouillonne, tâtonnante et foisonnante, faite d'aller-retour qui n'apparaissent plus dans la version finale rédigée.

Les méthodes de programmation actuelles tentent de s'en affranchir en proposant une conduite rigoureuse et quasi-automatique du raisonnement.

Peut-on toujours y arriver ?

Si oui, que peut-on en tirer pour la conduite d'une démonstration mathématique ?

2. *"Faire des mathématiques sans en avoir l'air"*

Les blocages psychologiques face aux mathématiques, rencontrés chez certains jeunes, tiennent, semble-t-il, au vocabulaire et à la nature un peu abstraite des objets manipulés alors que le but poursuivi est de développer la faculté de raisonner, l'esprit d'analyse et de synthèse.

...

La programmation d'une machine opérant sur des objets aussi divers que nombres, caractères ou chaînes de caractères, signaux lumineux ou plume d'un traceur de courbe conduit à un travail d'ordre logique, structurant les facultés de raisonnement mais mis en oeuvre dans des domaines variés, plus proches de la sensibilité personnelle de tel ou tel élève.

En outre, la réponse quasi-instantanée, neutre et spectaculaire de la machine constitue un stimulant que les clubs informatiques dans les lycées et collèges ont bien mis en évidence.

L'activité algorithmique et la programmation dans des langages structurés, tels PASCAL ou LOGO, constituent donc une forme d'enseignement mathématique dans un contexte nouveau, de nature à contourner certains blocages.

La nécessité de conduire la construction du programme, à partir d'une situation formulée en langage courant, jusqu'au dernier détail, est un facteur de structuration de la pensée et cultive l'intelligence, si l'on définit celle-ci comme la préhension de situations complexes et la faculté de les analyser.

3. *Les expérimentations*

3. 1. l'automate et les deux seaux (cf. fiche-support Annexe 4)

Expériences conduites en 82 et 83 avec des enseignants dans différentes spécialités.

Le but poursuivi était moins de résoudre le problème, que d'analyser la démarche utilisée et de noter, au fur et à mesure, les réactions et comportements engendrés.

3. 1. 1. Les démarches diffèrent selon la formation des intéressés

* ceux qui ont une certaine pratique de la programmation utilisent une méthode :

- soit recherche des structures de contrôle (répétitives, alternatives) puis écriture des conditions et des modules,
- soit déduction, à partir du résultat à obtenir des modules successifs, en remontant aux données, puis écriture des modules,
- soit recherche des instructions "centrales" de l'algorithme, puis insertion dans les structures répétitives, puis écriture des entrées et des sorties ;

* les enseignants de mathématiques utilisent leur manière habituelle de conduire une démonstration :

- analyse critique de l'énoncé sur le plan formel,
- étude d'un ou deux exemples,

...

- évacuation provisoire des évidences,
- recherche sur les points cruciaux (par exemple : comment repérer la suite de caractères M1 dans T ?),
- éventuellement résolution préalable d'un problème plus simple (M1 réduite à 1 seul caractère) ;

* l'énoncé des deux seaux soulève chez les mathématiciens de nombreuses questions d'interprétation des mots ("vider", "remplir",...) qui appellent des définitions plus formelles . L'énoncé est jugé à la fois trop directif (propose une méthode) et pas assez précis (cas qui ne "marchent" pas !).

3. 1. 2. Quelques réflexions sur l'apport à l'enseignement des mathématiques

- la notion de structure répétitive nécessite, plus fréquemment qu'en mathématiques, une globalisation de la démarche, tandis que l'alternative favorise l'esprit d'analyse.
- l'algorithmique met l'accent sur l'initialisation des variables et la dépendance des conditions initiales,
- comment exploiter l'algorithmique pour illustrer la notion de condition nécessaire et suffisante ?
- la notion de case-mémoire éclaire la notion de variable (exemple : $U \longleftarrow 2 * U + 1$ pour $U_{n+1} = 2U_n + 1$).

3. 2. Expérience LOGO (école élémentaire, indiquée pour mémoire).

L'ordinateur est-il un outil d'apprentissage efficace à l'école élémentaire ? Pour pouvoir répondre à cette question, il faut tout d'abord essayer... Nous avons donc placé un micro-ordinateur dans une classe de CM2 et exploré quelques possibilités d'utilisation, en une année scolaire, de ce nouvel outil. L'idée de départ est d'offrir au système scolaire un espace d'exploration, de créativité, d'apprentissage : utiliser l'outil informatique, c'est, pour nous, mettre à la disposition des enfants un outil pour "apprendre mieux".

En effet, si l'élève a réussi à apprendre un certain nombre de concepts géométriques, il doit être capable de les réinvestir dans des situations concrètes. LOGO crée, sans nul doute, un contexte favorable d'exploration. Face au micro-ordinateur nous souhaitons un enfant actif et non consommateur. Cependant, nous n'avons pas retenu la démarche préconisée par Seymour Papert - l'enfant invente les problèmes et les résoud - laissant à l'enseignant le choix des situations-problèmes et son rôle de guide des enfants.

L'intérêt principal d'une programmation active réside dans la démarche algorithmique qu'elle nécessite. L'idée qui nous intéresse est celle de l'aide que peut apporter à l'enfant cette démarche dansla résolution de problème. Amener l'élève à avoir une démarche structurée et à décomposer les difficultés qu'il rencontre est le but que doit permettre d'atteindre l'utilisation des procédures. En cela LOGO est un langage particulièrement adapté.

3. 3. L'option algorithmique des classes de terminale TA2 :

3. 3. 1. Situation de l'expérience :

La classe de terminale est la dernière classe du cycle d'enseignement secondaire et prépare au baccalauréat des élèves agés normalement de 17 à 18 ans.

La section A2 est une section littéraire à dominante philosophie et langues vivantes ou mortes, fréquentée surtout par des élèves de sexe féminin.

L'horaire hebdomadaire de mathématiques est de deux heures. (il est également de 2 h dans la classe précédente de 1ère A2).

Le programme de mathématiques de la classe se compose d'une partie commune très classique, d'analyse et de statistiques et d'une partie optionnelle à choisir dans l'arithmétique, les probabilités, la géométrie, l'astronomie, l'algorithmique. *(cf. annexe 1)*

Le programme de cette dernière option, propose une liste de thèmes qui nous a semblé trop longue et les derniers trop ambitieux, surtout pour des élèves qui n'ont pas encore été initiés à ce type d'activité. L'objectif recherché semble être une information sur les différents types d'algorithmes existants, mais pas un savoir-faire.

Nous avons préféré une démarche favorisant l'activité des élèves et leur permettant d'écrire eux-mêmes des algorithmes simples. En contre-partie, il fallait, naturellement, limiter le nombre des thèmes abordés.

Nous nous sommes beaucoup interrogés sur la place de la machine dans notre enseignement. Il nous semblait que l'algorithmique est une discipline qui peut exister indépendamment de son utilisation en informatique.

Cependant, la motivation du passage sur machine est essentielle dans cet apprentissage. En effet, la transposition quasi-immédiate des algorithmes écrits en un langage de programmation, et l'exécution des programmes obtenus par les ordinateurs donnent un intérêt concret aux algorithmes et justifie même l'activité.

En même temps, le test par ordinateur est un contrôle bien plus efficace que le contrôle du professeur qui semble toujours avoir des références arbitraires. La réussite devient objective et la volonté de dépasser l'échec par la résolution des difficultés est très fortement motivée.

L'utilisation des ordinateurs permet aussi de démystifier l'informatique auprès d'élèves qui, parce que classés "littéraires", ont souvent plus d'appréhensions et moins d'occasions de faire des essais dans ce domaine que les "scientifiques".

3. 3. 2. Les étapes prévues sont les suivantes (cf. fiches en annexe 3)

- notion de séquences d'opérations :
  - piloter un robot,
  - consulter le QUID,
  - consulter un dictionnaire.
- notion de variable et d'affectation de valeurs :
  - échange de contenus de bols,
  - notion de cases-mémoires,
  - recherche mentale : de la somme de 10 nombres, du minimum de ces nombres.
- contrôle n° 1,
- rangement et classement de 3 puis de 4 nombres par recherche des minimas successifs,
- généralisation à N nombres,
- vérification sur ordinateur,
- contrôle n° 2,
- algorithme d'un jeu.

Nous ne pourrons, faute de place, présenter les fiches élèves et professeur complètes dans cet article, nous nous contenterons d'en extraire quelques éléments, pour montrer l'esprit de la démarche :

* échange de contenus de bols (1er exercice)

fiche élève : vous disposez d'un bol BLANC contenant du lait,
d'un bol ROUGE contenant du café.

il s'agit d'échanger les contenus, à l'aide d'un troisième bol VERT.

fiche professeur :

A
VERT ⮌ BLANC
BLANC ⮌ ROUGE
ROUGE ⮌ VERT

"⮌" signifie "transvaser le contenu de"

On pourra suggérer de donner un nom à l'algorithme, et l'utiliser par la suite. Par exemple : ECHANGE (BLANC, ROUGE) pour A

* Extrait du test 1 (exercice 2)

On dispose de 4 mémoires T_1, T_2, T_3, T_4, contenant chacune un nombre entier

T_1	T_2	T_3	T_4
x	y	z	n

Les nombres x, y et z vérifient la relation $x < y < z$.

ALGORITHME

SI $T_4 \leq T_1$ ALORS $\left[\begin{array}{l} T_3 \longleftarrow T_2 \\ T_2 \longleftarrow T_1 \\ T_1 \longleftarrow T_4 \end{array}\right.$

SINON [SI $T_4 \leq T_2$ ALORS $\left[\begin{array}{l} T_3 \longleftarrow T_2 \\ T_2 \longleftarrow T_4 \end{array}\right.$

SINON [SI $T_4 \leq T_3$ ALORS [$T_3 \longleftarrow T_4$

Question 1 : on donne les contenus suivants

T_1	T_2	T_3	T_4
4	7	8	2

et on exécute l'algorithme ci-dessus.

Indiquez, dans un tableau, les contenus de T_1, T_2, T_3 et T_4 à chaque étape de l'exécution.

...

* rangement et classement de 3 nombres :

fiches élève :

- on désigne par x, y les contenus inconnus de deux mémoires M1, M2. Ecrivez un algorithme permettant de classer ces deux nombres, dans l'ordre croissant, dans les mémoires M1, M2. (on utilisera ⟵ et si...alors...).

- Ecrivez l'algorithme correspondant pour trois mémoires M1, M2 et M3 contenant trois nombres inconnus x, y, z.

fiches professeur :

- les exercices qui suivent ont pour but de préparer l'écriture d'un algorithme de classement des contenus de n cases mémoires.

n = 2 : SI M2 < M1 alors ECHANGE (M1, M2)
sinon RIEN

n = 3 : plusieurs solutions seront sans doute proposées par les élèves, une fois vaincues les difficultés des contenus inconnus. Ces algorithmes seront le plus souvent difficiles à généraliser à 4 (ou plus cases) (cf. fiche 4). Proposer alors la méthode suivante :

1°) rechercher le plus petit des trois contenus et le mettre dans M1,

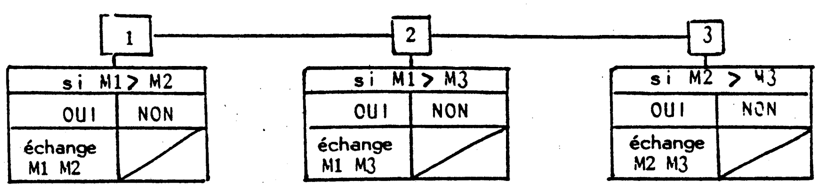
2°) recommencer avec les deux cases restantes M2 et M3.

d'où l'algorithme :

```
SI  M2 < M1  alors ECHANGE (M1, M2)
SI  M3 < M1  alors ECHANGE (M1, M3)
               (* M1 contient alors le plus petit nombre)
SI  M2 < M3  alors ECHANGE (M2, M3)
               (* M2 contient le plus petit des deux restants)
```

* Test n° 2

Soit l'arbre suivant :



a) M1, M2, M3 contiennent respectivement −2, 4, −5. Indiquez pour chaque test si l'on parcourt la branche OUI ou NON :

1	2	3

b) donner un exemple de contenu de M1, M2, M3 tel que l'on ait :

1	2	3
OUI	NON	OUI

c) Quels sont tous les déroulements possibles ? Pour chacun de ces déroulements, donner des contenus initiaux de M1, M2, M3.

fiche professeur : on pourra utiliser une présentation du type suivant

M1	M2	M3	1	2	3
2	1	4	oui	non	non

Remarque : trois contenus placés dans le même ordre, au départ, donneront une exécution du même type.

3. 3. 3. Remarques faites sur le terrain et conclusions

- vif intérêt des élèves pour ce type d'activité qui éveille leur curiosité,

- le thème du classement deviendra monotone à partir de la fiche 4 (8e heure),

- la manipulation de bols fait bien comprendre la notion d'échange de contenus mais introduit mal l'affectation : ("A ⟵ B entraîne B vide" restera une idée tenace),

- la construction d'algorithmes à partir de données numériques concrètes crée un obstacle lorsqu'il s'agit d'écrire un algorithme valable quelles que soient les valeurs mises dans les cases-mémoires. Il apparaît difficile à certains élèves de dissocier l'algorithme des données et des résultats,

- l'utilisation à un moment donné d'une machine, si elle n'est pas indispensable, constitue un regain d'intérêt des élèves et valide en quelque sorte le travail effectué sur feuille,

- le passage de N=4 à N quelconque pour le classement constitue une deuxième étape d'abstraction difficile à franchir par certains élèves. Elle coïncide avec l'introduction de la répétitive et soulève la difficulté de globalisation, déjà évoquée plus haut,

- l'exercice du ROBOT et le problème de l'octogone dans la fiche de contrôle révèlent des lacunes en géométrie, d'où l'idée d'une option géométrie algorithmique avec utilisation d'un outil tel que LOGO.

A N N E X E S

Extraits du programme de Terminale TA2

ACTIVITES ALGORITHMIQUES

1. Classements :

   Algorithmes de rangement de nombres par ordre croissant, de mots dans l'ordre lexicographique.

2. Tris :

   Ranger des objets par sous-ensembles selon certaines caractéristiques.

   Par exemple : ranger des entiers selon le nombre de leurs diviseurs, crible d'Eratosthène pour les nombres premiers.

3. Accès à un fichier :

   Recherche d'un nombre, d'un mot dans une liste.

4. Algorithmes arithmétiques :

   Division euclidienne, algorithme d'Euclide.
   Bases de numération et problème des opérations sur les "grands nombres" au moyen d'une calculatrice

Commentaires

ACTIVITES ALGORITHMIQUES

Nous entendons ici par "algorithme" une suite finie et ordonnée d'instructions de calculs, ou d'opérations logiques.

L'importance des procédures algorithmiques en mathématiques et surtout dans des activités para-mathématiques (gestion de données de stocks, constitution et utilisation de fichiers, codages...) liées à la banalisation de l'informatique, justifie amplement une initiation à ce type d'activités. Les objectifs d'une telle initiation sont :

- analyse d'un problème et de son traitement algorithmique,
- description d'un algorithme,
- comparaison de différents algorithmes permettant de résoudre un même problème.

On se servira d'exemples variés (voir programme) pour réaliser les objectifs ci-dessus.

On notera que certains algorithmes simples peuvent s'exécuter "à la main" ; d'autres ne nécessitent que des moyens très limités (calculatrice de poche, programmable ou non).

Enfin, on observera que, malgré l'apparente trivialité des problèmes de classement, une grande partie de l'activité informatique est consacrée à ce type de questions.

IREM - Groupe Informatique et enseignement des mathématiques
Seance du 25 novembre 1983

ALGORITHME DES DEUX SEAUX

Consignes :
* faire le travail le plus tard possible avant la réunion
* noter toutes les phases de la recherche, les impressions et réactions (compréhension de l'énoncé-approche de la recherche-points de blocage -manières de débloquer -exemples utilisés - schémas - fin du travail
* après résolution, reprendre les notes ci-dessus et relever les points qui semblent importants.
* pour chacun de ces points, rédiger une QUESTION permettant de comparer les démarches des participants

La réunion du 25 consistera à collecter ces questions (critères) puis à y répondre en faisant un tour de table pour chaque question , ce qui permettra l'analyse comparée des démarches .

Le SUJET

On dispose de deux seaux A et B , de capacité maximale AMAX et BMAX en litres (AMAX > BMAX). Il s'agit d'obtenir toutes les capacités intermédiaires entières en litres, les seules opérations possibles étant :

- remplir un seau (robinet)
- vider un seau (vidange)
- verser le contenu d'un seau dans l'autre
- remplir un seau avec l'autre

On écrira un algorithme permettant la simulation sur ordinateur des opérations précédentes , en utilisant les ordres habituels (lire, écrire, ← , si..sinon, tant que ..,les opérateurs arithmétiques sur les entiers etc...) .
Par exemple , A←AMAX correspond à "remplir le seau A".
Les capacités intermédiaires seront obtenues dans le seau A dont on affichera régulièrement le contenu . On cherchera à écrire un minimum d'instructions .

-------***ooo***-------

FICHE 1 - B — QUID ?

Répondre aux questions suivantes, en vous servant d'un "QUID". NOTEZ au fur et à mesure ce que vous faites pour trouver la réponse.

Question 1 Quels sont les meilleurs millésimes de Bourgogne depuis 1970 ?

Question 2 Le grand Rabbin de France est-il nommé ou élu, et par qui ?

Question 3 Le Bonheur, d'Agnès Varda, fut-il un film primé. De quand date-t-il ?

Question 4 On parle souvent de "puces" à propos des ordinateurs. En quoi est faite une "puce" et quelle quantité d'information peut-elle contenir ?

FICHE 1 - C — ALGORITHME DU DICTIONNAIRE

Vous savez chercher un mot dans un dictionnaire.
Déterminez les actions élémentaires nécessaires et
Ecrivez l'algorithme de recherche d'un mot (succession des actions élémentaires nécessaires)

Exercice

Ecrivez, sous forme d'algorithme exécutable par l'un d'entre vous, la confection d'une recette de cuisine que vous connaissez.

FICHE 2 - A | **UNE HISTOIRE DE BOLS**

1 Vous disposez d'un bol BLANC contenant du lait
d'un bol ROUGE contenant du café

Il s'agit d'échanger les contenus, à l'aide d'un troisième bol
La seule opération permise est :
"Verser le contenu d'un bol dans un bol vide", notée ↩
par exemple : ROUGE ↩ BLANC signifie verser le contenu du bol BLANC dans le bol ROUGE.
Ecrivez l'algorithme correspondant, qu'on appellera

ECHANGE (BLANC,ROUGE)

Y a-t-il d'autres algorithmes permettant d'obtenir le même résultat ?

2 Un quatrième bol bleu contient du thé. Ecrivez l'algorithme permettant le passage du contenu 1 au contenu 2 :

	BLEU	BLANC	ROUGE
CONTENU 1	Thé	Lait	Café
CONTENU 2	Lait	Café	Thé

- Illustrez le <u>déroulement</u> de l'algorithme en représentant les contenus successifs des différents bols.
- Appliquez l'algorithme obtenu, en prenant comme contenu initial, le contenu 2. Que contiennent alors les différents bols ?
- Y a-t-il d'autres algorithmes possibles pour obtenir le même résultat ?
- Réécrivez l'un de ces algorithmes en vous servant de l'opération ECHANGE (X,Y), vue au 1 , où X,Y désignent deux bols quelconques.

FICHE 3 - A | **UNE DICTEE**

1 On va dicter dix nombres.

. Pendant cette dictée vous ne <u>notez aucun nombre</u>.
. Vous ne pouvez garder en mémoire <u>que deux nombres</u>

a) A la fin de la dictée, vous devez donner <u>la moyenne</u> des dix nombres. Comment procédez-vous ?

b) A la fin de la dictée, vous devez donner <u>le plus petit</u> des dix nombres. Comment faites-vous ?

c) Même question pour dix mots en utilisant l'ordre alphabétique.

Annexe 4

PROGRAMMATION D'UN AUTOMATE

Il s'agit de fabriquer une machine fonctionnant de la façon suivante :

Lors de sa mise en route, on y met successivement trois billets :

- le premier contient un texte (suite de caractères) T
- le second contient un mot (" ") M1
- le troisième contient un mot M2

A l'issue du traitement, la machine fournit le texte T' obtenu à partir de T de la façon suivante : chaque fois que la suite de caractères M1 est contrée dans T, elle y est remplacée par la suite M2.

On peut faire exécuter à la machine une <u>succe ssion d'actions simples</u> ,ordonn sur une liste placée définitivement dans la machine avant usage .

On demande d'établir cette liste à partir des actions simples suivantes :

- lire un billet après introduction dans la machine
- ranger sous un nom donné (tiroir)
  - un nombre ou une suite de caractères (éventuellement vide)
  - le résultat d'une opération
  - le contenu d'un billet
- effectuer les opérations arithmétiques classiques sur les nombres
- écrire un billet et le sortir de la machine
- extraire une sous-suite $S_{i,j}$ d'une suite S , formée des j caractères (j>=1) rencontrés à partir du i-ème .
- déterminer la longueur l(S) d'une telle suite .
- ajouter une suite S' de caractères à droite d'une suite S (opération notée S|S')
- établir des <u>conditions</u> portant sur les objets précédents et détermin si elles sont vraies ou fausses .

On pourra donner un nom à une liste bien déterminée d'actions simples .

On pourra en demander <u>l'exécution</u> selon qu'une condition est vraie ou fausse :

si <u>condition</u> ‖ alors exécuter <u>liste1</u>
‖ sinon exécuter <u>liste2</u>

L'exécution d'une liste pourra être <u>répétée</u> tant qu'une condition reste vraie

tant que <u>condition</u> ‖ exécuter <u>liste</u>

<u>exemples</u> :

* <u>SOMME</u> (calcule la somme de N éléments introduits sur des billets)

- ranger 0 dans S
- lire le nombre de termes à additionner
- ranger ce nombre dans N
- ranger 1 dans i
- tant que i <= N exécuter CUMUL
- écrire le nombre N
- écrire la somme S

* <u>CUMUL</u>

- lire un nombre à additionner
- ranger ce nombre dans A
- calculer S+A
- ranger le résultat dans S
- calculer i+1
- ranger le résultat dans i

* <u>OCCUR</u> (compte le nombre de lettres "E" dans un texte introduit préalablement)

- lire un texte
- ranger ce texte dans T
- ranger 0 dans N
- ranger 1 dans i
- tant que i <= l(T) exécuter TEST
- écrire N

* <u>TEST</u>

- si $T_{i,1}$ ="E" ‖ alors calculer N+1,ranger dans N
‖ sinon rien
- calculer i+1 , ranger dans i

First Year Results of the Microcomputer Assisted Mathematics Remediation Project at Arizona State University

Dr. Gary Bitter
Allan Cameron
Sharman Pitcher

Abstract

Due to the concern regarding teacher math skills, a project was initiated at the Arizona State University (A.S.U.) Microcomputer Research Clinic. The purpose of the project was to examine the potential of a remediation program via the microcomputer.

A math achievement test was developed on the Apple III microcomputer. In the fall semester 1982, undergraduates in the elementary education program at A.S.U. took the test. The test was analyzed and revised. In the fall semester 1983, 114 students enrolled in A.S.U.'s Methods of Teaching Mathematics class were tested with the revised examination. As part of the course requirements, a score of 70% or above was necessary. Of the non-passing students, eleven were enrolled in a programmed textbook. There were ten students who used the programmed textbook. There were sixteen students who took a posttest, however, their mode of remediation was unknown.

There were significant gains in pretest and posttest scores in all three groups. There was no significant difference between the gain scores of the three groups. The project is scheduled to continue for one more year. The desired outcome is an effective microcomputer mathematics remediation program.

College of Education
Arizona State University
Tempe, Arizona
85287

First Year Results of the Microcomputer Assisted Mathematics

Remediation Project at Arizona State University

Introduction

The United States is experiencing a crisis in mathematics education. There is a severe shortage of teachers who are well-trained in mathematics; and nearly all of the states report critical shortages of qualified mathematics teachers.

There are several reasons for the problem of low mathematics competence of teachers. College admission standards in mathematics have been lax and teacher preparation programs have focused more on educational methods rather than subject matter competence. Prospective teachers with minimal backgrounds in mathematics consequently often avoid taking even required mathematics courses as long as possible. An underlying anxiety about studying mathematics creates a poor mental environment for approaching mathematics instruction.

In response to the crisis in mathematics edcucation, many local communities, states, and universities are increasing the standards for mathematics instruction. However, many preservice teachers are caught in a bind between facing increased mathematics competency requirements and increased levels of mathematics anxiety. Thus, there is clearly a critical need for a novel program leading students toward a positive approach to the learning of mathematics.

In the U.S., SAT mathematics scores have shown a virtually unbroken decline from 1963 to 1980; average mathematics scores dropped almost 40 points. In a report issued by the Conference Board of Mathematics, remedial mathematics courses in colleges have increased over 75 percent since 1975. These deficiencies come at a time when the nation's demand for highly skilled employees is rapidly accelerating.

These difficulties are compounded by a severe shortage of well-trained teachers. In a 1981 survey 43 of 45 states studied were experiencing critical shortages of mathematics teachers. The National Commission on Excellence in Education also reported that half of the newly employed teachers of mathematics and science are not qualified to teach these subjects.

While shortages have been created by the greater financial attraction of industry, forcing schools to look to less-qualified individuals; it is likewise true that many teachers have been inadequately trained in our colleges and universities. However, college admission standards and teacher preparation deemphasizing subject matter and competence are but a couple of the problems. In addition, prospective teachers have avoided or delayed taking needed mathematics instruction, indicating that the teachers themselves may be experiencing math anxiety.

In response to the concern for public education, state and local school officials are now raising the education standards in their communities. Since 1980, high school graduation requirements have been encreased in 30 states and 12 other states are considering initiatives to raise present requirements. Over half of the states are also raising admission standards for public universities, and many states are requiring more stringent competency tests for teacher certification. Increased mathematics requirements are a major component of the trend toward high academic standards. Schools at all levels, therefore, will be faced with increasing the number and quality of their mathematics course offerings.

Unfortunately, the increase in standards causes a double bind for schools. They are to provide more mathematics training but more teachers are not available; without more teachers, schools cannot provide the increased mathematics education. There are problems for colleges as well. Most university mathematics departments are extremely reluctant to offer mathematics courses to non-majors. When mathematics departments are forced to offer teacher-related mathematics courses, they often staff them with graduate students who are untrained to teach and may even be resentful about their job assignment. Such conditions drive prospective mathematics teachers, especially members of minority groups, into other fields thus exacerbating a looming teacher shortage.

<u>Statement of the Problem</u>

The purpose of this study was to explore the potential of a computer assisted mathematics remediation program that would identify preservice teachers who were lacking minimum mathematics compentency and effectivly correct their deficiencies. The program would have to be flexible in the time requirements of its administration. Examinations and remediation would be administered at times convienent to each student.

Review of Related Literature

Prospective teachers with minimal mathematics backgrounds often delay taking required mathematics courses because of anxiety or fear of failure. The mathematically anxious student associates an emotion, attitude, or expectation with mathematics in such a way that the attitude blocks mathematical performance (Donady and Auslander, 1980). Betz (1978) found that mathematics anxiety is more prevalent among women than men. Others have determined that women drop out of the study of mathematics as soon as mathematics becomes optional (e.g., Fauth and Jacobs, 1980; Fox, Fenna, and Sherman, 1977; Tobias, 1980). Since more than half of preservice students are women who have taken only the minimum number high school mathematics courses, colleges of education have many students with low mathematics skills and high mathematics anxiety. Such teachers who lack confidence in mathematics are often reluctant and ineffective mathematics teachers and find it difficult to adapt new curricula. And so another generation with mathematical deficiencies is spawned, continuing the cycle.

CAI is being used at all levels of education and in myriad applications from elementary reading to high-level military strategy. These studies indicate the effectiveness of CAI in these applications (e.g., Misselt, et al., 1980; Tatsuoka, et al., 1978; Chambers, 1980; Loop, et al., 1980; Marshal, et al., 1980; Lockhart, et al., 1980).

Of more concern to this particular study, though, are reports of the effectiveness of CAI programs used specifically in traditional school settings to enhance conventional methods of education. James Kulik and his colleagues have published an integrative study of 51 independently conducted evaluations of CAI programs used with students in grades six through twelve. Kulik's meta-analysis of these programs resulted in a healthy prognosis for CAI. In particular Kulik found that CAI raised students' final examination scores significantly: CAI also had positive, though less significant, effects on students' ability to retain concepts learned through CAI; CAI improved students' attitudes toward the material they were expected to learn as well as toward computers in general; and CAI substantially reduced the time students required to learn material (Kulik, 1983).

Kulik's results have been corroborated by other researchers. In another integrative report of ten studies of CAI programs, Vinsonhaler and Bass (1972) found that drill-and-practice CAI produced gains of between one and eight months of learning time in children taught via the computer as compared to those who were taught by traditional methods. Jamison, Suppes, and Wells (1974) showed that students taught with CAI achieved better end-of-course scores and required less study time than students taught by traditional methods. Jamison, Suppes, Wells (1974) found that CAI offers particular benefits to disadvantaged students. Comparing final examination scores, Edwards, Norton, Taylor, Weiss and Duseldorp (1975) showed that students taught with CAI achieved better end-of-course scores and required less study time than students taught by traditional methods.

Research has also been done to determine how effective computers are in teaching various subjects included in traditional curricula. For example, the Educational Testing Service studied the drill-and-practice programs taught in Los Angeles schools and found that computers could enhance the computational abilities of students. The same study determined that results in reading and language CAI were not as promising, although the results were often favorable (Fagosta, et al., 1981).

Many researchers have reached the same conclusion: CAI is an effective means of teaching mathematics in less time than required by conventional teaching methods. Reviewing 50 CAI programs in mathematics, Overton (1980) found that several programs reported substantial savings of time. More recently Jenson (1982) studied microcomputer-assisted teaching of addition to first through third graders and concluded that the CAI teaching did save time. He attributed the time savings primarily to the fact that the microcomputer repeats only problems with which the student has had difficulty.

CAI possesses qualities that make it an effective learning device. Bright (1983) outlines several of these qualities in explaining the high rate of success experienced by teachers implementing CAI programs. Bright points to the length of time that students are willing to spend interacting with computers. He notes that students will spend considerably more time with computers than with other more traditional instructional materials. Those who have supervised CAI programs tell stories of coaxing students away from computers to participate in other activities.

Bright also notes that the substantive interaction between computer and user is a distinct benefit in the CAI process. The microcomputer can provide personalized feedback which deals specifically with the user's response to a given question. This feedback provides immediate and relevant information to the learner, causing the student to become more actively engaged in learning.

Finally, Bright suggests that CAI appears to enhance the success experienced by the learner. With software programmed for individualized instruction, students learn at their own rate. The element of challenge within a program can be adjusted according to the abilities of the students. This enables students who have difficulties to progress at a slower rate, thus experiencing success which might be denied them if they were forced to attempt learning at the level of a group with mixed abilities. Similarly, a gifted student can be given a more challenging program of study, preventing boredom that can occur in a group setting.

Research has shown that computers are good teachers, particularly in the field of mathematics. The computer's admirable patience allows it to provide repetitious practice for which human teachers often have too little time. The computer's interactive capabilities enable it to communicate with students on a one-to-one, nonthreatening basis. And, perhaps most importantly, the advent of features such as graphics, speech synthesis, and music and color capabilities make the computer a vivid, exciting tool which helps learning be more enjoyable as well as faster

Methods and Procedures

Instrumentation

For this project, a mathematics achievement examination was needed that would:

1. Identify the students not in need of remediation
2. Determine an ability score for students in need of remediation (pretest).
3. Determine an ability score during remediation.
4. Determine an ability score after remediation (posttest).

In addition to the project requirements, it was decided that the test should also:

5. Make use of the random digit generation capabilities of the microcomputer to produce different items with each administration of the test.
6. Use completion-type responses instead of multiple choice-type responses.
7. Inform the student of the test results upon completion of test administration.

After an inspection of the SRA Level H (12 grade level) math battery, the Arizona Teacher Proficiency Examination, and seventh and eighth grade text books (copyrights 1978-1982) it was decided that the microcomputer math test should contain the following topics:

> Positive Integers, Common Fractions, Decimals, Percents, Negative Numbers, Exponents and Roots, Numeration, Pre-Algebra, Geometry, Measurement, Averages, Graphing, Metric Measurements, Probability, Rates, and Ratio and Proportion.

A "panel of experts" was selected from the subscription list of the "School Science and Math Journal" and asked to weight each topic. These results were used to determine the number of items per topic needed for a 66 item test.

It was also decided that the test items would sorted as to type: computation, concept, or application. Using the SRA math battery, CAT math battery, and the Arizona Teacher Proficiency Examination it was decided by the project director that the proportion of the iten types would be in the following ranges:

Computation : 33%-50% of test

Concepts : 25%-33% of test

Applications: 25%-33% of test

The test was written and programmed by a graduate student working with Dr. Bitter on the project.

The pre-tryout was conducted with elementary education undergraduates in the fall semester of 1982. A correlation of pre-tryout scores and SRA scores indicated a .88 correlation. Each item was reviewed for content, difficulty, and discrimination. 24 of the 66 original test items were discarded and 18 new items were constructed.

In the Fall semester of 1983, the revised test was administered to 114 students. The difficulty was determined to be:

difficulty = .6432

standard deviation = .179

standard error = .0167

A reliability estimate, using the Kuder-Richardson Formula No. 20 (KR20), was .885343

An item analysis determined 49 of the 60 items had difficulties between .4 and .9 with a discrimination index greater than .2 (using point-biserial correlation cofficient).

Students in EED 380, a math methods course for elementary education majors, were required to take the math achievement test. A score of 70 or above was a requirement for course completition. Students took the test on their own time in the microcomputer research lab. Available times for taking the test were 8:00 a.m. to 9:30 p.m. Monday through Thursday, 8:00 a.m. to 5:00 p.m. on Friday, and 9:00 a.m. to 12:00 p.m. on Saturday.

The subject appeared at the lab and was assisted by the lab attendant in setting up the computer for the test. Instructions were given by the lab attendant. Subjects were told how to enter their answer and make changes if necessary. Entering fractional and mixed number answers required instructions with regard to proper keys to be pressed. Other instructions appeared on the screen. The subject was told that he/she could use paper and pencil to compute answers, but no calculators were permitted to be used during the test. The average time spent taking the test was approximately one and one-half hours. When the student completed the test a score was displayed on the screen. The lab attendant then recorded the score.

By class announcements, posters, and memos, the students were made aware of the lab's offering of a remediation program for those students desiring to better their math skills. Students desiring remediation made appointments with one of the investigators for diagnosis and program orientation. There were 21 students who went through the remediation program. They were assigned to either a CAI remediation program or a textbook remediation program.

Those assigned to the textbook program were introduced to the text

"Arithmetic: A Programmed Worktext, Fourth Edition", Arthur Heywood. They were told that they could come into the lab at their convenience during the lab hours and check the book out. Study was to be done in the confines of the lab since there was only one book available. When the student felt prepared for the test he/she would then take the test again. The second test score was recorded for purposes of this investigation.

Those assigned to the CAI program were oriented to the Mathware program. They were also told that they could come into the lab at their convenience during the lab hours and work on the computer. When the student felt prepared for the test he/she would then take the test again. This score was recorded for purposes of this investigation.

Results

During the first month of the fall 1983 semester, 114 students enrolled in Arizona State University,s Methods of Teaching Mathematics class, went to the microcomputer labratory and completed the "Computer Assisted Mathematics Examination". The mean score was 64.32% of the items correct (S.D.= 17.93, S.E.= 1.68). Of this group, 58 students achieved a score of less than 70 percent of the items correct. These 58 students were notified by their instructors that to receive a grade in their methods course they would have to:

1. retake the test until they recieved a score of 70% correct or
2. pass the Arizona Teacher's Proficiency Examination, Mathematics subtest.

In the group of 58 non-passing students, 21 students did not retake the "Computer Assisted Mathematics Examination" for reasons which were not tabulated nor analyzed for this report.

One of the objectives of the project was to initiate a mathematics remediation program that would increase students performances on the Math Achievement Test. As table I shows, students who did not pass the pretest and took the posttest demonstrated significant score gains (t=10.9722, df= 36, p<.01).

Table I
Descriptive Statistics of Scores (% correct) of
Students who took more than one test (N=37)

Score Type	Range	Mean	Standard Deviation	Standard Error
Pre test	15 - 70	52.22	13.72	1.80
Posttest	43 - 85	72.89	9.10	1.49
Difference	4 - 55	20.37	11.30	1.85

The 37 students were classified by their method of remediation and their results compared (Table II). The Kruskal-Wallis one-way analysis of variance by ranks test was used to determine if the differences among the pretest-posttest difference scores were significant. This analysis indicated no significant difference among the three groups (H=.260155, df2, p>.8).

Table II
Descriptive statstics of pretest posttest difference scores (% correct) for three groups of remediation.

Type of Remediation	N	Range	Mean	Standard Deviation	Standard Error
Computer remediation	11	4 - 55	20.64	15.15	4.57
Text remediation	10	8 - 44	20.70	11.59	3.66
Unknown remediation	16	10 - 40	20.00	8.51	2.13

Discussion

Two questions arise from the findings of this study:

1) Why did all three groups achieve significant gain scores?

2) Why was there no significant difference between the gain scores of the three groups.

One possible answer to the first question would be seen as a motivation factor. In order for a student to successfully complete the mathematics methods course a student must correctly answer 70% of the items. The student would conceivebly be motivated to increase his/her performance in order to pass the class. Some of the students attend Arizona State University part time in the evenings and live as much as 80 miles from campus. These students chose textbook remediation because of the limited time spent on campus (the only available site for computer remediation). Many resident students preferred the microcomputer remediation because of its novelty and the immediate feedback provided by the computer assisted programs. Both groups using computer and textbook remediation received diagnostic counseling in order to individualize their program of study. The investigators felt that the students, therefore, chose the remediation and topics that would maximize their study time.

Students needed to score at least 70% in order to complete the course requirements. Once that level was achieved the student would discontinue remediation. Since the pretest mean was 52.22%, a gain score of approximately 20% for all groups would satisify the course requirement. Secondly, due to the small number of subjects in remediation groups, small differences, if any, would not be revealed after data analysis. As a result, this study may not have been senstive to gain score differences between groups.

Conclusions

The major intent of this particular study was to explore the possible use of computer mathematics remediation for pre-service educators. The first phase of the project required the developement of a microcomputer achievement test of skills relevent to teachers. First an examination of intermediate level mathematics textbooks and standardized tests was conducted and a list of item topics was constructed. Then the list of prospective test topics was sent to a panel of experts in the field of mathematics education for validation and weighting of each topic's contribution to the whole test. Another inspection of various standardized tests used for pre-service teachers' mathematics achievement ability was conducted and the proportion of concept, computation, and application items was determined. A pre-tryout computer administered mathematics test was then designed and programmed. The test was administered to a sample of undergraduate students. An item analysis was performed on the test and those items judged not useful were discarded. The revised version used in this study was completed. In the fall of 1983, 114 students were tested. Of the 58 students who failed to achieve at least 70% of the items correct, 37 were available for study. There were 11 students in the computer remediation group, 10 in the programmed textbook group, and 16 chose their own method of remediation.

The 37 students who participated in the remediation averaged a significant 20.37 pre-post gain score. Although all three groups showed significant gains, there was no significant difference between gain scores of the three remediation groups.

Recommendations for Further Study

Certain enhancements to the project would increase the potential questions and research activities that could be undertaken. A computer timing device that would measure time spent per test item could reveal inefficient test items. The same device could be used to more accurately examine and record time required to take the test for use in various correlational studies. Time-on-task studies for computer remediation students could be incorporated for additional information.

Administrative procedures that would record student attributes such as: gender, ethnic background, age, math experience, math anxiety, computer familiarity, and other variables could be used in further investigations.

BIBLIOGRAPHY

Betz, Nancy E., "Prevalence, Distribution, and Correlates of Math Anxiety in College Students," Journal of Counseling Psychology, Vol. 25, pp 441-48, 1978.

Bright, George W., "Exploring the Efficiency of Computer Assisted Instruction," AEDS Journal, pp 144-152, Spring 1983.

Chambers, Jack A. and Bork, Alfred, "Computer Assisted Learning in U.S. Secondary/Elementary Schools," ED 202 461, 1980.

Donady, B. and Auslander, S.B., Resource Manual for Counselors/Math Instructors, Washington School of Psychiatry, Washington, D.C., 1980.

Edwards, J., Norton, S., Taylor, S., Weiss, M., and Dusseldrop, R., "How Effective is CAI? A Review of Research," Educational Leadership, Vol.33, pp 147-153, 1975.

Fauth, G., and Jacobs, J.E., "Equity in Mathematics Education: The Educational Reader's Role," Educational Leadership, Vol. 37, pp 485-490, 1980.

Fox, L.H., Fennema, E., and Sherman, J., Women and Mathematics: Research Perspectives for Change, NIE, Washington, D.C., 1977.

Jamison, D., Suppes, P. and Wells, S., "The Effectiveness of Alternative Instructional Media: A Survey," Review of Educational Research, Vol. 44, pp 1-67, 1974.

Jenson, C.B., "The Enhanced CAI Tutorial," in Proceedings of the National Educational Computing Conference, Kansas City, 1982.

Kearsley, Huntr, and Seidel, "Two Decades of Computer Based Instruction Projects: What Have We Learned?" T.H.E. Journal, pp 90-94, January 1983.

Kulik, James, A., Bangert, Robert L., and Williams, George W., "Effects of Computer-Based Teaching on Secondary School Students," Journal of Education, Vol. 75(1), pp 19-26, February 1983.

Kvarnes, Robert G., "Anxiety and Math Anxiety: Some Theoretical Considerationns," Resourrce Manual for Counselors/Math Instructors, Middletown, CT, Washington School of Psychiatry, 1980.

Lockhardt, Kathleen A. and others, "Computer-Managed Instructionn in the Navy: IV. The Effects of Test Item Format on Learning and Knowledge Retention," Navy Personnel Research and Development Center, ED 202 462, 1980.

Loop, Liza, "Exploring the Microcomputer Learning Environment," ED 201 307, 1980.

Mathware Systemm Software, 919 14th Street, Hermosa Beach, CA 90254, Copyright 1981.

Melmed, Arthur, "The Acceptance of C.A.I. in the Educational Field and Its Probable Use in Elementary and Secondary Education in the Next Decade," Paper presented at the Third International Learning Technologies Congress and Exposition, Washington, D.C.,, February 25-27, 1980.

Misselt, A. Lynn, and others, "Implementation and Operation of Computer-Based Education," MTC Report No. 25, Final Report, ED 201 318, 1980.

Overton, V., "Research in Instructional Computing and Mathematics Education," Viewpoints in Teaching and Learning," Vol. 57(2) pp 23-26, 1981.

Phi Delta Kappa Center, "Microcomputer in Education--Two Inservice Experiences," Practical Application of Research, Vol. 4, No. 4, June 1982.

Ragosta, Marjorie, Holland Paul, and Jamison, Dean, "Computer-Assisted Instruction and Compensatory Education: The ETS/LAUSD Study, Final Report," Educational Testing Service, 1981.

Tobias, Sheila, "Math Anxiety: What You Can Do About It," Today's Education, p 26 Sept-Oct, 1980.

Tatsuoka, Kikumik, and Misselt, A. Lynn, "Attitude and Performance of Military Students and Instructor Attitude in Computer-Based Technical Training," MTC Report No. 23, Ed 201 320, 1978.

Vinsonhaler, J.F., and Bass, R., "A Summary of Ten Major Studies on CAI Drill and Practice," Educational Technology, Vol. 12, pp 29-32, 1972.

COMPUTER ANIMATIONS IN MATHEMATICS TEACHING
AT THE OPEN UNIVERSITY

D. J. Saunders

British Broadcasting Corporation
Open University Production Centre
Walton Hall
Milton Keynes
MK7 6BH
U.K.

Abstract:

This paper gives some of the background to the use of computer animation in the teaching of mathematics at the Open University. It discusses the difference between animation and other types of computer-generated graphics in terms of both teaching requirements and the details of design and production. Examples of computer animations covering several different topics are described.

Ce papier explique un peu l'utilisation de l'animation des ordinateurs dans l'enseignement des mathématiques à la "Open University". Il discute la différence entre l'animation et d'autres genres de graphiques produits par l'ordinateur en termes des besoins d'enseignements et des détails de dessin et de production. Des examples des animations d'ordinateur qui couvrent plusiers sujets sont décrits.

COMPUTER ANIMATIONS IN MATHEMATICS TEACHING
AT THE OPEN UNIVERSITY

0. INTRODUCTION

The Open University of the United Kingdom (OU) was founded in 1969 in order to provide degree-level education for adults who may not have the formal qualifications normally required for attendance at a University. It has a large number of students (25 000 in 1971, its first year of operation, rising to 66 000 undergraduate students in 1984). Courses offered by the University are designed for students who work at home, possibly hundreds of miles from the University campus in Milton Keynes, and so the normal teaching methods used by residential universities are not appropriate. Instead, the main teaching resources provided by the OU are large numbers of specially-prepared texts posted regularly to students, together with some audio-visual material and a programme of continuous assessment: this assessment, combined with an examination, provides the student's overall grade for the course.

Clearly, students working from home face disadvantages not normally experienced by residential students. To reduce this sense of isolation, many courses have associated television programmes broadcast on the national the national BBC television networks. These programmes are made for the OU by a special BBC production centre and, as well as aiding the teaching of their respective courses, also provide a shop window for the OU to display the variety of its undergraduate provision to the general public. In addition, the OU provides a small amount of face-to-face tuition for its students by arranging local tutorial sessions with part-time tutors from other local institutions.

The mathematical provision of the OU consists of a range of courses at undergraduate level and a rather smaller number for postgraduate students. The undergraduate entry course is known as M101 (all OU courses have a code consisting of several letters and a three-digit number)[1] and is meant to require about 300 hours work from the student. A total of six such courses is needed for an ordinary degree, or eight for an honours degree. Many of these courses have television programmes associated with them, and from the beginning it was realised that one of the most powerful television teaching techniques would be to use animations - moving or changing diagrams which would provide clearer illustrations than the static diagrams which are found in textbooks. Indeed, although animations are not intended to replace symbolic arguments, they can sometimes provide a better understanding of the meaning of that symbolism than a discussion using words. Since these moving diagrams often have a straightforward mathematical description it is natural to attempt to generate them by computer.

This paper is organised as follows. Section 1 expands on the use of computer-generated animations in OU television programmes, comparing their use with that of microcomputer-generated graphics now found in schools and universities. Section 2 gives several examples of the type of work produced using these techniques, and section 3 considers some technical points related to the difficulties of giving the impression of movement. Section 4 discusses the design and production of animations, and section 5 considers possible future developments.

1. GRAPHICS AND ANIMATIONS

The development of mathematical intuition is as much a part of mathematics education as the acquiring of skills in the manipulation of symbols. Diagrams have always been an aid to this process, whether diagrams in a textbook, on a blackboard, or in a student's own work. The latter kinds of diagram will tend to be rough and ready: allowance is made for freehand drawing, and the intention is to give an impression rather than to portray exact detail. In contrast, textbook diagrams have the benefit of much more preparation, and have the potential for being geometrically accurate.

The advent of computers - particularly cheap microcomputers with built-in graphics displays - has introduced important changes in the way that diagrams can be used. No longer is there a trade-off between accuracy and immediacy. The computer can draw a diagram of, say, a new graph while the teacher (or student) is sitting in front of the screen. This speed of response, where calculations which formerly took minutes or even hours can be performed in a fraction of a second, has opened up a whole new area of interactive teaching. It is now much more feasible to illustrate many of the abstract concepts in mathematics using a range of examples. Of course, the examples are not intended to replace the abstraction: their purpose is to give a feeling for what is going on, so that the abstract ideas can be more readily understood.

The first of these inexpensive microcomputers appeared on the market in the late 1970s. Before this, the OU had decided to use computing power to help its students picture some of the ideas in its mathematics courses. Here, however, the problem was rather different. Even if microcomputers had been around in 1971, it would not have been feasible to provide them for several thousand students spread around the country (even now, the economic argument is finely balanced). Instead, the two existing technologies of mainframe computers and television were combined to provide students with the chance to view computer graphics in their own homes.

Compared with the facilities now available for residential students at other universities, this arrangement has both an advantage and a disadvantage. The disadvantage is that the graphics are not interactive: students cannot explore different avenues for themselves (although as partial compensation the animations are accompanied by an commentary, a voice explaining what is happening at each stage and king one of the roles of a teacher in a more conventional situation).

The advantage is perhaps less obvious: the technique allows for moving computer graphics, similar to those effects known for years in the film industry as "animation". The first OU computer animations were broadcast on television as part of the original Mathematics Foundation Course in February 1971. Since then, several courses in pure and applied mathematics and in statistics have used this technique in their television programmes.

It ought to be said straight away that some modern microcomputers can provide a rudimentary form of animation in their graphics, but that this is entirely inadequate for many of the applications considered in this paper. While these machines are excellent for many purposes, there are inherent technical reasons why they cannot provide more than a limited amount of movement. However, within the past three years, microcomputers have also made their appearance in OU television programmes, although in a complementary role to traditional computer animations. One particular use has been in courses on probability and statistics, where simulation has been used to illustrate many ideas, and where the limitations of micro-computer graphics capabilities have been small compared with the advantages of real-time calculations using random number generators. An instance of this has been in a programme about the modelling of conflict as a Poisson process[2], where the microcomputer can be used both to provide a visual display of the conflict (in the manner of many modern computer games) and also to provide a graphical depiction as a two-dimensional random walk. This use of computer graphics is much closer to the way microcomputers are used in schools and universities today, and in some cases copies of the microcomputer software have been made available to local OU tutors so that they can use the graphics interactively in their tutorial sessions. Of course, animations are more expensive to produce than microcomputer graphics, and so the reason for choosing one technique or the other for any given topic in a television programme must be the importance of movement in the visualisation chosen to teach that topic.

2. SOME EXAMPLES OF COMPUTER ANIMATION

An obvious area where these techniques can be applied is in geometry. If geometry is defined as the study of properties which are invariant under rigid motions, then moving visual demonstrations can give a better intuitive feel for the subject than static diagrams. The need for this is particularly important when the geometry being demonstrated is non-Euclidean and therefore not part of the student's everyday experience. One OU programme[3] has used animation to explore the geometry of the Poincare disc, a standard model of a complete Riemannian 2-manifold of constant negative curvature. Geodesics in this geometry are (Euclidean) circles cutting the boundary circle at right angles; the animations demonstrate how small triangles (which appear curvilinear, of course) change shape and size as they move about the disc, becoming vanishingly small as they approach "infinity". The standard property of being able to take a fixed geodesic and a point not on it, and then of being able to construct many geodesics through the given point which do not intersect the original geodesic, also becomes more easily understandable.

Another obvious use is where three dimensions are involved. Here, there may (or may not) be movement in the animation which is intrinsic to the subject matter, but there is the additional opportunity of providing different points of view to give greater insight into the concepts being explained. An OU programme[4] explaining some of the ideas of catastrophe theory used this technique to good effect. An initial simple animation demonstrated the motion of the equilibrium points corresponding to a quartic potential function, where the position of those points were given by the roots of a cubic polynomial. Subsequently, a family of different cubic graphs was used to draw out the three-dimensional cusp catastrophe surface, and the reason for the name of this surface became immediately apparent by continuously moving the point of view to a position vertically above the cusp. The particular way in which the surface was drawn (in terms of almost parallel curves), and the perspective effect created by positioning the viewpoint a finite distance away from the surface, created a very clear relationship between the fold in the surface when viewed obliquely, and the double-shading when viewed from above.

Perhaps a more straightforward use of computer graphics is to display graphs of functions, and where a family of functions depends on a real parameter then animation can be used to show the effect of changing the value of the parameter. Here, the time dimension is being used to replace the extra spatial dimension which would be needed if the family of functions were to be regarded as a single function defined on a subset of the plane. There have been many examples of this in past OU programmes: for instance a programme in the Mathematics Foundation Course[5] demonstrated how the coefficients of the Taylor series for a particular trigonometric function were "best" by looking at a sequence of Taylor approximations and varying the coefficient of the highest-order term in each approximation.

More recently an introductory course on probability and statistics has used this technique extensively. For example, several animation sequences have been used to illustrate one-parameter families of probability distributions, such as the Poisson distribution and the Binomial distribution with fixed sample size[6]. There seem to be two advantages to this approach. First, the visual impression of how the shape of the distribution depends on the parameter is much more powerful when changes in shape are linked continuously to changes in parameter value. If one of the initial objectives in teaching probability is to give a feeling for what particular distributions "are like" then this visual impression is important. But secondly, there are spin-offs in the associated teaching of statistics: the picture of a distribution changing continuously with a parameter fits neatly into the explanation of a confidence interval for

COMPUTER ANIMATIONS IN MATHEMATICS TEACHING
AT THE OPEN UNIVERSITY

that parameter, as shown by animations for another programme in the course[7]. The idea of choosing upper and lower confidence limits to give specified values for the corresponding tail probabilities becomes much clearer.

This same course also contains an example of three-dimensional animation [8] where movement in the animation is used both for exploring parameter changes, and also for comparing two-dimensional and three-dimensional representations. The context is an explanation of bivariate distributions and the idea of regression, and the animation first explores the effect of changing the value of the correlation coefficient in a bivariate normal distribution with equal variances. Subsequently, a particular bivariate normal distribution is sliced several times parallel to one axis, and the resulting curves suitably scaled so that they represent the corresponding conditional distributions. The regression line is clearly visible in this representation, as indeed is the effect on the regression line of changing the correlation coefficient.

A final example is from applied mathematics. A video-cassette made for a course on fluid dynamics[9] contains several animations demonstrating features of wave motion. One of these shows the effect of superimposing two waves of slightly different speeds and wavelengths: the result is easily seen to be a wave packet with a group velocity of about half that of the individual waves. In examples like this, the time passing while watching the animation really does correspond to a time parameter in the mathematical model, although there are opportunities for stretching or compressing time as required for the purposes of the explanation.

3. TECHNICAL DETAILS

As mentioned earlier, the technical details of the construction of computer animations are relevant for an understanding of why microcomputer systems generally aren't suitable for creating moving graphics.

The problem is best explained by means of an example. Imagine a screen showing a single vertical straight line segment. In most computer graphics systems, the line is made up of a number of illuminated points on the screen - these points are called "pixels". There are several ways in which the line segment can move continuously on the screen; compare a vertical translation (in the direction of the segment) with a horizontal translation. In the former case, a few pixels must be added to one end of the line segment and erased from the other end for each step of the movement; in the latter case, the complete line must be erased and redrawn at each step. There is a noticeable difference in the amount of computation required for the two types of movement.

The importance of this fact is that the visual impression of continuous movement is only given when the individual steps in the movement occur sufficiently rapidly. Typical rates are 24, 25 or 30 steps per second as used in films, European television, and American television respectively. A microcomputer graphics display is generated in real time, and so each stage of the movement must be calculated (or otherwise transferred to the display screen) within about 40 milliseconds. While this is possible in simple cases, the calculation time obviously depends on the complexity of the movement within the picture. All computers thus have limits to the complexity of the animations they can produce in real time, and for most microcomputers this limit is very low.

The technique used by the OU for its computer animations is called "stop-frame". With this technique, the time to draw each individual picture is now no longer a constraint. as each step of the animation is recorded on some other medium (film or videotape) offline. one frame at a time. The film or videotape is then replayed at normal speed to give the animation effect. Most early OU computer animations were recorded on film by outside agencies. using data from computer tapes generated by a variety of mainframe computers. Since 1983, however. a video recording technique developed jointly by the BBC and the OU has been used for this purpose. Data for the animations is still generated on a mainframe computer (a DEC 2060), but now a computer graphics terminal is used to display the individual pictures for the animation and a video signal from the terminal is fed to a professional videotape recorder. The operation of the videotape recorder is controlled automatically by the computer. and in general recordings take place overnight. Typically, one minute of animation takes four hours to record with the equipment at present employed, although the use of more specialised equipment would allow this figure to be reduced by an order of magnitude. However. it would still be true that the more complex the animation required. the longer would be the recording time.

In principle, there is no reason why such a stop-frame technique should not be used with a microcomputer rather than a mainframe. However. there are a number of practical difficulties. The microcomputer storage is unlikely to be large enough. and the resolution of its screen display - while perhaps adequate for static pictures - gives strange visual effects at inclined or curved edges which are particularly noticeable where movement is involved. This effect is called "aliasing" and specialised hardware is usually needed to minimise its effect. However, the most important practical difficulty is that the microcomputer display must be electronically synchronised with the mechanism of the video recorder. and that the latter must be capable of performing electronically controlled video editing. Again, these facilities are not normally provided in compatible form on commercially available equipment, and this is likely to provide an insuperable difficulty to the provision of an inexpensive microcomputer-based technique for the generation of animations in the near future.

4. DESIGNING AND PRODUCING COMPUTER ANIMATIONS

Since the production of computer animations requires the use of several items of expensive equipment, it is cost-effective to design them well in advance, rather than to let them evolve with experimentation. In fact, the same constraint is felt elsewhere in the preparation of OU course material. and the solution which has been adopted is to use a team of people who prepare drafts and provide constructive criticism. In the context of producing computer animations, this typically involves an OU lecturer and a BBC producer who in collaboration prepare a "storyboard" - a handwritten sequence of drawings and movement instructions which specify the animation. As with the specification of any other item of computer software, this document should ideally satisfy the requirements of being complete, consistent and unambiguous. In some circumstances. experiments with a microcomputer will have assisted the preparation of the storyboard. but only rarely will any computer-generated data be part of this document. In other circumstances. an experienced graphics designer will have been asked to give expert advice on screen layout.

Once an agreed storyboard has been produced, the animations are programmed using a commercial graphics software package. The package used on the OU DEC 2060 is called VGM (Virtual Graphics Machine). Still pictures from the animation can be seen to confirm that the required effects have been generated, and when the programming is complete the animation is recorded onto videotape.

Of course, once an animation has been recorded it is available for use in other contexts apart from the original television programme for which it was designed. While some animations just consist of small teaching elements which would not be intelligible without considerable explanation. others could form the basis of useful resource material which could be used in courses elsewhere. Indeed. elements of several OU courses have been used by other institutions in just this way.

In this respect, it is perhaps worth noting that computer animations are rather different from microcomputer graphics demonstrations. The latter can be produced fairly inexpensively in any organisation where there is the expertise in teaching and in computer programming (although the results will usually be more satisfactory if reasonable design procedures are followed than if the programs just evolve in the way that much microcomputer software seems to do). In contrast, the production of animations needs a much larger investment in hardware and consequently is only feasible where there are economies of scale in the use of the resulting product: they are like books rather than lecture notes in that they would tend to be produced centrally and distributed by one means or another to the places where they would finally be used. The combination of a university computing service and a television production centre on one campus at Milton Keynes has been instrumental in matching teaching requirements with the technical means of satisfying those requirements.

5. CONCLUSIONS

Computer animations were originally used in OU mathematics teaching as a way of providing (on television) accurate diagrammatic representations of certain mathematical concepts, with the use of movement being a way of exploring those concepts. Movement in animations is now seen as performing several different functions, and so the use of comparatively expensive stop-frame recording techniques using mainframe computers has been retained (although microcomputer graphics are employed for simpler tasks).

In the future, as technology advances even more rapidly, there are two directions in which these techniques might progress. First, the power of small computers may become sufficiently great to allow them to generate animations in real time, and thus provide "interactive" animations. Of course, the complexity of such animations would be inherently limited for the reasons described earlier, and so such developments should perhaps be viewed as an extension of the microcomputer graphics techniques which are already in use. Alternatively, the availability of such devices as videodiscs could allow the interactive use of pre-recorded animations. Both possibilities suggest that some interesting avenues in the use of computer-based technologies for mathematics teaching could soon be awaiting exploration.

REFERENCES

1. M101 "Mathematics Foundation Course": The Open University 1978
2. M245 TV8 "Conflict" (in M245 "Probability and Statistics", OU 1984)
3. M203 TV22 "A Non-Euclidean Geometry" (in M203 "An Introduction to Pure Mathematics", OU 1979)
4. M101 TV32 "Mathematics, a Social Perspective: Catastrophe Theory" (in M101 "Mathematics Foundation Course, OU 1978)
5. M101 TV14 "Taylor Polynomials" (in M101 "Mathematics Foundation Course, OU 1978)
6. M245 TV3 "A Probability Model for Rare Events" (in M245 "Probability and Statistics", OU 1984)
7. M245 TV6 "Confidence" (in M245 "Probability and Statistics", OU 1984)
8. M245 TV10 "Regression" (in M245 "Probability and Statistics", OU 1984)
9. MST322 VCR (in MST322 "Mathematical Methods and Fluid Mechanics", OU 1984)

What Happens When You Switch Off The Machine?

John H. Mason
Centre for Mathematics Education
Mathematics Faculty
Open University

My aim is to set the question posed in the title in context, to argue that it is at least as important a question as finding or designing appropriate software, and to make some suggestions which apply equally well to other modes of pupil-teacher-content interaction.

Context

Computers open up a plethora of possibilities for students of mathematics. As everyone points out, rapid and complex calculations no longer present a barrier to exploration, and colourful graphics can bring the dynamic of mathematical ideas to life. There are two major caveats however. The very power to do computations invites us to 'compute first and think second'. It is terribly easy to be seduced into 'running off a few examples' before getting a sense of what the problem is really about. The second caveat is that a graphics sequence can be very engaging, to the extent of mesmerising. 'Screen fixation' is common, both in programming ('if I just change this line it ought to work ...'), and in program running ('this time I'll try ...and it's sure to get the answer'). The two caveats merge when we observe that the hardest button to press is the off-button. Once a program is up and running, the path of least resistance is to 'try another case', or 'let's see what happens if ...'. Such behaviour would be admirable if it involved conjecturing, but sadly it is more often curiosity spawned from reluctance to turn off the machine than anything else. One of the main forces for keeping the machine going is the sudden sense of let-down, of emptiness when the machine is switched off.

Although videotapes seem far removed from interactive computers, experience at the Open University leads me to suggest that the two media, interactive computers and videotape, have a lot in common. In particular, they share the sudden hiatus when turned off, as well as being powerful means for stimulating mathematical thinking. The kinds of videotape I have particularly in mind are animated sequences generated by computer, either in real time or frame-by-frame. Of course animations can be generated in other ways, and can show the use of physical apparatus and people talking. My main points apply to any use of videotape in the mathematics classroom, but for present purposes, I confine my remarks to animations.

Why Video and Computers are little used in classes

The classroom teacher, whether in school, college or university, when contemplating any class activity, must ask themself 'Why am I doing this?', and 'What are my students supposed to get from it?'. These questions are not as easy to answer as they might seem, and I put forward as evidence the fact that virtually all mathematics teaching to students over 16 is done in Lecture-Tutorial or Exposition-Exercise modes. I submit that one strong contributing factor is that 'telling' people things is easy, especially if it is ad-lib or from self-prepared notes. As soon as something prpeared by others is involved, life becomes more difficult and less attractive. Other people's approaches are confining. This observation, goes some of the way to explaining the mountain of undergraduate textbooks, whose prefaces read in part 'I was unable to find any suitable text ...'. The same phenomenon explains the absence of preprepared video tapes and computer programs from most lecture theatres and tutorials. Where electronic teaching aids are being used, they are usually home-grown. The lecturerer is committed to their use, having participated in the design, and so feels comfortable integrating them into the lecture.

A second major force against the use of 'teaching aids' is the hiatus referred to earlier, when the device is turned off. It takes a moment or two to switch modes, to come back from the imaginative inner world generated by animated visuals, and to re-enter the world of the classroom. It requires movement from a reactive, trying-to-make-general-sense-of orientation, to asking and trying to answer specific questions; in short, from passive to active mode. Even in the case of interactive programs which appear to require active participation, there is nevertheless a strong force to react to the machine rather than to drive the machine.

I believe that academic solipsism, as described above, is not the real force against the use of teaching aids, but rather a symptom. The main force is a mixture of fear and ignorance, bolstered by a narrow view of what learning and doing mathematics involves. The fear stems from uncertainty as to how to cope with new media, and particularly how to make transitions into and out of exposition, from screen to lecture. The ignorance is of other ways of engaging with students apart from lectures and exercise classes, and particularly in the presence of computer animations and programs. Many lecturers seem unaware of the many different ways of working with a class, and because they have few options available, they have little choice in the moment. At the heart of it all lies a basic epistemology in which students are told things and then go away and learn them. This worked when only the brightest of students made their way to colleges, but it is not appropriate in the present climate of more open college and university admissions and universal secondary schooling.

One of the most powerful reasons for making use of electronic media, apart altogether from their contribution to dynamic imagery, is that they enable the teacher and students to face the same direction, literally. All other modes of classroom interaction involve confrontation between teacher as expert disciplinarian and source of rewards, nad students. With a screen to watch and then work on, studnets and teacher are trying to make sense of some other agent. It is then much easier for the teacher to abandon the 'know-it-all' role, and to work with the students on what they recall from the screen.

So, having shown someone something on a screen, what do you do when it is finished? People who focus on interactive computers may decry the loss of interaction when moving to video, and may wish to reply to my question with 'let them interact with the program, of course!'. But what happens when they have finished 'trying' things out? What happens when the machine is turned off? It is not enough to say that the mere watching, or even watching and interacting, is sufficient. All the evidence suggests the contrary, that students remember very little from watching television programs or from using other people's computer programs. Indeed, they often remember little from lectures and tutorials either.

Some Suggestions

The suggestions I wish to offer, which I know from my own experience can increase the range of options available to a lecturer, and which in particular can help bridge the gap when the screen goes blank, are based on the assumption that students have plenty of experience of working through exercises, and getting through lectures and tutorials, but little experience of working on such events. Thus they have little in the wayof study learning techniques to bring to bear on media such as videotapes, computer animations and programs. The following suggestions are designed to help with this.

Some Suggestions

1. Reconstruction

It is a common but false adage that we learn from experience. If experience teaches us anything at all, it is that we rarely learn from experience unless we make some specific effort. After expending considerable effort to work through a sequence of exercises, it might be expected that students have 'picked up' what the author intended, namely, general principles ricly supported with examples. In fact, what usually happens is that students work each exercise independently, with considerable support from the answers if supplied, and often fail to make any connection at all between different questions. Thus, the student, embroiled in the particularities of individual exercises, fails to appreciate the underlying generality. The same thing happens when viewing videos, and when fiddling with computer programs.

Reconstruction is a very simple technique which can make a great deal of difference, both to students' appreciation of the immediate event, and to their study habits as a whole. Students are asked (working collectively or individually) to write down all the technical terms encountered in a sequence of exercises, in a topic or in a mathematical event. They are then asked (working individually or in pairs, in their head or in writing) to weave the terms together into an account of what the exercises/topic has been about. This account can then be aired publicly, and modified as necessary by comments from colleagues.

The role of the teacher in reconstruction is as listener, and as monitor and chairperson, not as source of correctness. The students can sort out most difficulties themselves, and only in a last resort is it necessary to regress back to exposition. Once students have learned to work this way, they can do much of it outside of the class, though I maintain that far from 'using up too much time', reconstruction actually increases the efficiency of the teacher. By listening respectfully and closely to what students have to say, a teacher can support the students' own construction of sense and meaning, and if necessary, can use what they hear to pitch much more accurately any subsequent exposition.

The principle behind reconstruction is that when I have constructed my own account, it is mine, reconstructable at a future date. As long as it remains someone elses account, it is lodged in retentive memory and is liable to fade out or disappear altogether.

Reconstruction also applies to videotape viewing and computer programs. If there are few or no technical terms available, students are asked to recount (collectively or in pairs) the fragments that they can recall vividly. Then they try to weave these together into an overall account.

The advantage of working in a group is that students can support each other, and discover that their colleagues also have a sparse memory, yet, an overall account <u>can</u> be put together with a little cooperation. It also demonstrates a study technique that many students seem never to discover: using spare moments to tell yourself a story about each topic that is studied, so that it is your own story and not someone elses.

2. The use of small groups

It is important for students to take every possible opportunity to try to express what they understand to others. Unfortunately there is often little opportunity provided for this, and many students do not discover its importance. It is rather odd, really, considering how important it is to mathematicians to have a friendly colleague who will listen, and cooperate in a conjecturing atmosphere. By contrast, the atmosphere in many classrooms is one of competition to be first, and to be judged right or wrong by an 'expert'.

Getting students to work every so often in pairs, perhaps for only a few seconds, rehearsing something that has been said, not only demonstrates the importance of talking ideas over, but also quickly reveals misunderstandings which the teacher can then pick up. It is not necessary that the teacher monitor everything that students say - that clearly would take too long. The beauty of brief discussion in pairs is that everyone has a chance to try to say something. Consequently more people are willing to say something publicly, and confusions manifest themselves earlier and more quickly.

3. The role of Imagery

The idea of a function is critical in the mathematical topics to which 16+ students are subjected, and it is widely admitted that many students have difficulty with f(x). The twentieth century view of functions is as a process, a mapping, a transformation, a movement. Despite changes in the primary and secondary curriculum, this sense of function is often missing. Students are trapped in the older perspective of function as image. Computer animations are an ideal means for suggesting movement. They can do it in a way which is virtually impossible for all but the most accomplished of showmen in the lecture theatre.

What animations can do is provide the seed of an inner sense, perhaps even an image of functions as process. Functions are of course just one example, albeit fundamental. A few examples from Open University programmes include

The dynamic generation of the sine, cosine and tangent as projections of a circle.

The conics as a continuous family of curves, not just as sections of a cone, but as loci as well.

Probability distributions with one or more parameters varying.

Solutions to differential equations with one or more boundary values varying.

Taylor and fourier approximations shown dynamically as making the best fit at each stage.

In each of these cases the accomplished mathematician probably has her own form of awareness of how the various cases fit together. Unless she speaks explicitly and graphically <u>from</u> that awareness so that an image is generated spontaneously by each student, few students will make their own links except in a haphazard way. Without occasional specific reference to imagery, or to 'getting a sense of', many students may never realise that mathematics takes place in the head and not at arms length on paper. Dynamic images can help a majority of students make links. Note however that just watching an animation usually has little impact. Something like reconstruction is needed in order to assist students to work on the images, to make them their own.

Students need to become aware that imagery or some inner sense is critical for success in mathematics. Unfortunately many teachers do not themselves have a developed sense of the role of imagery. They speak not directly from their images, but rather use the images to inform their speech. For some students this is adequate. For most it is not. Students and teachers alike could usefully attend to the act of talking to someone else making direct and explicit reference to their imagery.

I take the view that mathematics is the formalisation of awareness of relationships. The diagrams and symbols are not the objects of mathematics, despite the widely held student opinion to the contrary. They are merely pointers to, or indicators of some awareness. Put another way, the awareness is what is invariant behind all the different articulations. Perception of invariance is a sophisticated act involving search for stability and commonness, and expressed as generality. Far more students could succeed at this if teachers themselves worked on becoming aware of the role of imagery in their own thinking.

<u>Summary</u>

Concentration solely on production and distribution of software/videotapes will result in lovely materials that sit on shelves, because academics are not good at using other peoples' materials, and because of the uncomfortable hiatus when the screen goes blank. If such materials are to be used effectively, then students and teachers must adopt appropriate ways of working which foster and enhance mathematical thinking. Both teachers and students will need training in techniques such as reconstruction. For training to be effective, teachers have to wish to question and change their ideas of what learning and teaching mathematics are about. There is some hope that the electronic media will act as a force for change, if only because they enable the teacher and the students to face in the same direction rather than facing each other.

I have suggested a way of working which applies equally well to computer animations, whether run in real time or on videotape, to interactive programs, to lectures, tutorials and exercises - in short, an aspect of learning and doing mathematics which seems to be missing in many classrooms.

Sources

The ideas expressed in these notes are a fresh articulation of concerns which can be found in philosophical and educational writers from Plato to the present time. What is important is to find a way of speaking which resonates with today's teachers, rather than establishing an academic pedigree.

The Automath mathematics checking project and its influence on teaching

by N. G. de Bruijn

1. Automath

Computers influence mathematics in many ways. One of these lies in the fact that we can learn to explain mathematics to a computer, and in this process we may learn about how to organize mathematics and how to teach some of its aspects.

At the Technological University Eindhoven (Eindhoven, the Netherlands) the project Automath was developed from 1967 onwards, with various kinds of activities at the interfaces of logic, mathematics, computer science, language and mathematical education. Right from the start, it was directed towards the presentation of knowledge by means of symbolic manipulation, with the possibility to leave much of the work to a computer, with quite a strong emphasis on doing things in a humanly way. One might say that it is a modern version of "Leibniz's dream" of making a language for all scientific discussion in such a way that all reasoning can be represented by a kind of algebraic manipulation.

The basic idea of Automath is that the human being presents any kind of discourse, how long it may be, to a machine, and that the machine convinces itself that everything is sound. All this is intended to be effectively carried out on a large scale, and not just "in principle".

This paper does not intend to describe the Automath system in any detail, but rather to explain a number of goals, achievements and characteristics that may have a bearing on the subject of the ICMI discussion. The paper is definitely not trying to sell Automath as a subject to be taught to all students in standard mathematics curricula. The claim is more modest: as Automath connects so many aspects of logic, mathematics and informatics, it may be worth while to investigate whether the teaching of mathematics could somehow profit from ideas that emerged more or less naturally in the Automath enterprise. The idea of Automath is to "explain things to a machine". Students are no machines and should be approached in a different way. But as teachers we should know that if we cannot explain a thing to a machine then we might have difficulties in explaining it to students.

1.1. A basic idea of Automath is to write in the form of a complete book, line by line. A computer can check it line by line, and once that has been done, the book can be considered as mathematically correct.

1.2. As a starting point we think of a book written entirely by human beings. Later on we may think of leaving part of the writing to a machine. That part might be simply tedious routine work, but also possibly the more serious problem solving (i.e., "theorem proving", a branch of artificial intelligence).

1.3. We should make a clear distinction between the Automath system and Automath books. The system consists, roughly speaking, of language rules and a computer program that checks whether any given book is written according to those rules.

The system of Automath is mainly involved with the execution of substitution, with evaluation of types of expressions, and comparing such types to one another. It is very essential that everything that is said in a book, is said in a particular context: the context consists of the typed variables that can be handled, but also of the list of assumptions that can be used. The system keeps track of those contexts.

The Automath system does not contain any a priori ideas on what is usually called logic and foundation of mathematics. Any logical system (e.g., an intuitionistic one) can be introduced by the user in his own book, and the same thing holds for the foundation of mathematics. In particular, the user is not tied to the standard 20-th century set theory (Zermelo-Fraenkel). And the user can choose whether to admit or not to admit things like the axiom of choice. From then on, the machine that verifies the user's book will be able to do this according to the user's own standards.

1.4. In an Automath book, logic and mathematics are treated in exactly the same way. New logical inference rules can be derived from old ones, just like mathematical theorems are derived, and the new inference rules can be applied as logical tools, in the same way as mathematical theorems are applied.

1.5. Writing in Automath can be tedious. All details of arguments have to be presented most meticuously. At first sight this might be very irritating. The questions are (i) whose fault this is, and (ii) what can be done about it.

The questions are related. Part of the negative impression that the length of an Automath book makes, is due to the fact that no attempt was made to "do something about it" at the stage of the design of the general system. This is based on the philosophy that generality comes first, and that adaptability to special situations is a second concern.

The reason why Automath books become so long is that we claim to be able to handle all usual mathematical discourse, but the mathematician has more in his mind than he explains. Perhaps we may say that part of mathematical work is done subconciously. Mathematicians have a vast "experience" in mathematical situations, and such experience may give a strong feeling for how all the little gaps can be filled. Possibly much of the experience is consulted subconciously "on the spot".

Moreover, mathematical talking and writing are social activities. In every area, people talk and write in a style they know they can get away with. Some poor or incomplete forms of discourse are so wide-spread that it seems silly to bother about improvements; certainly it is not a very rewarding task to try.

The answer to question (ii) is that very much can be done about it indeed. But just like every user can write his own book under the Automath system, he can implement his own attachments to the system. This may involve special abbreviation facilities, but also automatized text writing, producing packages of Automath lines by means of a single command, in cases where there is a clear system behind such a package.

1.6. Are computers essential for Automath? Not absolutely. The computer sets the standard for what the notion "formalization" means. If we cannot instruct a computer to verify mathematical discourse, we have not properly formalized it yet. In the standard form, the author of an Automath book has to write all the symbols one by one, and since he knows that what he writes is correct, he would also be able to check it by hand.

Nevertheless humans make mistakes. Automath books have been written with a number of characters of the order of a million, all typed by hand. It is hard to guarantee correctness of such a text without the help of a modern computer.

1.7. As the Automath system has no a priori knowledge of logic and set theory, it can be used to write in a style that might be more natural than what we see in other formalizations.

There is a wide-spread idea that propositional logic comes down to manipulating formulas in a boolean algebra, a kind of manipulation that is either carried out by handling formulas with the aid of lists of tautologies (in the same way as one used to do in trigonometry), or by a machine that checks all possibilities of zeros and ones as values for the boolean variables. A very much better formalization lies in the system of "natural deduction". This is very easy in Automath. The boolean bit-handling propositional logic can be done in Automath too, but it is much more clumsy than natural deduction.

A second option we get from the liberty of using Automath in the style we prefer, is to give up the 20-th century idea that "everything is a set". There is the magic Zermelo-Fraenkel universe in which every point is a set, and somehow all mathematical objects are to be coded as points in that universe. The particular coding is a matter of free choice: there is no natural way to code.

Zermelo-Fraenkel set theory is quite a heavy machinery to be taken as a basis for mathematics, and not many mathematicians actually know it. An alternative is to take "typed set theory", in which things are collected to sets only if they are of the same type: sets of numbers, sets of letters, sets of triangles, etc. It may take some trouble to make up one's mind about the question what basic rules for typed set theory should be taken as primitives, but if we just start talking the way we did mathematics before modern set theory emerged, we see that we need very little. Anyway, in Automath we have no trouble at all to talk mathematics in a sound old-fashioned way.

Yet, if someone still wants to talk in terms of Zermelo-Fraenkel universe, Automath is ready to take it.

1.8. One of the advantages of Automath not being tied to any particular system for logic and set theory, is that we can think of formalizing entirely different things too, again in a natural style. As an example we may think of the algorithmic description of geometrical constructions like those with ruler and compass. Although it has not actually been produced, we may think of a single Automath book containing logic, mathematics and the description of ruler and compass constructions, with in particular the description and correctness proof (both due to Gausz) of the construction of the regular 17-gon. This description will be quite different from coding the construction as a point in the Zermelo-Fraenkel universe. We might even think of a robot equipped with ruler, compass, pencil and paper, who reads the details of the construction from the Automath book and carries them out in the way Gausz meant.

1.9. Many parts of science are patchwork consisting of pieces of theory, connected by rather vague intuitive ideas. Ever since the last part of the 19-th century it has been one of the ideas of the mathematical community that mathematics should be integrated: all parts of mathematics are to become sub-domains of one single big theory. The patchwork picture still applies to most physical sciences, but also to several parts of the mathematical sciences. One such part is informatics.

It seems to be a good idea to integrate informatics into mathematics, at least in principle. And, as in the case of geometrical constructions, Automath is a good candidate for describing this. It is possible to write an Automath book containing: logic, mathematics, description of syntax and semantics of a programming language, and particular programs with proofs that the execution achieves the solution of particular mathematical problems. One might even think of going further: description of the computer hardware with proof that it garantees the realization of the programming language semantics. Or directly, without the intervention of a programming language, that a given piece of hardware produces a result with a given mathematical specification.

Needless to day, this kind of integrated theory will always contain a number of primitives we have no proof for, but it will be absolutely clear in the Automath book what these primitives are.

1.10. One thing people like in Automath, and other people strongly dislike, is the way Automath treats proofs as if they were mathematical objects. This is called "propositions as types". As the type of a proof we have something that is immediately related to the proposition established by that proof.

One should not be worried about this. Automath does not say that proofs are objects, but just treats them syntactically in the same way as objects are treated. This turns out to be very profitable: it simplifies the system, as well as its language theory and the computer verification of books. A third case where things are treated as objects is the one of the geometrical constructions we mentioned in 1.8.

1.11. In standard mathematics, most identifiers are letters of various kinds, possibly provided with indices, asterisks and the like. And then there are the numerals, of course. We have learned from programming languages, however, to use arbitrary combinations of letters and numerals as identifiers, (with restrictions like not to begin with a numeral). We do the same thing in Automath, thus having the possibility to choose identifiers with a mnemonic value, like "Bessel", "Theorem137", "commutative". This certainly helps to keep books readable.

In contrast to programming languages, the Automath system does not have the numerals 0,1,...,9. One can introduce them as identifiers in a book containing the elements of natural number theory, taking "0" and "succ" (for "successor") as primitive, and defining 1:=succ(0), 2:=succ(1),..., 9:=succ(8), ten:=succ(9). After having introduced addition and multiplication, we can define things like thirtyseven:=sum(prod(3,ten),7), but the Automath system has no facilities to write this as 37. This decimal notation might be added as an extra (it is one of the possible "attachments" mentioned in 1.5).

1.12. One of the basic aims of the Automath enterprise was to keep it feasible. This has been achieved indeed: considerable portions of mathematics of various kinds have been "translated" into Automath, and the effort needed for this remained within reasonable limits. If we start from a piece of mathematics that is sound and well understood, it can be translated. It may always take some time to decide how to start, but in the long run the translation is a matter of routine. As a rule of thumb we may say there is a loss factor of the order of 10: it takes about ten times as much space and ten times as much time as writing mathematics the ordinary way. But it is not overimportant how big this loss factor is (it would not be hard to reduce it by means of suitable attachments, adapted to the nature of the subject matter). What really matters is that it does not tend to infinity, which happens in many other systems of formalizing mathematics. The main reason for the loss factor being constant is that Automath has the same facilities for using definitions (which are, essentially, abbreviations) as one has in standard mathematics. The fact that the system of references is superior to what we have in standard mathematics, makes it possible that the loss factor even decreases on the long run when dealing with a large book.

1.13. Another feature that makes Automath feasible is that we need not always start at the beginning: we can start somewhere in the middle, and if we need something that we have not defined, or have not proved, we just take it as a primitive (primitive notion or axiom) and we go on. We can leave it to later activity to replace all these primitives by defined objects and proven theorems.

This kind of tactics was often (about 30 cases) applied at Eindhoven by students (mathematics majors). It usually took the student not much more than 100 hours work to learn about the system, to translate a given piece of mathematics, to use the conversational facilities at a computer terminal, and to finish with a completely verified Automath book containing the result. In order to give an idea of the subjects that had to be translated we mention a few: (i) The Weierstrasz theorem that says that the trigonometric polynomials lie dense in the space of continuous periodic functions, (ii) The Banach-Steinhaus theorem, (iii) The first elements of group theory.

1.14. Of the more extensive books that were written in Automath we mention two. The first one is L.S. Jutting's complete translation of E. Landau's Grundlagen der Analysis. In order to test the feasibility of the system, the translator kept himself strictly to Landau's text, rather than inventing some of the many possible shortcuts and improvements that would make the translation easier and shorter. The second one we mention here was by J.T. Udding, who wrote a new text with about the same results, much better suited to the Automath system, both in its general outline and in its details. The gain over Landau's text, in space as well as in time, was roughly 2.5.

1.15. One of the ideas of the Automath enterprise was to get eventually to a big mathematical encyclopaedia, a data bank, containing a vast portion of mathematics in absolutely dependable form. This is a thing that would take many hundreds of man years (thus far the Automath project took something like 40). But the idea is feasible. Most of the students mentioned in 1.13 used the Landau translation (see 1.14) as a data bank, and that way they added to the bank.

2. Standard mathematical language.

In close connection with Automath a language was studied with the same level of precision, but closer to ordinary language as written by mathematicians, at least when tney are very precise. Let us call it MV (for "mathematical vernacular"). MV is the familiar mixture of words and formulas in which some of the letters and formulas play a syntactic role just as if they were ordinary parts of a sentence, like subject, direct object, etc.

2.1. It is possible to formulate logic and the foundation of mathematics in terms of the grammar of such a language. The grammar of MV can be kept quite simple, since all sorts of idiom of natural language can be caught in terms of definitions in the book. This way we do not need to distinguish more then the following four grammatical categories: (i) sentences, (ii) substantives, (iii) names, (iv) adjectives. Each one of these four can occur as a group of words, but also as a mathematical symbol, a formula, or a mixture of words and formulas. The four categories correspond to the four kinds of definitions that mathematicians give. In the definitions of the first kind the new term is a sentence (like:"we say that p divides q if ..."), in the second case it is a substantive ("a square is a ..."), in the third one a name (... is called the n-th Bessel coefficient), in the fourth an adjective (a sequence is called convergent if ...).

2.2. The difference in syntax is not the only difference between Automath and MV. The main difference is that in Automath each line contains exactly all information about how the stated result follows from previous lines: all theorems and inference rules which are used are mentioned, and their role is made absolutely clear. In MV such indications do not belong to the language itself, but can be considered as having been written in the margin. In other words, in Automath they are language, in MV metalanguage.

One can use MV as a stage in the process of writing in Automath. If the steps in MV are small, and if the indications in the margin are sufficiently clear, the translation into Automath is a routine matter.

2.3. Inspecting textbooks in mathematics on school level one finds very little MV. Most of the texts are written in metalanguages of various kinds. Quite often, the intersection of the text with its own representation in MV is little more than the mathematical formulas, i.e. the part that was formalized hundreds of years ago.

3. Effects on mathematical education.

The question was: "How do computers and informatics influence mathematical ideas, values and the advancement of mathematical science?". There will be all sorts of influences, like the taste for constructivity, and, as far as education is concerned, the new possibilities to let students have their own stimulating discoveries with the aid of a computer. But the influence we get from the fact that we can explain mathematics to a computer, should not be forgotten. We shall look into this in some detail.

3.1. First, there are the philosophic aspects. Is it really mathematics we explain to a computer? Or is it just some piece of code we happen to interpret as mathematics? How arbitrary is our interpretation?

There is no definite answer to such questions. If we have to compare a formal system to something that is partly intuitive, then the comparison cannot be completely formal.

For example, in the partially intuitive mathematical world, the question whether the mathematical objects exist in a platonic reality, might seem to make some philosophical sense. But if we consider a completely formalized version to be explained to a computer, such a question cannot even be formulated. Some people will react by saying that this definitely puts an end to platonism, others will say that it shows that no formalization will ever be complete.

3.2. Having to phrase our mathematics in a very definite language, we have to make clear what part of ordinary mathematics belongs to the language and what part is metalanguage. Many paradoxes arise just by confusing language and metalanguage. Making the distinction will certainly help to understand mathematics better.

3.3. Today, most mathematicians have the idea that the foundation of mathematics is too hard to learn for a non-specialist, and can only be taught to students who know mathematics already. This means that the foundations of the building of mathematics are laid only after the building is completed, so they can impossibly play the role of the basis of mathematics. The teaching of the foundations at that late stage assumes the students to be acquainted with mathematical ideas (the role of definitions, axioms, theorems) for which one expects the foundations to give explanations. On a lower level, the same thing happens in the boolean propositional calculus: it is a mathematical system which is erected by standard mathematical techniques, and nevertheless it is a popular belief that it can explain what logic is, what proofs are.

3.4. Outsiders would be very surprised to hear that mathematicians are so vague about their own foundations, even now, towards the end of the 20-th century, that great century for logic.

If one really takes the task seriously to write (like it can be done in Automath) the foundations of mathematics up to a level such that the working mathematician would be able to build on it, one will see that it is not at all that hard. A sound basis can easily be given at the age of 17 to 19. For many questions about the relation between mathematics and computers (questions like program correctness) it is very essential to have such a basis.

Of course, the basis need not be given itself in a formal language. It can be quite informal, but the teacher should know the formal background.

The method of natural deduction is a very good candidate for explaining the foundation of mathematics. It opens the possibility to treat the introduction and elimination rules of the propositional calculus in exactly the same style as those of the predicate calculus. Moreover, it can be pointed out to the student, by means of an informal metalanguage, what is a proof, an axiom, a definition, an assumption, a theorem. And it opens the way to understanding notions that cannot be properly explained at all on an informal basis. In this connection we mention the notion of existence, which has remained a mystery to many generations of mathematicians.

3.5. A foundations course at an early stage should be recommended. This is not only because of the computer; another important reason is the disintegration of the teaching of geometry.

Traditionally, school geometry used to give the initiation into mathematical reasoning. Other mathematical subjects used to train the art of calculation, not the art of proof. But geometry had its drawbacks: it was hard and unattractive to keep the reasoning pure, i.e., to remove every appeal to what we learn by observation of the physical world. In particular this refers to the matter of order on the line and in the plane. Another drawback was that quite often the arguments failed in some exceptional, often trivial, situations, and that these had to be treated separately. And a satisfactory treatment of the axiomatic basis was too difficult to be treated at school. And, lastly, the logical content was so limited: no predicate calculus, no quantifiers, apart from a few cases where sets played a role (the geometric loci). On the other hand, geometry showed a wonderful interplay between intuition and argumentation.

Possibly because of the drawbacks mentioned here, traditional school geometry was almost entirely discarded in most countries, and replaced by the study of "structures", called "new math". In these new subjects there was hardly a chance to train the art of proof, and now we are left with the sad situation that upon entry of the university the students, even mathematics and computer science majors, are very weak in this respect.

3.6. In many parts of the new math, in particular in algebraic areas, it is quite hard to draw the boarderline between mathematics and metamathematics (cf. 2.3). And reasoning about sets, with or without Venn diagrams, is often on a low logical level. In particular, it gives hardly any opportunity for handling variables. It has to be admitted that the innovations in mathematical education have given us quite some progress, both in insights as in practical applicability, but the price we paid by neglecting the art of proof may have been too high.

3.7. Mathematics majors on the university level usually learn to handle predicate calculus in courses on the foundation of analysis. At least they learn it implicitly, on a practical basis, and directly tied to the formalization of notions with an intuitive background, like uniform convergence.

Needless to say this kind of material will become gradually harder now that the students enter the university with such a poor preparation in the art of proof.

Another matter is that it is no longer clear whether informatics students should take courses in the foundation of analysis. There is a danger that in the near future the only intersection of the curricula for mathematics and informatics will be some kind of simple calculus.

3.8. As to teaching the art of proof, it may be a good idea not to tie it to geometry, and not to any new subject like combinatorics, set theory or algebra, but to take it as a subject in its own right, in the form of an elementary logics course.

As a kind of experiment such a course was tried for computer science students, right from school, at the Technological University Eindhoven since 1982. It seems to have been successful in teaching the structure of proof by means of explaining the rules of the game of propositional and predicate calculus. The basis was natural deduction (cf. 3.4). Only after the building of logic was erected, it was shown how the notion of valuation gives the link with the boolean algebra aspect.

The course started with a chapter on syntax, involving the study of parentheses, representation of formulas as trees, infix notation, bound variables, lambda calculus notation, substitution, etc. It turned out to be illuminating to take the trees as the central theme, in particular in connection with substitutions in formulas with bound variables.

In the treatment of predicate logic, predicates were taken to be defined on sets, and in that respect the course took a naive point of view. It was not attempted to develop the language of mathematics in all its glory: that would probably have taken twice as much time as could reasonably be devoted to the course.

This introductory course on logic took not more than 18 hours teaching, with about 24 hours added for excercises.

In a sequel of this course (again 18 hours teaching plus excercises), applications were made to mathematical fundamentals (treatment of sets and mappings, the system of natural numbers, the method of induction, recursion and definition by recursion), but also to a number of subjects on the boarderline of mathematics and informatics. These were mainly: the terminology of the free monoid and its relation to language, contextfree grammars in a mathematical setting (with terminals and non-terminals), and the relation of this with the Backus-Naur form.

3.9. A course like the one described in 3.8 might be recommended as the body of the intersection of the curricula of mathematics and informatics.

What might be added to the intersection is a mathematical description of what is a computer, a program, input, output, program specification and program correctness. At that stage it is better not to go into details of a programming language, apart from the description how such languages can be defined by recursion.

3.10. Parts of the logics course, like syntax and propositional calculus in natural deduction, might be shifted to the school age (16-18 years). The natural deduction would be very appropriate for showing what a proof is, and it would raise the teaching of logic above the "trigonometry level" (cf. 1.7). And lambda calculus might really help to make school mathematics easier.

3.11. Some of the material mentioned in section 2 was taught at Eindhoven since about 1977 in a course called "Language and structure of mathematics", for those mathematics majors who wanted a teachers certificate in mathematics. Much of it would be fit for all mathematics majors at an early stage of their university career.

References on Automath:

N.G. de Bruijn, A survey of the project Automath.
In: To H.B. Curry: Essays in combinatory logic, lambda calculus
and formalism, Academic Press 1980.

L.S. van Benthem Jutting, Checking Landau´s "Grundlagen" in
the Automath system. Mathematical Centre Tracts nr.83,
Amsterdam 1979.

Symbolic Manipulators and the Teaching of College Mathematics: Some Possible Consequences and Opportunities

Kenneth D. Lane
Department of Mathematics
Colby College
Waterville, ME 04901*

ABSTRACT

The widespread availability of computer systems possessing some of the most cherished skills of our undergraduates is fast approaching. There now exist systems that factor, differentiate, integrate, solve, and what-have-you; all symbolically. It is arguable that mathematics curricula have, for the most part, ignored the existence of digital computational power. Will symbolic manipulative power also be ignored?

The author outlines an experimental mathematics curriculum under development at Colby College which makes use of symbolic manipulators. Specific examples of the integration of the technology into the curriculum are presented.

The proper inclusion of this technology in mathematics curricula raises some difficult issues. In June of 1984 a small group of mathematics educators gathered at Colby to study Colby's experience, to experiment with the capabilities of several systems, and to discuss the implications for further curricula considerations. The author reviews some of the issues that arose during this meeting and offers suggestions for their resolution.

Current Address: Department of Mathematics
Harvey Mudd College
Claremont, CA 91711
(USENET scgvaxd!ymir!klane)

Symbolic Manipulators
and the
Teaching of College Mathematics:
Some Possible Consequences and Opportunities

With a full page ad in The American Mathematical Monthly [1], Symbolics Inc. announced that the powerful computer algebra system MACSYMA is "available to colleges and universities at special rates." The ad goes on to say: "It can simplify, factor or expand expressions, solve equations analytically or numerically, differentiate, compute definite and indefinite integrals, expand functions in Taylor or Laurent series, compute Laplace transforms, ... " On the basis of the example given in the ad, the system seems to be able to solve problems that one would not consider attempting with paper and pencil. We note with interest that this software will run or a mini-computer.

MACSYMA is only one of several comparatively inexpensive computer algebra systems that are available for mini-computers and microcomputers. Several years ago Wilf [2,3] wrote articles speculating about life in a classroom in which all the students have this manipulative power at their fingertips. As so often seems to happen in this age of rapidly advancing technology, these science fiction like circumstances are fast approaching.

How, if at all, should mathematics educators react to these developments? This revolution in symbolic manipulative power is not unlike the corresponding revolution in digital computational power. Cheap digital computational power has been available to the masses for some time with little or no corresponding adjustments in how mathematics is being taught. Is it in the best interests of mathematics education to ignore the corresponding revolution in the availability of symbolic computational resources?

In this paper we indicate a curricular environment into which symbolic manipulation is being incorporated. After so doing, we identify some of the issues, challenges, and opportunities surrounding these developments. Hopefully, by offering comments with which few will agree a much needed discussion will ensue.

The Colby Curriculum Project

Colby College is a highly selective traditional liberal arts college located in central Maine. The introductory mathematics curriculum at Colby must serve three broadly defined, overlapping constituencies: 1) students preparing to take advanced courses in mathematics, 2) students preparing for nonmathematics courses with mathematical prerequisites, and 3) students who want experience with mathematics as a major component of the body of liberal arts. Frequently all of these groups are represented in introductory courses. At a liberal arts college, all courses exist for the purpose of education rather than training. The courses must therefore focus on the concepts and processes of mathematics, rather than on routine manipulations.

Events and developments of the past several years have brought pressure for reform and change in the undergraduate mathematics curriculum. The evolution of computer science and related equipment, the mathematization of subject areas outside of the physical sciences, and the gradual redefinition of student skills and preparedness are important component forces in the production of these pressures. In June of 1983 the Colby College Mathematics Department was awarded a grant by the Alfred P. Sloan Foundation "for the development of a new curriculum for the first two years of undergraduate mathematics in which discrete mathematics will play a role of equal importance to that of calculus" [4]. An important aspect of this curriculum was that it was to reflect the existence and widespread availability of computing machinery and computational systems. Computer based experiences were to be provided to motivate and illustrate concepts whenever appropriate. Throughout the proposed curriculum, the existence of new technologies and its effect on the relative importance of subject matter was to be considered.

During the development of the proposal it became obvious to those involved that the calculus curriculum would have to be substantially revised. Equal time for discrete mathematics seems to imply that the standard three semester single and multivariable calculus sequence would need to be distilled into a single one year course. Not only would the syllabus for such course be different, but the spirit of the course needed to be radically altered.

The final design of the curriculum called for three one-year courses to be developed. Of these courses, one was to combine linear algebra and discrete mathematics and was to be taken by students before taking a calculus course. A similar course was to be developed to follow a one-year calculus course. With this model students could take either one or two years of mathematics and if they take only a single year it can be either calculus or discrete mathematics. (It is actually possible to take a semester of each.)

<u>A New Calculus Course</u>

We decided to begin the project by carefully constructing a new calculus course. A textbook [5] was written and the course was first taught in the fall of 1983. Two aspects of the course are somewhat novel: (1) Single and multivariable topics were to be done concurrently, and (2) Symbolic manipulators were to be incorporated.

The incentive for a fresh approach to calculus comes not only from the pressure of introducing discrete mathematics but also from a sense of dissatisfaction with the lack of success of the traditional course in teaching processes and methods of mathematical thinking. Students often feel that the real meat of the course is the computation of derivatives, integrals, powers series, or any of the other manipulative activities. Attempts to get students to focus on analysis and synthesis often end in failure. If the instructor focuses on the ideas of calculus he/she is viewed as being hopelessly and inappropriately hung up on useless abstractions or irrelevant garnish. (After all, we just want the answer!)

It is no small wonder that many of our students have these attitudes. Consider the daily exercises and examination questions that students ultimately use to gauge the relative importance of topics in the course. Student perception of the importance of computational activities is reinforced by the fact that these assignments are often difficult and take a lot of time to complete.

In order to address these difficulties we identified three "Fundamental Processes" of calculus. As topics were woven together to design the course the prime consideration was the reinforcement of these fundamental issues. Somewhat arbitrarily the Fundamental Processes were identified as:

(1) Approximation

(2) Transformation

(3) Comparison.

In order to be able to spend more time in the classroom developing ideas, the Colby group decided to introduce the symbolic system MACSYMA in the experimental calculus course. The hope was that time normally spend on developing computational skills, such as techniques of integration, would be greatly reduced if this technology was made available to students. MACSYMA was an optional tool in a section of the calculus course taught during the 1983-84 academic year. Students were not given extensive instructions in the use of the software and any such use was strictly optional. During the fall 1984 term, students in the experimental calculus course were required to use the symbolic system MAPLE for a variety of exercises.

Student reaction to these systems was varied. Some were genuinely enthusiastic while others were reluctant to incorporate the system as a helpful tool. In all cases, the software ran on a VAX 11/780 under Berkeley 4.2 UNIX. Terminals were available to the students virtually around the clock and the software could be accessed by anyone with a UNIX account. We will discuss the performance of both MACSYMA and MAPLE in this time-sharing environment later in this paper.

Some Specific Examples

In the beginning we had only some "fuzzy" notion of how we might utilize compute algebra in the calculus course. Our plan was to proceeded cautiously by experimentation and to see what might develop! We have been genuinely surprised by much of what occurred and are excited about the prospects for yet more of the same. We now consider some examples to illustrate the possibilities.

Among the more interesting applications of derivative concepts is curve sketching. Although this activity ought to be perfect for illustrating and reinforcing newly learned concepts, I, for one, had been advocating that this no longer be included in calculus courses. After all, typically students have trouble learning the overall structure of the process since attempts to utilize the calculus tools usually abort when some simple computational mistake is made. What was hoped to be an illustration of the power of calculus ends up being an exercise in drudgery. Without having enough successful complete experiences with these techniques the overall picture is not grasped by the student. The intended goal in teaching the sketching techniques is lost in a sea of confusion over manipulation.

In the presense of a symbolic manipulator these exercises take on an entirely different character. The introduction of the symbolic system elevates the level of sophistication of this particular exercise. Consider what happens with a specific example.

Exercise: Sketch the graph of $f(x) = \frac{x^2-4}{x^2-1}$. Indicate all "interesting" features.

In what follows we illustrate a possible student session using MACSYMA to do this exercise. The basic format is for the system to prompt the user with (cn) and for the user to respond by entering a command. The system will then respond to the command in a line preceded by dn and give the prompt (c_). In the first command and response combination the student identifies the object equation and the system responds by "pretty printing" the expression.

(c1) (x^2-4)/(x^2-1);

(d1) $\frac{x^2-4}{x^2-1}$

(c2)

We might now compute the zeros of the expression.

(c2) solve(c1=0,x);

(d2) [x=-2, x=2]

Note that the student can refer to the equation in question as "c1". We might also want to compute the singularities of the expression. The next command strips off the denominator.

(c3) part(d1,2);

(d3) x^2-1

(c4) solve(d3=0,x);

(d4) [x=-1, x=1]

This is a calculus course - compute a derivative.

(c5) diff(c1,x);

(d5) $\frac{6x}{(x^2-1)^2}$

The task for the student is now focused on what particular question must be asked of the mathematical object at hand: What do you do with the first derivative? He/she may choose to compute the zeros and singularities of this expression.

(c6) solve(d5=0,x);

(d6) [x=0]

(c7) solve(part(d5,2)=0,x);

(d7) [x=-1, x=1]

The process continues in this fashion. The student must consider the information at hand and decide how to further process it. The student is forced to consider the relevant questions. The computation is discounted. Suddenly I'm interested in teaching curve sketching!

Our next example occurs in the study of Taylor's theorem. Again, it is difficult to communicate the usefulness of this powerful result because derivative computations require symbolic manipulation skills.

Exercise: Use Taylor's theorem to compute an estimate of sqrt(7) accurate to 0.001 and verify this accuracy using the theorem.

We might use the function f(x) = sqrt(x) and consider the polynomial centered at a = 4. A key to the exercise is the computation of

$$R(5,n) = \frac{|f^{(n+1)}(c)|}{(n+1)!}$$

We need to compute various derivatives of f(x) = sqrt(x) and estimate these derivatives for 4 < c < 5. This is the heart of the exercise. The real educational goal in teaching Taylor's theorem is to teach students how to estimate. In the traditional setting we seldom fulfill the goal because it is so difficult to reach this stage of the exercise. We begin with n=2.

(c1) sqrt(c);

(d1) sqrt(x)

(c2) diff(c1,c,3);

(d2) $\frac{3}{8x^{5/2}}$

Now, for c in the interval 1/c < 1/4 and we can overestimate the remainder by substituting 4 for c.

(c3) substitute(4,c,d2);

(d3) $\frac{3}{256}$

We now need only divide by (n+1)! = 6.

(c4) d3/6;

(d4) $\frac{1}{512}$

This estimate falls short of our requirement. We try n=3.

(c5) diff(c1,c,4);

(d5) $-\frac{15}{16x^{7/2}}$

Again, we can overestimate with c = 4.

(c6) substitute(4,c,d5);

(d6) $-\frac{15}{2048}$

Finally, if we include the factor 1/(n+1)! = 1/24 we get the desired accuracy.

(c7) d6/24;

(d7) $\frac{5}{16384}$

We know that a third degree Taylor polynomial will do and it is easy to compute.

(c8) taylor(c1,x,4,3);

(d8) $\frac{5}{8} + \frac{15x}{32} - \frac{5x^2}{128} + \frac{x^3}{512}$

Now we evaluate it at x = 5.

(c9) substitute(5,x,d8);

(d9) $\frac{1145}{512}$

Discussion

Our limited experience seemed to raise more questions than it answered. Fresh from the experiments of the 1983-84 year, the Colby group organized a small workshop to consider these issues. With the support of the EXXON Foundation and the Alfred P. Sloan Foundation we brought together two dozen mathematics teachers from colleges and secondary schools. During three days in June of 1984 the group was given instruction on the use of five systems: MACSYMA, SMP, MAPLE, REDUCE, and muMATH. The participants were given opportunities to experiment with each system after which discussion sessions were held.

In what follows I will raise some of the issues discussed by the group and offer some ideas for their resolution.

Why use the technology at all? Does mere existence imply that everyone should have one?

This question really deserves an answer and I feel quite comfortable with the following arguments. First of all, I think the technology deserves inclusion by virtue of its possible effect on student attitudes toward mathematics. Let me pursue this argument by way of example.

Consider the task of teaching students to properly handle the absolute value function. I can vividly recall from some years ago my first experience with this task. "All you have to do is to use the definition." Having imparted this wisdom to my class I felt confident of their immediate enlightenment. Of course, when asked to graph the relation $y = |x - 1|$ they bombed. Nearly all of them produced the graph of $y = x - 1$. The obvious question arose: "Why didn't they listen to me?"

The answer, I have come to believe, may be very simple. The students are taking a course, and I am teaching a course, but these courses may have almost nothing in common! To get the students to take the course that is being offered is one of the fundamental struggles of the experience. To most students mathematics is the study of using formulas and procedures. For these students to do mathematics is to compute, never asking what is being computed or why it is being computed. Concepts, ideas, and statements are not part of mathematics.

My limited use of symbolic manipulators in freshman calculus has been the single most effective weapon that I have ever found for combating these misconceptions. The presence of the symbolic systems serves to permanently alter the students' sense of mathematical value. If students are stuck doing all the gory computations there is no time to consider concepts and ideas. If the computational difficulty is removed what remains is the essence of the matter, the rationale for teaching the course in the first place.

There are other compelling reasons for using these systems. Most of us would agree that mathematics is a useful tool for solving problems and for helping to explain and understand the world around us. Why then, don't more people use mathematics to solve problems? One reason is that it requires a lot of knowledge and skill to use mathematics. If I might use an analogy, when computers were first developed, even if they had been widely available few people (even mathematicians) would have found them useful. They were just too hard to use. It required a lot of specialized knowledge to program early machines.

This situation is not unprecedented in mathematical history. Were it not for the invention of notational (read symbolic) devices a few centuries ago mathematics would be a subject that would be too hard for all but a few who practice it today. In the same manner the existence of expert mathematical systems will allow more people to use mathematical techniques for solving problems. This phenomenon ought not to go unrecognized.

Isn't it dangerous to rely heavily on these systems? What about students' hand manipulation skills?

Consider the student with little or no skills in hand symbolic manipulation. Suppose that this is a consequence of a curriculum based heavily on automated symbolic manipulation. There is the temptation to dismiss this complaint by pointing to the lack of skill exhibited by the traditionally trained students. We will resist the temptation and try to respond on a "higher" level.

It is difficult to disagree with the fact that students using computer algebra would leave a course with a different set of skills than they might have acquired in a more traditional curriculum. The real issue is whether or not they are missing anything essential. If we believe that symbolic manipulation power will soon be widely available then I think that it follows that the value of training large numbers of people to do sophisticated manipulation by hand may diminish.

No doubt there will be other skills whose importance will increase. The ability to use mathematics to describe the world will certainly be a skill requiring more of our instructional time. Again, the point is that mathematics and what is important in mathematics is changing rapidly.

Are these systems really ready for the classroom?

In some sense none of the systems reviewed are ideal for classroom use. MACSYMA, REDUCE, and SMP were designed for research and applications. They are extraordinarily powerful and require a lot of computing power. Performance of the VAX 11/780 system degrades noticeably in the presence of several active sessions of these packages. Even on an otherwise idle system workshop participants found it impossible to run as many as 10 computer algebra sessions concurrently. At most sites it is probably impractical to have an entire class use any of one of these three systems unless a couple of VAX 11/780 sized or larger machines were dedicated to symbolic manipulation.

The systems MAPLE and muMATH were designed (and are still under development) with the classroom more in mind. They are smaller and require less resources than the other systems but are still reasonably powerful. They could easily do the examples given here. Increasing demands on the Colby VAX have forced the use of MAPLE in the experimental course and there are certain activities for which it is very well suited. The system muMATH, which will run on a microcomputer is certainly capable of a variety of interesting exercises. But still, it would be nice to give students a system as powerful and friendly as MACSYMA without breaking the Computer Center budget!

The real future lies in microcomputer systems that would be capable of running software as powerful or even more powerful than MACSYMA. With the development of 32 bit microprocessors and cheaper memory chips affordable systems may be only a couple of years away.

One point of view that seemed to emerge during the workshop was that although the perfect classroom symbolic manipulator hasn't been developed yet, now is the time to start investigating how to best use what we have and to develop strategies for using what will soon be available.

What sort of specific work needs to be done to integrate symbolic systems into mathematics education?

It seems clear to me that you can't just give each student a symbolic manipulator and go on teaching the course like you always have taught it. It is necessary to rethink the purposes of assigned exercises as well as the types of exercises. Students can be given more complicated and meaningful exercises as well as exercises aimed at finding patterns. An experimental approach can be fostered. It takes a lot of imagination and experimentation to find suitable exercises.

There needs to be a broad discussion among mathematicians, the consumers of mathematics, and mathematics educators aimed at identifying the mathematical skills that will be important if symbolic manipulators become ubiquitous. We need to use our imaginations to develop ideas that will help to teach the skills so identified.

References

[1] The American Mathematical Monthly, Vol. 91, No. 2, 1984, page 159.

[2] Herbert S. Wilf, "The disk with the college education", The American Mathematical Monthly, Vol. 89, No. 1, 1982, pp. 4-8.

[3] Herbert S. Wilf, "Symbolic manipulation and algorithms in the curriculum of the first two years", in The Future of College Mathematics, (edited by Ralston and Young), Springer-Verlag, New York, 1983, pp. 27-40.

[4] John Hosack, Kenneth Lane & Donald Small, A Proposal for a New Curriculum, a proposal to the Alfred P. Sloan Foundation, 1983.

[5] Donald Small, John Hosack & Kenneth Lane, Calculus of One and Several Variables: An Integrated Approach, Colby College, 1983.

USING COMPUTER SYMBOLIC MATH FOR LEARNING BY DISCOVERY

by
David R. Stoutemyer
University of Hawaii
Honolulu

December, 1984

ABSTRACT

Computer symbolic math can help support mathematics education in numerous ways. However, the most exciting and easiest way is to encourage discovery via experiments that would be too tedious to perform manually. This paper presents a scenario of working on a project of this nature, then presents of list of analogous project ideas relating to elementary algebra, linear algebra, summation, power series and calculus.

1. INTRODUCTION

As outlined elsewhere [1], computer symbolic math, also known as "computer algebra", can play a major role in traditional computer aided instruction. This role can span from routine drill through sophisticated mixed-initiative tutoring systems that attempt to discover, model and repair student's misconceptions. However, effective realization of this potential will require lengthy collaboration between experts in computer algebra, computer-aided instruction systems, and math education. The potential is well worth this effort, but meanwhile there are other immediate ways of using existing computer algebra systems to support mathematics education.

The most exciting of these immediate opportunities is to use these systems to motivate students by exercising the process of **discovery.** By this I mean the experimental process by which mathematicians guess then prove new mathematics. Our traditional mathematics curriculum is so burdened with teaching useful known mathematics that little or no time is spent cultivating this experimental process. Indeed, experimental math was such a slow process in the past that devoting nonnegligible secondary school and undergraduate time to it was understandably regarded as impractical.

However, computer symbolic math systems now permit such rapid and flawless processing of nontrivial examples that it is easy to search for patterns which suggest conjectures and generalizations, then search for counterexamples or machine-aided proofs. With their rapid ability to process examples that are impractical to perform manually, these systems permit us to wander deeply and widely, following our curiosity as it is provoked by features that only large examples reveal. Moreover, the experience of working with the assistance of such a tireless brute-force assistant provokes curiosity about the underlying known mathematics too.

One can imagine the ultimate computer-aided educational mathematics project assignment being:

"Use a computer algebra system to discover something interesting, and submit a corresponding report."

However, most students are likely to need guidance and concrete suggestions -- at least initially. Accordingly, the purpose of this paper is to provide an example of such guidance and a collection of concrete project suggestions.

Before commencing such a project, the student should become comfortable with routine use of the system, including the protocol for invoking the system, for editing expressions and for accessing on-line or printed documentation. This does not necessarily include the use of "programming" control constructs such as procedures, loops, and conditionals. Most systems have a rich set of commands that are directly executable in a straightforward "calculator mode", and many explorations and projects require no more than a modest subset of these commands together perhaps with the use of assignment to preserve results for use in subsequent expressions. For example:

```
p: DIF (-93 x^4 y^3 + 439/2 x^2 y^2 - x y^5,  x);
```

might assign the partial derivative $-372\,x^3\,y^3 + 439\,x\,y^2 - y^5$ to the variable p, after which the command

```
FACTOR (p + 163308 x^4);
```

might produce the equivalent form $(372\,x^3 + y^2)(439\,x - y^3)$. Proficiency in such elementary use can be promoted by straightforward exercises such as

"Use the computer algebra system to factor $x^{16} - 64\,y^{24}$."

Without first witnessing a demonstration, most students are unlikely to know how to use a symbolic math system on an open-ended project. Consequently, the instructor should demonstrate the pursuit of at least one such project. Accordingly, section 2 is a partial scenario of how such a demonstration could proceed, with a corresponding supplementary discussion in section 3. The appendix lists a number of suggested discovery-oriented computer algebra projects related to various mathematical topics.

2. A DETAILED EXAMPLE

To be specific, the demonstration here uses an experimental version of the muMATH^{tm} system. This version is scheduled for distribution sometime during 1985 for the IBM-PC and other computers using the similar MS-DOS operating system [2]. The example is well within the capabilities of virtually all systems, some of which are referenced in [3].

The system used here prompts the user with a numbered label beginning with the letter "i" for "input" and followed by a colon. The user then enters an expression terminated by a semicolon. The system then displays the computing time in seconds if it is nonnegligible compared to the computer clock resolution. Next, the system displays a numbered label beginning with the letter "o" for "output", followed by a colon then the corresponding result. The outputs can be numbers, expressions or function plots.

Previous inputs and outputs can be recalled for editing or for use in subsequent expressions. For ease of typing, inputs use "/" for division and "^" to denote raising to a power. For ease of reading, outputs use raised exponents and use built-up fractions where it is attractive to do so. The entire dialogue can automatically be recorded on diskette for subsequent editing, printing or reentry. The students would be familiar with such details from their earlier trivial exercises mentioned in the introduction.

I would give the students time to ponder the following mock project assignment for several minutes before commencing the demonstration:

> **Project:** Use your computer algebra system to explore inter-relationships among the coefficients of $(x + y)^n$, expanded for increasing n. Discuss the issues listed below and any other relevant ones that you discover:
>
> a) the number of terms;
>
> b) relations among the exponents in successive terms;
>
> c) symmetries among the coefficients for a particular n;
>
> d) relations among coefficients for two successive values of n;
>
> e) relations between a coefficient and factorials involving the corresponding exponents;
>
> f) the asymptotic growth of the largest coefficient with n.
>
> g) the asymptotic growth in computation time with n.
>
> Include plots that helped lead to your discoveries or that vividly summarize them. Include proofs if you can. Don't worry if you cannot decisively address all of these issues.

Superficially, this particular example would seem most appropriate at the point in the curriculum just before first exposure to binomial expansion. However, some parts of the project might require more maturity. It certainly doesn't ruin such a project if the students already know some of the answers. Such reinforcement can be beneficial. Moreover, elementary **demonstration** examples permit the students to concentrate on the exploratory techniques without being distracted by a flood of new mathematical facts.

I have enclosed the spoken narration below in quotes to help distinguish it from the computer dialogue with which it is interspersed.

2.1 Coefficient Patterns

"Well class, here is how I might proceed with this project if it were as new to me as it is to you. First, I would try a few successive values of n to see what that reveals:"

i1: EXPAND: TRUE;
o1: **TRUE**

"Let's set the expansion control variable to request automatic expansion until further notice."

i2: (x + y)^0;
o2: 1

"I knew this result, but such degenerate cases may be an important part of a pattern."

i3: (x + y)^1;
o3: $x + y$

"This is the only other degenerate case that I can perceive."

i4: (x + y)^2;
o4: $x^2 + 2\ x\ y + y^2$

i5: (x + y)^3;
o5: $x^3 + 3\ x^2\ y + 3\ x\ y^2 + y^3$

i6: (x + y)^4;
o6: $x^4 + 4\ x^3\ y + 6\ x^2\ y^2 + 4\ x\ y^3 + y^4$

i7: (x + y)^5;
o7: $x^5 + 5\ x^4\ y + 10\ x^3\ y^2 + 10\ x^2\ y^3 + 5\ x\ y^4 + y^5$

"It appears that there are n+1 terms when $(x+y)^n$ is expanded."

"The exponents of x appear to start at n and decrease by 1 to 0 in each successive term while the exponents of y appear to start at 0 and increase by 1 to n in each successive term."

"The coefficients appear to be symmetric about the center term or central pair of terms."

"The end coefficients appear always to be 1."

"The penultimate coefficients appear always to be n."

"I can't yet see how the other coefficients relate to n."

"However, the project assignment first suggested looking for relations between the coefficients for **successive** values of n, and I'm not too proud to accept a hint."

"It does appear that the coefficient 6 in o6 equals the sum of the coefficient 3 directly above and the coefficient 3 to its left in o5. In fact, this "sum of above and to its left" pattern holds for every coefficient if we imagine coefficients of zero surrounding the displayed nonzero coefficients! This remarkable pattern seems too simple to be true. I'll check n = 6 to see if it provides a counterexample:"

i8: (x + y)^6;
o8: $\mathbf{x^6 + 6\,x^5\,y + 15\,x^4\,y^2 + 20\,x^3\,y^3 + 15\,x^2\,y^4 + 6\,x\,y^5 + y^6}$

"The pattern still holds!"

"How far should I explore? I could write a procedure with a loop that increments n by 1 each time and compares the coefficients in $(x + y)^n$ with the appropriate sums of those in $(x + y)^{n-1}$ until a counterexample is encountered or until the computer runs out of memory. I can run the program overnight. Even if the program does not find a counterexample by tomorrow morning, the increased evidence for the rule would encourage me to seek a proof. Parts f and g of the project may even permit me to estimate how large n can become before I run out of memory space or patience. However, I'll postpone writing, debugging and starting that program until I have no further ideas for quick interactive experiments."

"The next part of the assignment is to discover a relationship between each coefficient and factorials involving the corresponding exponents. Well, 0!=1!=1, 2!=2, 3!=6, 4!=24, 5!=120 and 6!=720; so the coefficient of $x^k\,y^{n-k}$ is clearly not simply n!, k! or (n-k)! Thus the coefficient must be some composition of factorials if it involves factorials at all. Moreover, since the coefficients are symmetric, the composition should be symmetric in k and n-k."

"I cannot yet perceive an obvious relation, so I will give up on that -- at least for a while. Perhaps an inspiration will occur after some experience with other aspects of the project or after a sufficient incubation period."

2.2 Coefficient Growth

"The next suggestion is to study the asymptotic growth in the largest coefficient as n increases. The largest coefficient appears to always be the central one when n is even or either of the equal central pair when n is odd. Through n = 6 the growth is rather modest, so rather than continuing to creep along by uniform increments of 1, let's next try n = 8, 16, 32, ..., doubling n each time until we run out of memory or patience."

"When n is even, the center coefficient is that of $x^{n/2}\,y^{n/2}$. Accordingly, we can use the built-in coefficient extraction function as follows to avoid cluttering our screen with superfluous information:

i16: COEF ((x + y)^8, x^4 y^4);
0.3 sec.
o16: **70**

i17: COEF ((x + y)^16, x^8 y^8);
0.8 sec.
o17: **12870**

i18: COEF ((x + y)^32, x^16 y^16);
2.6 sec.
o18: **6010_80390**

i19: COEF ((x + y)^64, x^32 y^32);
7.6 sec.
o19: **1832_62414_09425_90534**

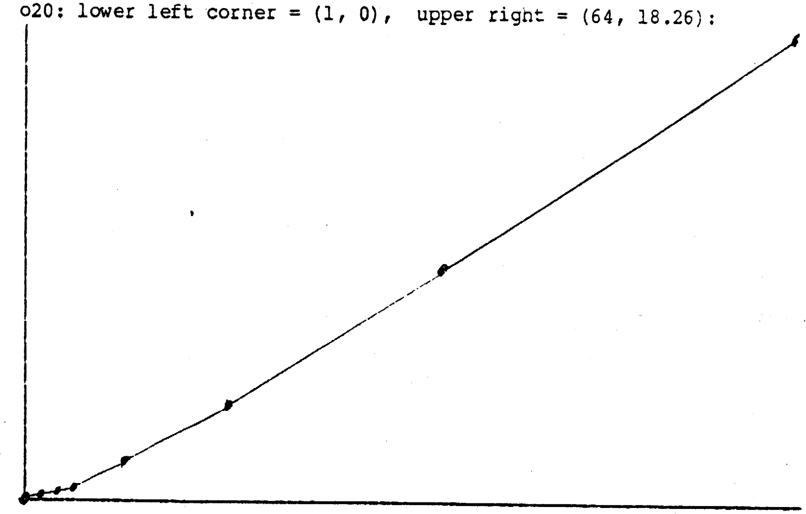
"This sequence will soon become too time consuming for interactive exploration. If I decide to do more, I'll write a procedure containing a loop and run it overnight. A vague pattern of sorts has already emerged anyway:"

"Considering also the previously done cases n = 1, 2 and 4, each doubling of n appears to approximately double the number of digits. Thus, the number of digits in the largest coefficient appears to be roughly proportional to n."

"Since the number of **digits** in a coefficient is approximately proportional to the logarithm of the coefficient, the coefficient itself appears to grow approximately exponentially with n. Logarithms to differing bases are proportional, so the choice of base is not crucial. However, since we are interested in the number of **decimal** digits, let's plot the piecewise linear interpolant of LOG_{10} (largest coefficient) as a function of $n \geq 1$ to see how well it approaches a straight line with increasing n:"

i20: LINEARSPLINE ([1,LOG(1,10)], [2,LOG(2,10)], [3,LOG(3,10)], [4,LOG(6,10)], [5,LOG(10,10], [6,LOG(20,10)], [8, LOG(o16,10)], [16, LOG(o17,10)], [32, LOG(o18,10)], [64, LOG(o19,10)]);

o20: lower left corner = (1, 0), upper right = (64, 18.26):



"The semi-log plot appears to approach a linear asymptote quite well, so let's fit a line through the two largest measurements to use for predicting the number of digits with larger n:"

i21: SOLVE(s-LOG(o18,10))/(LOG(o19,10)-LOG(o18,10))=(n-32)/(64-32), s);
o21: **{s = 0.296379 n - 0.705209}**

"$(x + y)^n$ has n+1 coefficients varying from 1 through this maximum number of digits s. Their average appears to be more than half s, so let's conservatively estimate the total space as n*s:"

i22: n RHS (o21 [1]);
o22: **0.296379 n^2 - 0.705209 n**

"Thus, n = 128 would use a total number of digits about:"

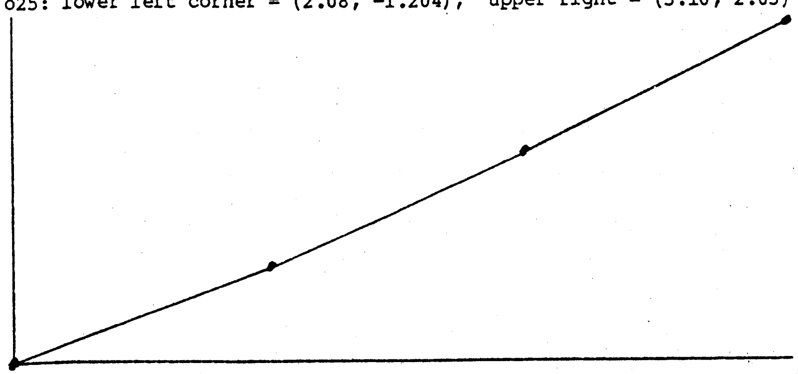
i23: SUBST (o22, n: 128);
o23: **4765.61**

"My computer has enough memory for simultaneously holding a few tens of thousands of digits total. Consequently, allowing a generous margin for other numbers created during the expansion, there should be sufficient room for one or two more doublings.

2.3 Computing Time

"Now let's estimate how much time these larger values on n will require: The computing time appears to increase by a constant factor of about 3 as n increases by a factor of 2. This suggests an asymptotic power-law dependence: $t \sim c\, n^p$. Just as exponential growth is associated with a straight-line **semi-log** plot, power-law growth is associated with a straight-line **log-log** plot. The choice of base is not crucial, so I'll use the natural log:"

i24: LINEARSPLINE ([LN 8, LN 0.3], [LN 16, LN 0.8], [LN 32, LN 2.6], [LN 64, LN 7.6]);
o25: lower left corner = (2.08, -1.204), upper right = (5.16, 2.03)



"The log-log plot appears to approach a linear asymptote quite well, so to fit a line through the logarithms of the two largest measurements to use for prediction:"

i25: SOLVE (
(LN t - LN 2.6)/(LN 7.6 - LN 2.6) = (LN n - LN 32)/(LN 64 - LN 32), t);
o25: **{t = 0.0191558 $n^{1.54749}$}**

"Thus, I guess that if we don't run out of space, the number of seconds required to compute $(x+y)^{256}$ would be about:"

i26: RHS (o25 [1]);
o26: **0.0191558 $n^{1.54749}$**

i27: SUBST (o26, n: 256);
o26: **64.9389**

"This is feasible to try right here in class, but my plan was to compute **and compare** expansions for **all** successive n up through the maximum allowable by the memory. Consequently, our total time as a function of the last value of n, which I'll call m, would be at least:"

i28: SUM (o26, n, 0, m);

o28: **$0.0191558 \sum_{n=0}^{m} n^{1.54749}$**

"The system was unable to find a closed form for this indefinite sum, and I would't be surprised if none exists in terms of the elementary functions with which we are all familiar. Consequently, let's try approximating the sum by an analogous integral:"

i29: DEFINT (o26, n, 0, m);
o29: **0.00751948 $m^{2.54749}$**

"Now we can estimate how far we can get in a 12-hour computation:"

i30: SOLVE (o29 = 12*60*60, m);
o30: **{m = 450.107}**

"It appears that an overnight run will indeed be the right order of magnitude for proceeding by increments of 1 until we exhaust the memory available for numbers." ...

3. DISCUSSION

The space limitation here prevents me from completing the scenario. However, the demonstration would continue on to the point of showing how computer algebra can be used to support theorem proving. Next I would distribute a sample written report based on the demonstrated experiments and proofs. Then I would distribute an appropriate project assignment of this nature for the students to do. The appendix contains a list of such projects addressing a variety of mathematical topics. This list is the beginning of a list that I plan to collect and refine for publication. Suggestions and additions will be gratefully acknowledged.

It might be wise to give each student two or three choices, because their individual insight could vary erratically on open-ended problems such as these. For mathematical topics that suggest a great many projects, it might be especially motivating to allocate the choices in such a way as to collectively attack most of the problems, with each report then presented to the group so as to pool experiences.

It is important to note that the scenario and the projects in the appendix have not been classroom tested. I am not a mathematics educator, so I do not expect to have an opportunity to test these ideas directly myself. Rather, these are merely proposals that I encourage math educators to try, criticize, supplement and modify. More specifically, I hope to have available throughout the conference at least one suitable computer so that you can try out some of these ideas and react to them.

4. REFERENCES

1. Position statement, study group 3.1.4: "Symbolic mathematical systems and their effects on the curriculum", Proceedings of ICME 5, University of Adelaide, Australia, (forthcoming).

2. The Soft Warehouse, Honolulu, Hawaii, 96822, USA; Telex 6502417437,

3. Stoutemyer, D.R., "A Radical Proposal for Computer Algebra in Education", forthcoming, ACM SIGSAM Bulletin (Association for Computing Machinery, 1133 Avenue of the Americas, New York, N.Y. 10036, USA.) A shortened version will appear in the Proceedings of ICME 5, University of Adelaide, Australia.

5. APPENDIX: MATH DISCOVERY PROJECTS USING COMPUTER ALGEBRA

5.1 Elementary Algebra:

1. Experiment with your computer algebra system to form a conjecture about how the reduced form of the algebraic expression $(x^m - 1) / (x^n - 1)$ depends on m and n. Then, use the system to help prove your conjecture inductively. Discuss the growth of computing time (and perhaps also space) with m and n.

3. Use your symbolic math system to factor $x^n \pm y^n$ over the integers for increasing n. Form some conjectures about the number and form of the factors versus n. For example, are the factors of n relevant? How do the coefficient magnitudes vary with n? Try proving your conjectures. What is the asymptotic growth of computing time (and perhaps also space) with n?

4. Using computer algebra, determine the reduced forms of

$$1/(1 - x^2/(3 - x^2/5)),$$
$$1/(1 - x^2/(3 - x^2/(5 - x^2/7))),$$
$$1/(1 - x^2/(3 - x^2/(5 - x^2/(7 - x^2/9)))),$$

etc., with the constants being successive odd integers. Superimpose plots of these functions. Does the sequence of functions appear to be converging to a well known function? Is the convergence monotonic? How does the computing time appear to grow asymptotically with the number of operations in the truncated approximation? If you know a power-series approximation for the same function, how does it compare in speed versus accuracy for different ranges of x?

5.3 Matrices & Determinants:

1. Use your computer algebra system to form the matrix products

$$\begin{pmatrix} a & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} b & 1 \\ 1 & 0 \end{pmatrix}, \quad \begin{pmatrix} a & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} b & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} c & 1 \\ 1 & 0 \end{pmatrix}, \quad \text{etc.,}$$

each time including one more matrix until you can infer the general form of the elements in the product. Then, see if you can use the system to help inductively prove your general form.

2. For each of the following exercises, use your computer algebra system to compute successively higher-order determinants of the indicated family until you can conjecture the general form. Try to prove your conjecture. What is the nature of the growth in computing time and space versus order? Beware that the behaviour may differ for odd and even orders. Also, you may need to expand, factor, or otherwise rearrange the nominal results in order to reveal the most regular form.

a) $\begin{vmatrix} 1 & 1 & 1 & 1 \\ 1 & a+1 & 1 & 1 \\ 1 & 1 & b+1 & 1 \\ 1 & 1 & 1 & c+1 \end{vmatrix}$

b) $\begin{vmatrix} 1 & a & b & c \\ a & 1 & 0 & 0 \\ b & 0 & 1 & 0 \\ c & 0 & 0 & 1 \end{vmatrix}$

c) $\begin{vmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & b & b \\ 1 & b & 0 & b \\ 1 & b & b & 0 \end{vmatrix}$

d) $\begin{vmatrix} 1 & -1 & 0 & 0 & 0 \\ x & h & -1 & 0 & 0 \\ x^2 & hx & h & -1 & 0 \\ x^3 & hx^2 & hx & h & -1 \\ x^4 & hx^3 & hx^2 & hx & h \end{vmatrix}$

e) $\begin{vmatrix} 1 & a & a^2 & a^3 \\ 1 & b & b^2 & b^3 \\ 1 & c & c^2 & c^3 \\ 1 & d & d^2 & d^3 \end{vmatrix}$

f) $\begin{vmatrix} x & 0 & 0 & y \\ y & x & 0 & 0 \\ 0 & y & x & 0 \\ 0 & 0 & y & x \end{vmatrix}$

5.4 Summation:

1. Note that $\sum_{k=1}^{n} k^0 = n$ and $\sum_{k=1}^{n} k^1 = n^2/2 + n/2$.
   Guess a relationship between the highest degree term of $\sum_{k=1}^{n} k^m$ and $\int n^m \, dn$, then prove this relationship if you can. Use your computer algebra systems to experimentally determine all of the terms in $\sum_{k=1}^{n} k^m$ for several successive m beginning with m = 2, and use the system to inductively prove each of your formulas. Then see if you can devise a formula or an algorithms that works for arbitrary nonnegative integer m.

5.5 Generating Functions & Power Series:

1. Using your computer algebra system, verify the following power series and determine their intervals of convergence:

   $(1 - x)^{-1} = 1 + x + x^2 + x^3 + \ldots$

   $(1 - x)^{-2} = 1 + 2x + 3x^2 + 4x^3 + \ldots$

   $(1 + x)/(1 + x + x^2) = 1 - x^2 + x^3 - x^5 + x^6 - x^8 + x^9 - \ldots$

   $(1 + x)/(1 - x)^3 = 1 + 4x + 9x^2 + 16x^3 + 25x^4 + \ldots$

   Then, using these as building blocks or inspirations, see if you can experimentally discover rational expressions having the following power series expansions:

   a) $1 - x + x^2 - x^3 + x^4 - x^5 + \ldots$

   b) $1 + 2x + 4x^2 + 8x^3 + 16x^4 + 32x^5 + \ldots$

   c) $1 + 2x + 3x^2 + x^3 + 2x^4 + 3x^5 + x^6 + 2x^7 + 3x^8 + \ldots$

   d) $1 + 2x + 3x^2 + 2x^3 + x^4 + 2x^5 + 3x^6 + 2x^7 + x^8 + \ldots$

5.6 Integration & Differentiation

1. Use your computer algebra system to evaluate the indefinite integral of $x^n e^{a x}$ for increasing n beginning with n = 0, until you can infer the general form. Then use the system to help you inductively prove that form.

2. The size of successive partial derivatives can grow rapidly, especially if the original expression involves nested function compositions or nontrivial denominators. Find a particularly compact and innocent looking expression whose successive derivatives grow remarkably. The most dramatic example earns a prize!

Formula manipulation in teaching perturbation methods

by A.Ollongren
(Department of Computer Science, Leiden University, Wassenaarseweg 80, 2333 AL LEIDEN, The Netherlands).

Abstract

The paper gives an example of the application of an interactive formula-manipulation system in obtaining a solution of a simple perturbed ordinary differential equation of the second order. In a general way it is shown how one arrives step by step at the terms of a solution in the form of a truncated Fourier expansion. Afterwards one notices certain regularities, which may be used as a basis for further expansions or theorems. The methods are sufficiently general and may be applied to less simple perturbation equations, for which symbolic solutions are required.

1. Methods for the solution of differential equations can roughly be said to fall in two categories: exact integration methods (in which the quadrature of integrals is sought) and asymptotic methods (in which solutions in the form of series expansions are considered). Among the expansions we may distinguish between Taylor expansions with respect to the independent variable, and Fourier-expansions with terms periodic in the independent variable. In this paper we consider only asymptotic methods and we are especially interested in methods which yield the coefficients of Fourier expansions. Such expansions are particularly relevant if an ordinary differential equation contains a "small" term, i.e. if there is some coefficient in which the solution can be proved to be analytic, and which can be considered to be so small that the associated term is a perturbation term in the equation. As an elementary example of such a situation we consider the following equation

$$y'' = -y + \delta y^2 \qquad (1)$$

where x is the independent variable, $y' = dy/dx$ and δ is a small quantity. Then δy^2 is a perturbation term. For $\delta = 0$ and choosing the integration constants as follows

$$y(0) = 0 \qquad y'(0) = 1 \qquad (2)$$

we find the exact solution

$$y = \sin(x)$$

It is known that for $\delta \neq 0$ there is no asymptotic periodic solution in x. But there is such a solution periodic in u where u depends analytically on x:

$$u = x(1 + c_1 \delta + c_2 \delta^2 + \ldots)$$

In the next section we discuss a method of finding this solution, in which the usefulness of modern powerful formula-manipulation systems is shown.

2. First we observe that the given equation transforms to

$$\ddot{y}(1 + c_1 \delta + c_2 \delta^2 + \ldots)^2 = -y + \delta y^2 \qquad (3)$$

where u is the independent variable and $\dot{y} = dy/du$. Our task is then the determination of the coefficients y_i as periodic functions in u in

$$y = y_0 + y_1 \delta + y_2 \delta^2 + \ldots.$$

In order to resolve this problem we use one of the modern formula-manipulation systems, such as REDUCE or MACSYMA. In this paper we use the REDUCE formalism.
First we decide on the order to which we shall calculate the required expansion, say n : = 3. Next we assign the lefthand side of the equation to a variable l by

```
s : = for i : = 0 : n   sum c(i) * δ * * i;
y : = for i : = 0 : n   sum y(i,u) * δ * * i;
l : = d f (y,u,2) * s * * 2;
```

The righthand side of the equation is assigned to a variable r by

```
r : = -y + δ * y * * 2;
```

and then we can assign the first n coefficients of l-r to n elements of an array ar by

```
array ar(n);
coeff (l-r,δ,ar);
```

Inspection shows the value of ar(0), which is $c_0 \ddot{y}_0 + y_0$.
In view of the chosen integration constants and because $c_0 = 1$ we set in REDUCE notation

$y(0,u) := \sin(u);$ $\quad c(0) := 1;$

so that ar(0) vanishes.

Inspection shows as the next step the value of ar(1), which is $-2c_1 \sin(u) - \sin^2(u) + y_1 + \ddot{y}_1$. In this term one has to go over from the angle u to the double angle 2u and express $\sin^2(u)$ in terms of cos(2u). In REDUCE we set the transformation by the rule

<u>for</u> <u>all</u> x <u>let</u> sin (x) ** 2 = (1 - cos (2 * x))/2;

We are then left with

$$ar(1) = -2\, c_1 \sin(u) + \tfrac{1}{2} \cos(2u) + y_1 + \ddot{y}_1 - \tfrac{1}{2}$$

The objective at this stage is to let ar(1) vanish by suitable choices of c_1 and y_1. Inspection of above formula shows that we must use the following form for y_1:

$$a_{11} \sin(u) + b_{11} \cos(u) + b_{12} \cos(2u) + const_1$$

After substituting this into the formula for ar(1) it is easily seen that ar(1) vanishes if and only if

$$c_1 = 0 \quad b_{12} = \frac{1}{6} \quad const_1 = \tfrac{1}{2}$$

which leaves us with the problem of determining the values of the remaining integration constants a_{11} and b_{11}. But these are found from the initial conditions(2), which formulated in terms of , say

$$y(0) = 0 \qquad \dot{y}(0) = 1/\ (1 + c_1\delta + c_2\delta^2 + \ldots)$$

This is satisfied up to the first degree of δ if and only if <u>sub</u> (u = 0,y(1,u)) and <u>sub</u> (u = 0, <u>df</u> (y(1,u),u,1)) yield zero. That is the case if and only if $a_{11} = 0$ and $b_{11} = -\frac{2}{3}$, so that

$$y_1 = -\frac{2}{3} \cos(u) + \frac{1}{6} \cos(2u) + \frac{1}{2}$$

3. Before we are prepared to draw some conclusions we consider the next step and inspect the coefficient of δ^2, which is ar(2). It is

$$-2\, c_2 \sin(u) - \sin(u) \quad - \frac{1}{3} \cos(2u) \sin(u) + \frac{4}{3} \cos(u) \sin(u) + y_2 + \ddot{y}_2$$

As there are two products of sines and cosines in this expression we introduce the rules

<u>for</u> <u>all</u> x <u>let</u> cos(2 * x) * sin(x) = (sin(3 * x) - sin x))/2;
<u>for</u> <u>all</u> x <u>let</u> cos(x) * sin(x) = (sin(2 * x))/2;

This gives for ar(2)

$$-2c_2 \sin(u) - \frac{5}{6} \sin(u) + \frac{2}{3} \sin(2u) \qquad - \frac{1}{6} \sin(3u) + y_2 + \ddot{y}_2$$

From this formula it is seen that we must use the following form for y_2:

$$a_{21} \sin(u) + b_{21} \cos(u) + a_{22} \sin(2u) \qquad + a_{23} \sin(3u) + const_2$$

Upon substitution in the expression for ar(2) it is seen that ar(2) vanishes if and only if

$$c_2 = -5/12 \quad a_{22} = 2/9 \quad a_{23} = -1/48 \quad const_2 = 0$$

The remaining integration constants a_{21} and b_{21} are again found from the integration conditions (2). We find from the requirements that <u>sub</u> (u = 0, y(2,u)) and <u>sub</u> (u = 0, <u>df</u> (y(2,u),u,1) -5/12) must yield zero:

$$a_{21} = \frac{5}{144} \quad b_{21} = 0$$

We have therefore

$$y_2 = \frac{5}{144} \sin(u) + \frac{2}{9} \sin(2u) - \frac{1}{48} \sin(3u)$$

Continuing in the same way we find

$$c_3 = 0$$
$$y_3 = -\frac{137}{216} \cos(u) + \frac{2}{27} \cos(2u) + \frac{1}{24} \cos(3u) - \frac{1}{432} \cos(4u) + \frac{25}{48}$$

As a final check on the calculations we determine anew s,y, l and r (see section 2), but now using

$$c_0 = 1 \quad c_1 = 0 \quad c_2 = -5/12 \quad c_3 = 0$$

and the calculated values of y_0, y_1, y_2 and y_3. It is then seen that 1-r indeed is zero up to degree 3 in δ. Note that the "new" r contains products of sines and cosines stemming from y^2. These, however, cancel one antoher in the "new" ar(0) ... ar(3) so that no new <u>for</u> <u>all</u> rules need to be introduced.

4. Summarising the discussion of the preceding sections we can state that we have obtained a solution of the differential equation $y'' = -y + \delta\, y^2$ subject to the initial conditions $y(0) = 0$ $y'(0) = 1$ as a truncated Fourier series in u, where

$$u = x\ (1 + c_1\delta + c_2\, \delta^2 + \ldots)$$

by a semi-automatic method. We have used an interactive formula-manipulation system (REDUCE in fact) and we have determined step by step c_i ($i \geq$ and y_i ($i \geq 0$). Each step consisted of the following substeps:

- inspection of ar(i) in which the coefficients of δ^i in $\ddot{y}$ * $(1 + c_1\, \delta + \ldots + c_i\, \delta^i)^2 + y - \delta y^2$ are collected; this yields c_i and the form of y_i as a linear combination of sines and cosines of multiples of u, supplemented with a constant term
- determination of the coefficients of the linear combination from the

condition ar(i) = 0
- determination of the two remaining coefficients (the integration constants) from the conditions $y_i(0) = 0$ $\dot{y}_i(0) = h_i$ (see appendix).
A posteriori we note that only for even i, $c_i \neq 0$ and furthermore for even i, y_i contains only sines and for odd i, y_i contains only cosines. Furthermore the multiples of u in the Fourier terms of y_i range from u to (i + 1) * u. One may have suspected these facts in advance, but only after having carried out the expansions as shown it becomes worthwhile considering a proof.
In conclusion we learn from this example the usefulness of a programming environment which supports semi-automatic formula manipulation. The idea of a perturbation method is nicely illustrated by the facilities of the system and can even be shown in an interactive session in a classroom situation. At the same time a solution is developed. One shows how one keeps track of the various steps involved; at any moment the database containing results obtained so far can be inspected. All of this helps to clarify the methods employed.

Literature

A.Ollongren 1984, "Classroom experience with interactive FORMAC-73", Proceedings ICME 5, Adelaide, forthcoming.

Appendix: REDUCE script

Below follows a set of instructions written in REDUCE for the symbolic solution of the differential equation

$$\ddot{y} = (-y + \delta.y^2)\ s^{-2}$$

discussed in the paper. The set is a kind of script. It contains assignments of which it is known a priori that they are needed, and assignments and substitution rules, which are found to be needed in order to meet certain requirements as explained in the paper. The latter are shown by indentation. We have prepared a VHS-video tape displaying an actual session based on the script.

```
COMMENT: THIS IS A REDUCE SCRIPT
         FOR THE SOLUTION OF
         DF(Y,U,2)*S**2 =
                   -Y + D*Y**2
         WHERE
         S = 1+C(1)*D+C(2)*D**2+ . . .
         AS A FOURIER SERIES TO THE
         THIRD DEGREE IN D;
COMMENT: COPYRIGHT
         ALEXANDER OLLONGREN
         LEIDEN UNIVERSITY
         THE NETHERLANDS 1984;
LINELENGTH(40) $
ON LIST;
ON DIV;
N:=3 $
S:=FOR I:=0:N SUM C(I)*D**I $
Y:=FOR I:=0:N SUM Y(I,U)*D**I $
L:=DF(Y,U,2)*S**2 $
R:=-Y+D*Y**2 $
ARRAY AR(3*N) $
NFACTORS := COEFF(L-R,D,AR);
CLEAR S,Y,L,R $
FOR I:=(N+1):NFACTORS DO AR(I):=0 $
AR(0);
COMMENT: AFTER INSPECTION CONCLU-
         SIONS ARE DRAWN AND NEW
         INFORMATION IS SUPPLIED
         TO THE SYSTEM (SHOWN BY
         INDENTATION);
    C(0):=1 $
    H(0):=1 $
AR(0);
    Y(0,U):=SIN(U) $
AR(0);
AR(1);
    LET SIN(U)**2 = (1-COS(2*U))/2 $
AR(1);
    Y(1,U):=A(1,1)*SIN(U)+
            B(1,1)*COS(U)+
            B(1,2)*COS(2*U)+
            CONST(1) $
AR(1);
    <<C(1):=0; CONST(1):=1/2;
      B(1,2):=1/6>> $
C(1)*H(0)+C(0)*H(1);
      H(1):=0 $
AR(1);
OFF LIST;
SUB(U=0,Y(1,U));
    B(1,1):=-2/3 $
SUB(U=0,DF(Y(1,U),U,1)) - H(1);
    A(1,1):=0 $
ON LIST;
```

```
AR(2);
     LET COS(2*U)*SIN(U) =
          (SIN(3*U)-SIN(U))/2 $
     LET COS(U)*SIN(U) =
          (SIN(2*U))/2 $
AR(2);
     Y(2,U):=A(2,1)*SIN(U)+
               B(2,1)*COS(U)+
               A(2,2)*SIN(2*U)+
               A(2,3)*SIN(3*U)+
               CONST(2);
AR(2);
     <<C(2):=-5/12; CONST(2):=0;
       A(2,2):=2/9; A(2,3):=-1/48>> $
C(2)*H(0)+C(1)*H(1)+C(0)*H(2);
     H(2):=5/12 $
AR(2);
OFF LIST;
SUB(U=0,Y(2,U));
     B(2,1):=0 $
SUB(U=0,DF(Y(2,U),U,1)) - H(2);
     A(2,1):= 5/144 $
ON LIST;
AR(3);
     LET COS(2*U)**2 =
          (1+COS(4*U))/2 $
     LET COS(2*U)*COS(U) =
          (COS(U)+COS(3*U))/2 $
     LET COS(U)**2 = (1+COS(2*U))/2 $
     LET SIN(3*U)*SIN(U) =
          (COS(2*U)-COS(4*U))/2 $
     LET SIN(2*U)*SIN(U) =
          (COS(U)-COS(3*U))/2 $
AR(3);
     Y(3,U):=A(3,1)*SIN(U)+
               B(3,1)*COS(U)+
               B(3,2)*COS(2*U)+
               B(3,3)*COS(3*U)+
               B(3,4)*COS(4*U)+
               CONST(3) $
AR(3);
     <<C(3):=0; CONST(3):=25/48;
       B(3,2):=2/27; B(3,3):=1/24;
       B(3,4):=-1/432>> $
C(3)*H(0)+C(2)*H(1)+
C(1)*H(2)+C(0)*H(3);
       H(3):=0 $
AR(3);
OFF LIST;
SUB(U=0,Y(3,U));
     B(3,1):=-137/216 $
SUB(U=0,DF(Y(3,U),U,1)) - H(3);
     A(3,1):=0 $
ON LIST;

COMMENT: NOW THE FINAL CHECKING;
S:=FOR I:=0:N SUM C(I)*D**I $
INVS:=FOR I:=0:N SUM H(I)*D**I $
Y:=FOR I:=0:N SUM Y(I,U)*D**I $
L:=DF(Y,U,2)*S**2 $
R:=-Y+D*Y**2 $
ARRAY BR(3*N) $
COEFF(373248*(L-R),D,BR) $
COMMENT: THE FACTOR 373248 RENDERS
         THE FIRST ARGUMENT OF THE
         FUNCTION COEFF A POLYNOMIAL
         WITH INTEGER COEFFICIENTS;
BR(0);
BR(1);
BR(2);
BR(3);
COMMENT: THESE SHOULD BE ZERO;
BR(4);
SUB(U=0,Y);
SUB(U=0,DF(Y,U,1)) - INVS;
COMMENT: THESE SHOULD BE ZERO;
OFF LIST;
S;
INVS;
ON LIST;
Y(0,U);
Y(1,U);
Y(2,U);
Y(3,U);
END;
```

This script can also be used to obtain higher order solutions for the equation considered. Supposing y(1,u),...,y(k-1,u) known for k ≤ 4 we can use the script as a kind of masterplan to determine y(k,u) as follows:

- ar(k) is inspected and the necessary substitution rules for products of sines and cosines are defined
- y(k,u) is set to a linear combination of sin(k+1)u,..., sin(u) for k even, or cos(k+1)u,..., cos(u) and const(k) for k odd
- y(k,n) is substituted in ar(k) and by inspection the values of c(k) and all except two of the coefficients of y(k,u) are determined uniquely such that ar(k) becomes zero
- the remaining two coefficients (the integration constants) are then determined from the requirements

  <u>sub</u> (u = 0,y(k,u))= 0

  <u>sub</u> (u = 0, <u>df</u> (y(k,u),u,1)-h(k)) = 0, where $(\Sigma\, c(i)\, \delta^{i})^{-1} = \Sigma\, h(i)\, \delta^{i}$

For k = 4 one finds in this way

$$c(4) = -\frac{95}{192}$$

$$y(4,u) = (-1753 \sin(u) + 8320 \sin(2u) - 180 \sin(3u) - 128 \sin(4u) + 5 \sin(5u))/20736$$

Leiden, June 7, 1984; revised April 7, 1985.

Discussion paper for ICMI seminar, Strasbourg, March 1985.

Some software for teaching algebra

Introduction

We have been invited to respond to the issues raised in the ICMI discussion document: THE INFLUENCE OF COMPUTERS AND INFORMATICS ON MATHEMATICS AND ITS TEACHING. This is a difficult task because the document raises many important and challenging issues. The problems raised by computers for the teaching of mathematics are in the consciousness of most teachers of mathematics. For most there is an awareness that the teaching of mathematics will have to change. Some few teachers are taking up the challenge and are designing courses and teaching strategies which reflect the availability of computers. More teachers are waiting, hopefully, for others to produce material and ideas they can use. What has surprised many people is how slow material has been in coming. This is perhaps an indication that the problems of designing courses, or parts of courses, incorporating the use of computers is harder than one at first expects. Another indication is that, in spite of my sending copies of the discussion document to quite a number of people who might be interested, I have not got one response.

In other words it is easy to dream about ways in which computers might assist the teaching of mathematics and alter the content of courses in mathematics. It is harder to turn these dreams into reality. There is still not much information available about working packages. Documentation is limited or non-existent and not easily accessible. Thus it seems appropriate to report on a software package which has been developed in Australia and to use that report to make a few comments related to the issues raised in the discussion document. Before I begin, I should point out that I report as an outsider - I have not been involved in the development, use or assessment of this package.

The package is called MATRIX. It was designed by Dr J.J.Cannon of the University of Sydney to assist undergraduate students with the learning of linear algebra. Thus it fits into the category of computer-assisted learning. It was first used for this purpose in 1975. Since 1981 it has also been used to support courses in linear algebra at the University of Melbourne. Moreover in Melbourne there have been investigations into the impact of MATRIX on student's learning. It is also being used more widely but I know of no reports on the experience of other users. There is also a related package, CAYLEY, for doing calculations in groups. CAYLEY was designed primarily for use as a research tool. However it has been used for a number of years as an adjunct to a first course in the theory of groups at the University of Sydney.

MATRIX

MATRIX now consists of
(a) a calculator for calculating with matrices with rational or variable precision real number entries,
(b) a simple command language which allows a student to use the calculator and to access exercises which are set in advance by a teacher, and
(c) a facility for writing exercises (and hints and solutions) which can be used by teachers.

I will just give a brief description of some of the more significant features, more details are available in references [1],[2] and [5]. The calculator is fairly simple. Nevertheless it is quite adequate for teaching purposes, it certainly allows the setting of reasonably realistic exercises. (In fact it is used by people such as engineers in their ordinary work.) It can do all the usual operations on matrices including elementary row and column operations, pivoting and reduction to echelon form (echelonising). It can calculate Hadamard and Kronecker products. It can calculate eigenvalues and eigenvectors. It can generate random matrices (useful for exercises and quizzes). It can operate on vectors considered as row or column matrices. It can determine whether a column of a matrix is a linear combination of other columns. It can handle linear programming problems. From the designer's point of view a critical point for teaching purposes is that some of the operations can be denied to students until they are considered competent in the use of them; thus, for example, both pivoting and echelonising can be withheld until a student has successfully used elementary operations to reduce a matrix to echelon form.

This last remark raises an important paedagogic question: to what extent should black-boxes be used in the teaching of mathematics (or for that matter in any teaching)? Thus one could argue that for many students a black-box equation solver is all they need. If they have a good understanding of the meaning of the output the black-box produces they can readily check that the solution presented by the black-box is correct. Or one could argue that a student should only be allowed to use a black-box they themselves have constructed. This will remain a subject for debate for a long time and it is unlikely that agreement is possible or even desirable. Thus it seems entirely appropriate to have the possibility of teacher control in the way MATRIX does. It is easy to envisage increasing the scope of such a facility. Even so an element of black-box is likely to remain. For example, it would seem quite unrelated to the goals of a first course in linear algebra to discuss the data structures which underpin the calculator.

The writing facility is designed so that a teacher who can communicate with a computer can use it to write exercises, no knowledge of programming is needed. The writing facility has been successfully used by a number of people with such limited knowledge. As pointed out in the discussion document, the use of computers requires different kinds of exercises and different kinds of hints and solutions. In a context where there is reduced direct contact between learner and teacher, more sensitivity of the teacher/writer to the needs and interpretations of students is also required. This proves to be quite demanding of time and skill. In addition it is possible, with a little more experience, to write quizzes for the students. The writing facility applies in a somewhat more general context than MATRIX, it can also be used in conjunction with CAYLEY and with some of the other programs of a similar kind which are being developed.

The use of MATRIX

At both Sydney and Melbourne (universities) MATRIX is used to support conventional (computer-free) teaching of linear algebra by lectures, tutorials, practice classes, exercise sheets, assignments and examinations. At Sydney in most of the relevant courses about half the tutorials (which consist of about twenty students meeting with a tutor for an hour) are set in a computing laboratory where each student has access to a computer via a terminal. Otherwise at Sydney and always at Melbourne the situation is that MATRIX is available to students in linear algebra courses outside scheduled class hours and with no supervision. To compensate for this it is supported by an instruction manual and at Melbourne also by workbooks and a videotape which introduces the relevant parts of the package.

At Melbourne in 1982 and 1983 Dr E.A.Sonenberg has conducted surveys into the use of MATRIX to support courses involving a significant component of linear algebra. Her results and conclusions are reported in [3] and [4]. Her main source of information is student questionaires of which over 800 were completed. She found that at least three-quarters of the students tried MATRIX and about half of those continued to use it throughout the course. Moreover at least two-thirds of those who tried MARTIX would recommend it to friends. Major reasons for not trying MATRIX or for giving-up were: lack of time, not necessary to understand the course. Naturally the amount of time a student can spend on a course is limited, so time spent learning to use an additional tool must come from that time and must be seen to be well spent. In the context of MATRIX one hopes the time comes from that otherwise spent in doing unproductive arithmetic calculations. The other response is more interesting. Teachers of mathematics have always believed that the essential ideas in a course could be explained on simplified, artificial examples. For some students they seem to be right; for the majority it seems less simple, less artifical examples are more attractive - and perhaps more effective?

Among other reasons given for not trying or for giving-up were: unfamiliar with computers, don't like using computers. The former will presumably go away. What about the dislike of computers? Are a majority of these female? If so, will it exacerbate the problems involved in the learning of mathematics by females?

To me one of the more significant comments was: it can't be used in examinations. This is a well-known constraint on innovation of any kind. With computers there are additional constraints such as the lack of resources to allow appropriate examination of all students in a reasonable period of time in a manner which is accepted as fair. These are serious barriers to computer influenced changes in curriculum. The power of computer based calculators raises another issue of assessment and certification. Perhaps it will become important to be able to certify that a person can correctly and efficiently use a particular calculator. Of course this is an issue for calculators in general. Is it being faced at all? Even in the simplest case of hand-held four-function calculators?

In summary MATRIX is making a significant contribution to the teaching of linear algebra. Undoubtedly it could have more features and it could then be even more effective. However such a more ambitious design might not yet have been in operation. The more pragmatic approach seems well justified. The idea of teacher controlled functions and the idea of an easy to use writing facility are both ones with wider relevance. For example, the many symbolic algebra systems could undoubtedly be made into quite effective learning packages with the use of such techniques.

References

[1] Jim Richardson, MATRIX - Teaching Linear Algebra by Computer, contribution to ICME 5, 1984.
[2] Jim Richardson, MATRIX Author's Guide, University of Sydney, 1984.
[3] Elizabeth A. Sonenberg, The use of MATRIX in mathematics courses, Technical Reports 83/2,13, Department of Computer Science, The University of Melbourne, 1983.
[4] E.A. Sonenberg, A survey of the use of MATRIX in mathematics courses, contribution to the CALITE conference, 1983.

[5] The Linear Algebra System MATRIX Reference Manual(Second Edition), Department of Pure Mathematics, The University of Sydney, 1983.

M.F.Newman
Department of Mathematics, IAS
Australian National University
CANBERRA, AUSTRALIA
February 1985

Introducing Computer Algebra to Users and to Students

Jacques Calmet
LIFIA/ENSIMAG, Grenoble, France (*)
and
Dept. of Computer Science, University of Delaware

ABSTRACT

This paper is concerned with three different aspects of the impact of Computer Algebra Systems on Mathematical Education. The first part presents some remarks drawn from introducing these systems to users with a strong mathematical background. The second one deals with the teaching of Computer Algebra to students with different mathematical skills. The last part looks at the capabilities of available systems as teaching aids in a mathematical curriculum.

May 16, 1984

Introducing Computer Algebra to Users and to Students

Jacques Calmet
LIFIA/ENSIMAG, Grenoble, France (*)
and
Dept. of Computer Science, University of Delaware

1. Introduction

Many applications show the excellence of Computer Algebra (CA) as a computational tool in different field of science [1]. An unsolved problem is to evaluate it as a teaching aid in mathematics. The goal of this paper is to help to find an answer to this question.

It must be noted that some CA Systems (CAS) or programs either have been or are written for educational purposes. Some are already commercial products and heavely advertised. This means that we are already beyond the stage of answering to the "original" question: do we really need educational computer algebra systems? They are with us and going to stay for a long time, whether we like the idea or not.

It may be time still to ask ourselves if we want to use CAS only as a computing tool either supplementing or illustrating the skills of students, or as a mean of introducing conceptual insight in some domain of mathematics.

The paper is organised as follows. In section 2 some conclusions are drawn from our past (and extensive) experience in introducing CA systems and techniques to theoretical physicists. The reason for such a section is twofold. First, if one analyses the methods and techniques routinely used in this field it becomes obvious that they are closer to mathematics than to physics. Indeed a strong mathematical background is required to be successful in this field. Second, it is probably the domain where the most remarkable applications of CA have been performed. It is therefore an area where it is possible to rate the impact of the programming language on mathematically oriented applications and to draw some conclusions on how to design a system fitting the needs of mathematicians.

In section 3 we mention briefly some experiences related to teaching CA at the undergraduate level. The emphasis is put on the mathematical background required from a student in such a curriculum and on the adequacy of the available systems for this purpose. We also look at the adequacy of CAS to the needs of instructors in different contexts (i. g. high school, university and engineering school both in Europe and in the US).

The last section lists some design ideas that we think would be helpful for designing symbolic mathematical systems better suited for being teaching aids. They can be summarised by two questions. Is it possible to get some insight - instead of simply results - in a problem by using these systems? What capabilities can we add easely to make them more useful?

2. Computer Algebra and Mathematical Physics

Theoretical Physics is often referred to as Mathematical Physics. The reason is pretty obvious when looking at the different mathematical techniques required to make any progress in this field. They range from the use of special algebra (W, C^*, Von Neumann...), ultra distributions,

200

homotopy theory, groups (SU(n), U(n), Poincaré ...) or non-euclidean geometries to simple calculus. This list is by no means exhaustive. Just for the fun of it, one may add Cantor set, special or transcendental functions including Riemann's function and polylogarithms for instance. Even the well known (by physicists) multidimensional integrals arising from Feynman graphs are similar to those obtained when studying the acceleration of convergence of series.

They are thus a group of users with a strong mathematical background. It may then be worthwhile to observe their attitude toward CAS and to learn some lessons from this observation.

2.1. The attitude of Physicists

Following are some remarks drawn from several attempts to introduce CAS to physicists. First as a fellow physicist convinced of the importance of this computational tool. Then as a computer algebraist whose eagerness lies more in learning what implication applications may have on CAS than on "marketing" them.

(i) To learn a new programming language is a "waste of time". They will invest time and efforts only when this yields to worthwhile results.

(ii) If the CAS they are using is not completely debugged (an early version for instance) it then takes several years to convince them that the next version is error-free. This is the strongest criticism they can make.

(iii) They have, most often, access to powerful computers and this is probably why they were among the first users of CAS.

(iv) The mathematical content of their most successful applications is always simple. Although many problems, as previously mentioned, call for sophisticated mathematical methods this does not show when looking at published applications. They are usually dealing with a mechanisable subset of the mathematics they use.

(v) Tutorials on a CAS seldom turn a listener into a user right away. But he may go to a local "expert" later with a specific problem. If a CAS helps him to get a solution, then he usually becomes "addicted".

(vi) Because of the preceding remark the style of tutorial is important. Provided the programming language is interactive and looks natural the emphasis must be put on the capabilities. These users do not care to know what algorithm is used to realise a specific operation. They just want a reliable procedure to do it. Users are attracted often by presenting a successful application and stressing the operations common to different classes of problems.

(vii) CAS are not, as a rule, used by physicists to acquire any mathematical knowledge. For instance to factorise a polynomial requires using finite fields. It is an open question to know whether a user performing this operation is not interested in the technique implemented because no system documents it or because he does not want to learn about it.

(viii) They are always frustrated by the limited capabilities of CAS. This as resulted in the design of various specialized packages by physicists [2].

2.2. Implications about Systems

The preceding remarks arising from the attitude of theoretical physicists must be weighted by the fact that this domain has witnessed some of the most spectacular applications of CAS. Each of them has some obvious consequences on the design principles of user oriented CAS: ease of programming, well debugged code, interactivity ... But all have been formulated and written down many times.

We prefer to make some comments on why the present CAS have not been successful as tools which give insight into the calculations performed.

Some systems, such as SAC2/ALDES, are transparent enough to allow a user to understand what is going on during a calculation. But the programming style is such that it is almost unusable as a computing tool in physics. Other CAS have some mathematical knowledge built in (REDUCE, MACSYMA, ...) but it is difficult to have access to it. Finally, some specialized

package (SCHOONSCHIP, ...) are lacking any sophisticated mathematical knowledge but are anyway liked by users. SCRATCHPAD was not known by physicists because not available.

The obvious conclusion is that the available CAS have not been designed as teaching aids. What is needed is a mathematical knowledge representation system. Some of them are under design. They will represent a new generation of systems.

Without even considering what the future CAS will be, it must be noted that the present ones are lacking some easely implementable capabilities, i.e. those connected with data bases of either results or values. Typical examples are special functions and definite integrals. These objects are often encountered in calculations. One is usually forced to look into tables or books to find either some of their values or some relation they obey. It would be helpful if CAS could provide these informations. To achieve this goal implies to introduce non-constructive algebraic methods in a CAS.

Another point worth mentioning is the poor level of communication between a program and an user: he would be pleased to get informations on how his calculation is done. What he gets usually is the number of garbage collection calls and the number of occupied memory cells.

3. Teaching Computer Algebra

What we are interested in is to investigate whether computer algebra is an appropriate tool to teach (some) mathematics. To answer to this question requires to evaluate the mathematical content of a CAS with respect to the knowledge of a student body. We are not, in this section, rating the CAS capabilities to achieve this goal. Similarly we are not at all interested in the question of teaching how to use a CAS. We distinguish three different cases.

3.1. Computer Algebra in a Computer Science Curriculum

There are at least two different approaches for teaching computer algebra. One assumes that the emphasis is on algebraic algorithms only. The other considers the system aspect as well. One illustrative area is simplification. An algorithmic introduction to simplification will be possibly based on the presentation given by Buchberger and Loos [3]. A system oriented approach will also present the techniques of pattern matching implemented in some CAS such as MACSYMA. It is our opinion that the second approach has to be selected for a computer science course.

Although such courses are better suited for graduate studies, a recent experience with final year graduate students gives some answers to the question asked above. The student body comprised computer science majors from two origins: applied mathematics in an usual university curriculum and engineering school. The former had a good formation in mathematics and a much limited one in computing. The latter had a symmetric training. The course had two goals. The first one was to introduce the basic algebraic algorithms for polynomial manipulation. The second one was an introduction to the concept of simplification in computer algebra. An attempt to use [3] as material for the course had to be rapidly ended. The presentation of simplifiers implemented in some CAS (mainly MACSYMA) went well with students familiar with programming languages but badly with those without this knowledge (despite an introduction to list manipulation).

Without entering into superfluous details, we just want to stress that this course was a good way to teach some algebra to pure computer science students. But it was not efficient to teach some programming language theory to students with almost no knowledge in "classical" computer science. Another remark is that a course on algebraic algorithms must often be system independent. Often, the only language taught to student during their curriculum is PASCAL. Furthermore, many institutions are opposed -and this is a right decision- to introducing other programming languages.

3.2. Computer Algebra in other Curricula

Most of the scientific curricula begin with some courses in mathematics: biology, chemistry, physics, engineering. Usually they are calculus oriented and intend to give students some skills in evaluating integrals, manipulating series ... Most of the material thus covered is present in CAS which could therefore be useful in this context.

For majors in mathematics the situation is different. Although they must acquire the same skills, they also must get a much better insight in what they are doing. Also they, very rapidly, are faced with topics which are -not yet!- part of CAS. These are analysis and topology for instance. It must also be noted that in many countries, math department are not very well equipped with computers: they have often access to minicomputers only.

A possible conclusion from the two previous sections is that computer algebra is probably a good teaching aid for non-math majors. The mathematical content of the present systems, i.e. the methods implemented, is not sufficient for majors in math.

3.3. High School Curriculum

This is probably a place very well suited for using computer algebra methods and systems as teaching aids. Some of the obvious problems are:

1. Each country has its own type of mathematical curriculum (and sometimes several!). For instance abstract concepts are introduced very soon in a French curriculum.
2. The only computers available in such schools are microcomputers. They cannot support a very large CAS at present.
3. Very few systems exist for such purposes. One of them is obviously muMATH [4]. Because of the first remark, it cannot be used in every context. For instance, it is not suited for a French high-school.
4. Several projects are under way to design CAS directed toward these schools. Whether any of them will be as well designed as muMATH is remains an open question.

4. CAS as Teaching Tools

We are no longer concentrating on the methods and techniques implemented in CAS but in their programming language features. Obviously this aspect must affect the methods which are implemented as well.

If technology is going to play a role in a mathematical curriculum, computer algebra systems must be instrumental in that respect. But this goal is not yet reached. This is well illustrated by the recent book of Sims [5] on abstract algebra. He uses APL as a programming language instead of MACSYMA for instance. What we tried to show so far is that CAS are not mandatory tools to be introduced in the mathematical part of any curriculum. We split our discussion into two parts: existing CAS and design principles of future one.

4.1. Existing CAS

They already offer some capabilities for being teaching aids. We do not attempt to attach each capability to one or several CAS but simply to list a few of them.

1. In a calculus course CAS may be helpful in different ways: to support the course with examples or to check exercise answers for instance.
2. In a course on algorithms, they can be used to compare different methods to do a given operation and thus illustrate the complexity analysis studies.
3. In many opportunities they can be used to free a student from simple calculus problems in a similar manner to using a pocket calculator for numerical calculations.

These are some of the existing available capabilities. Although useful, they are still of marginal interest when compared to what could be done.

4.2. Some design principles

What follows are some of the design principles of what a large symbolic manipulation system could be. We are interested in those related with the topic of this paper only.

One of the basic idea is that such a system must be a mathematical knowledge representation system rather than an algebraic system only. To achieve this goal means that it must offer the following features.

1. Implementation of as many methods as possible to perform a given operation.
2. Extension of the fields of problems tractable with CAS. Nowadays only constructive methods are implemented. We must bypass this limitation by using heuristic methods when they are the only implementable ones. Technically this means using some techniques of artificial intelligence.
3. Once the principle of point 2 is adopted, it is natural to also add theorem proving to the capabilities of such a system. This is better done using PROLOG than any other language. Now at least two versions of PROLOG implemented in LISP are available. This enables to realise this point of our program.
4. In order for point 1 to be useful we must improve the communication between the user and the system. Some or all of the following informations or capabilities must be accessible by any user upon request: information on the method used for a given calculation, selection of an alternate method at will, on-line documentation on these methods (we do not refer here to the documentation on the command but on the method itself).
5. Independently of the design options related to the programming language aspect, the previous points suggest that the best organisation for the algebraic part is a collection of algorithms (library) which are put together by means of macros.
6. Some features of the programming language part of this project can be used to improve the mathematical knowledge of the system: handling of types and check of correctness of the semantics are among them. Depending on the approach selected for simplification of expressions, this may also add to the mathematical knowledge of the system.

These are just a few of the principles which would make computer algebra systems better suited for teaching purposes. Nowadays when setting up a course on algebraic algorithms we use books, such as Knuth's volume 2 [6] or [1], never a CAS. A well suited CAS would make the option of selecting both a book and a system meaningful. It must be emphasised that this project is not to design a specific teaching tool, but that many of its aspects would make it more valuable for such a purpose than the present CAS.

References

[1] J. Calmet and J.A. van Hulzen, *Computer Algebra Applications,* in *Computer Algebra.* Edited by B. Buchberger et al., Springer-Verlag, Wien-New York, 2nd edition (1983).

[2] J.A. van Hulzen and J. Calmet, *Computer Algebra Systems,* in volume cited in [1].

[3] B. Buchberger and R. Loos, *Algebraic Simplification,* in volume cited in [1].

[4] A. Rich and D.R. Stoutemyer, *Capabilities of the muMath-79 Computer Algebra System for the INTEL-8080 Microprocessor* in Proc. EUROSAM 1979, E.W. Ng ed., Springer-Verlag LNCS, 72 ,241-248 (1979).

[5] C.C. Sims, *Abstract Algebra - A Computational Approach,* J. Wiley & Sons, N.Y. (1984).

[6] D.E Knuth, *The Art of Computer Programming,* volume 2, *Seminumerical Algorithms,* 2nd edition, Addison-Wesley, Reading, Mass. (1983).

Commentaires sur le rapport CIEM

Effet de l'informatique sur la redaction et la pensee mathematiques.

D.Lazard
Institut de Programmation
Universite P. et M. Curie
75 230 Paris CEDEX 05

I- IDENTIFICATION LEMME = PROCEDURE.

Les enonces de lemmes contiennent souvent des parametres analogues a ceux des procedures. Le fait que lors du l'invocation de ces lemmes les parametres n'ont pas le meme nom conduit souvent a des complications de redaction bien inutile; par exemple, je lis dans l'Algebre Commutative de Bourbaki (II.3.4): "la proposition resulte alors de I.2.7, prop.8 appliquee en remplacant A par B, B par A, E par T^-1 B et F par N." Il aurait ete bien plus simple de designer la prop.8 du nom parametre' de prop.8(A,B,E,F), et terminer la demonstration par "la proposition resulte alors de I.2.7, prop.8(B,A,T^-1 B,N)". C'est a la fois plus concis, plus precis et plus informatif: Il est souvent difficile de distinguer les variables libres (i.e. parametres) des enonces, surtout s'ils sont complexes et/ou mal rediges; autant les expliciter.

Cette modification de style de redaction traduit me semble-t-il une modification de pensee qui consiste a concevoir un enonce comme un operateur qui associe un resultat a des donnees d'entrees, c'est a dire l'analogue exact d'une fonction informatique.

II- RECURRENCES ASCENDANTE ET DESCENDANTE.

La vision habituelle de la notion de la recurrence par les mathematiciens est ascendante: si P est vraie pour n, elle est vraie pour n+1. Les informaticiens rencontrent quotidiennement des recurrences arborescentes qui ne s'expriment pas facilement en terme de recurrence ascendante ou iteration, mais sont tres simples en terme de recurrence descendante (pour que P soit vraie pour n, il suffit qu'elle le soit pour n-1) ou recursion (au sens informatique). Ainsi, pgcd(m,n) peut etre vue comme une application de NxN dans N definie recursivement par:

```
pgcd(m,n) := si n=0 alors m
             sinon si m>n alors pgcd(m-n,n)
             sinon pgcd(n,m).
```

Il est manifeste qu'il n'existe pas de definition recurrente ascendante aussi simple de cette fonction ainsi definie recursivement.

III- ANALOGIE FORMULE PROGRAMME.

La definition ci dessus du pgcd peut etre vue comme une formule, certe inhabituelle en mathematique traditionnelle; mais c'est aussi un programme de calcul parfaitement operationnel, bien qu'inefficace, a condition d'utiliser un langage de programmation ayant cette syntaxe (par exemple MACSYMA, apres traduction en anglais). Plus generalement, tout programme peut etre vu comme une formule definissant les sorties a partir des entrees. L'utilisation de variables auxiliaires locales a de nombreuses applications en mathematiques: d'abord, cela peut permettre d'utiliser dans la demonstration d'un lemme des noms de variables qui, a l'exterieur peuvent avoir un autre sens.

Mais, cela permet d'unifier des presentations autrement bien distinctes: La description de la division euclidienne est generalement distincte de la methode de calcul habituelle. Si on enonce:

THEOREME: Etant donnes deux polynomes A et B (B#0), il existe des polynomes Q et R tels que A = B Q + R et deg(R) < deg(B).

La demonstration peut etre:

Considerons le programme:

```
Q:0 , R:A ,
tant que deg(R) >= deg(B) faire
          (C: X^(deg(Q)-deg(R)) * prem_coeff(Q)/prem_coeff(B),
           Q: Q+C, R : R - C*B);
```

montrons (c'est facile) qu'il s'arrete au bout d'un temps fini, et que lors de l'arret Q et R satisfont les conditions de l'enonce du theoreme: cela demontre le theoreme et donne en meme temps la methode de calcul manuelle habituelle: celle-ci simule le programme (ici ecrit en MACSYMA francise), les divisions de la feuille de papier correspondant au contenu des variables B,Q, et R, le contenu de R etant la derniere ligne ecrite sur la partie gauche.

IV- DEMONSTRATIONS CONSTRUCTIVES ELEMENTAIRES

L'exemple precedent en est une: un autre exemple est le theoreme suivant qui implique le theoreme de structure des sous-modules d'un module libre et le theoreme de structure des modules de type fini sur un anneau principal (partie existence).

THEOREME: Soit M une matrice mxn sur un anneau principal; il existe des matrices inversibles R et S telles que la matrice N := R M S soit definie, et que ses seuls elements non nuls soient sur la diagonale et verifient N[i,i] divise N[j,j] pour tout i < j.

Demonstration: Faire operer une matrice 2x2 sur deux lignes ou deux colonnes de M revient a multiplier celle-ci a gauche ou a droite par une matrice inversible: les echanges de lignes ou de colonnes sont des cas particulier de telles operations, ainsi que le fait d'ajouter a une ligne (resp. une colonne) un multiple d'une autre ligne (resp. colonne). Il suffit donc de montrer que l'on peut transformer M en N a l'aide de telles operations: c'est ce que va faire l'algorithme suivant, ecrit cette fois en pseudo-PASCAL.

```
procedure reduction ( M:tableau [1..m,1..n] de elements);
        var i,j,k:entier;

        fonction pgcd ( x,y:elements ): tableau [1..2,1..2]
                                          de elements;
                 {Cette fonction retourne une matrice 2x2
                 inversible A telle que A[1,1] x + A[1,2] y soit
                 le pgcd de x et y et que A[2,1] x + A[2,2] y =0;
                 si l'anneau est euclidien, l'algorithme
                 d'Euclide permet de calculer A.}

        fonction chercher (i,j,k:entier) : booleen;
                 {Cette fonction cherche j > i et k > i tels que
                 M[j,k] soit non nul et retourne vrai en cas de
                 succes}

        fonction chercher_non_mult (i,j,k:entier) :booleen;
                 { agit comme chercher mais cherche M[j,k] non
                 multiple de M[i,i] (avec j>i, k>i) }

        fonction chercher_ligne (i,k:entier) : boolean;
        fonction chercher_colonne (i,k:entier) : boolean;
                 { cherche k>i tel que M[i,k] (resp. M[k,i]) <>0 }

        procedure echanger_ligne(i,j:entier);
        procedure echanger_col(i,j:entier);
                 { l'action de ces procedures est claire }

        procedure ajouter_ligne (i,j:entier );
                 { ajoute la ligne i de la matrice M a la ligne j}

        procedure oper_ligne (A:matrice[1..2,1..2] de elements;
                                          i,k:entier);
        procedure oper_col (A:matrice[1..2,1..2] de elements;
                                          i,k:entier);
                 {fait operer A sur les lignes (resp. les
                 colonnes) i et k de M; oper_col revient a
                 multiplier a droite par une matrice obtenue a
                 partir de la transposee de A}

{*****      C'est ici que commence le calcul        ******}

        begin
        i:=0;
        tant que chercher (i,j,k) faire
             begin
             i:=i+1;
             echanger_lignes(i,j);
             echanger_colonnes(i,k);          {donc M[i,i] <> 0 }
             j:=0;
             repeter
                  si j <> 0 alors ajouter_ligne(j,i);
                  repeter
                       tant que chercher_ligne(i,k) faire
                            begin
                            A := pgcd (M[i,i],M[i,k]);
                            oper_col(A,i,k)
                            end;                { M[i,k] = 0 }
                       tant que chercher colonne(i,k) faire
                            begin
                            A := pgcd (M[i,i],M[k,i]);
                            oper_ligne(A,i,k)
                            end                 { M[k,i] = 0 }
                  jusqu'a non chercher-ligne(i,k);
             jusqu'a chercher_non_mult(i,j,k);
        end; end.
```

Il est presqu'immediat que cette procedure s'arrete quand la matrice M a la forme voulue pour la matrice N. Que les iterations ne soient pas infinies provient de ce que les operations sur les lignes et les colonnes remplacent M[i,i] par un de ses diviseurs, et qu'un anneau principal est noetherien.

Outre le fait d'etre constructive, cette demonstration presente l'avantage sur les demonstrations habituelles de ne pas necessiter de notions auxiliaires (algebre exterieure,...). Un mathematicien traditionnel repugnerait a cette demonstration car il ne saurait pas faire la description des operations effectuees sur la matrice M en un langage suffisamment precis pour permettre une demonstration.

V- IMPORTANCE RELATIVE DES NOTIONS.

La presentation habituelle de l'algebre lineaire privilegie la notion de determinant et de formule de Cramer. Or seule la reduction de Gauss (triangularisation) permet des calculs raisonnables: un bon exercice rarement soumis aux etudiants est le suivant:

On se propose de calculer un determinant 50x50 par 3 methodes differentes: estimer le nombre d'operations necessaires et le temps de calcul, a raison d'un milliard d'operations a la seconde (bien mieux que les meilleurs ordinateurs actuels); pour la premiere methode, on utilisera comme unite de temps l'age de la terre: pour la troisieme on se contentera de cent mille operations seconde (realiste). Les methodes sont le develloppement complet, le developpement par rapport a une ligne, les mineurs etant calcules par la meme methode, en evitant de les calculer plusieurs fois, et enfin la triangularisation.

Cet exercice montre bien que cette derniere methode qui peut etre enseignee sans introduction de la notion de determinant est de loin la meilleure. Bien plus elle permet, outre le determinant, de calculer et definir le rang de la matrice et de resoudre les systemes d'equations lineaires. Or mon experience prouve que la plupart des calculs de determinants servent a calculer des rangs de matrice. Ainsi, ce point de vue algorithmique conduit a enseigner l'algebre lineaire sans les determinants (en premier cycle), mais en definissant le rang directement. Bien entendu, la notion de determinant garde son interet a un niveau superieur, notamment pour demontrer l'unicite de certains resultats.

VI- CONCLUSIONS.

J'espere, par ces quelques exemples, avoir montre que l'enseignement et la redaction mathematique ont enormement a profiter du developpement de l'informatique et de la pensee informatique; cette influence va plus loin et est bien plus profonde que le role bien connu de l'experimentation sur calculette ou logiciel d'"EAO".

27 mars 1985.

REFLEXIONS SUR CERTAINES BASES MATHÉMATIQUES DE L'INFORMATIQUE

Contribution à la question n° 10 du §2
du texte d'orientation de la CIEM

JACQUES STERN
Université de CAEN

Il est bien évident aujourd'hui que les mathématiciens ne peuvent ignorer l'informatique ; il en résulte que tous les étudiants en mathématiques et en particulier ceux qui se destinent à l'enseignement, doivent avoir été exposés à la pratique d'un langage de programmation évolué. Cela dit, il n'est guère possible d'en rester là : on ne peut éluder la question des fondements théoriques de l'informatique puisque c'est en somme un point de passage obligé pour relier la pratique mathématique à la pratique informatique.

En France, une reflexion sur ce sujet a abouti à la création cette année, d'une épreuve optionnelle d'informatique à l'agrégation de mathématiques, concours de recrutement d'enseignants qui est, on le sait, de niveau élevé. L'auteur du présent texte est parfaitement d'accord avec les thèmes qui ont été retenus pour constituer le programme de cette épreuve ; il prépare actuellement, en collaboration avec C. FJECH, un ouvrage dont le contenu sera très proche et il présente ici quelques réflexions personnelles qui ont accompagné la première phase d'élaboration de l'ouvrage. Il va de soi que ces réflexions n'engagent en rien les membres du jury de l'agrégation ni les collègues qui ont rédigé le programme.

Il est bien clair tout d'abord qu'un approfondissement des base théoriques de l'informatique ne transforme pas un mathématicien en informaticien ; c'est plutôt une sorte de "conversion mentale" qui met celui qui l'a pratiquée en mesure d'appréhender les réalités informatiques ; à cet égard donc, cet approfondissement est peut être plus adapté encore à la formation en mathématiques qu'à la formation en informatique.

§1. AUTOUR DE LA NOTION DE CALCUL

La théorie de la calculabilité a, auprès de certains, la réputation d'être pénible et formelle ; pourtant, la première tâche du théoricien est bien de délimiter ce qui est calculable (ou effectif comme on dit parfois) de ce qui ne l'est pas. On peut bien sûr se contenter de déclarer calculable tout ce qui est susceptible d'un traitement machine. Tout en n'étant pas dénué de sens, ce point de vue est vague et ne permet pas de tester la "robustesse" de la notion ainsi isolée. Par ailleurs, ce point de vue est historiquement incorrect ; on peut citer de nombreux travaux de la plus haute importance sur la calculabilité antérieurs à l'apparition des premiers ordinateurs : par exemple ceux de Turing [1936], de Kleene [1936], de Post [1936] sur la calculabilité, mais également ceux de Mc Culloch et Pitts [1943] sur la modélisation des systèmes de neurones, qui a donné naissance à la théorie des automates.

C'est précisément la théorie des automates que nous proposons comme point de départ ; elle présente l'avantage d'être une théorie simple et bien développée ; elle se prête bien à un traitement informatique : on peut par exemple simuler l'action d'un automate par un programme écrit dans un langage comme PASCAL. La théorie des automates permet également de présenter les premiers rudiments d'algorithmiques et d'évaluation de complexité (par exemple en comparant divers algorithmes de minimisation) ; elle autorise aussi une introduction relativement simple du concept de non déterminisme. Enfin, et ce n'est pas un argument négligeable, elle s'applique : aux éditeurs de texte et à l'analyse lexicale en particulier.

Cela dit, la modélisation des machines par les automates aboutit à un constat d'échec ; par la considération de langages simples non reconnus par automates mais aussi par l'observation évidente qu'une notion centrale en informatique est évacuée, celle de capacité mémoire. Il faut donc reprendre le problème et il est raisonnable de montrer que différentes méthodes permettent de définir la même notion de calculabilité ; ce qui prouve le caractère naturel de cette notion et ce qui étaye la thèse de Church affirmant l'égalité du "récursif" et du "calculable".

On citera quatre voies d'approche :

1) L'adjonction aux automates d'une capacité mémoire, ce qui conduit aux machines de Turing.

2) L'abstraction directe des calculateurs, qui conduit à la notion de machine à accès direct (ramdon access machine, cf. Cook et Reckhow [1973]) controlée par l'intermédiaire d'un langage simple, type langage machine.

random

3) La définition d'une classe simple de programmes, par exemple ceux qui sont écrits dans un PASCAL réduit au type entier et aux structures de contrôle IF... THEN... ELSE et WHILE... DO.

4) La définition des fonctions récursives, qui peut se faire en adoptant un point de vue "programmation fonctionnelle" et des constructions analogues à celles du langage LISP.

La démonstration de l'équivalence entre ces diverses définitions est à bien des égards instructive. Par exemple, la simulation d'une machine à accès direct par une machine de Turing est un bon exercice de gestion d'une mémoire à accès séquentiel. On peut noter à ce sujet que les technologies nouvelles impliqueraient des solutions différentes et envisager des machines de Turing où seule l'écriture est permise comme sur les disques optiques (write-only memories).

Une fois dégagée, la notion de fonction calculable et donc de problème décidable, il est raisonnable de parler de problème indécidable : la construction d'une machine universelle ne demande plus guère d'effort ce qui permet de poser le "problème de l'arrêt". On est ensuite amené à examiner si cette dichotomie décidable / indécidable est réellement opérante ce qui conduit naturellement à la notion de temps de calcul. Les simulations des diverses machines entre elles, montrent le caractère stable du "temps polynomial" ; se trouve ainsi définie la classe P qui autorise une abstraction convenable de la "faisabilité".

§2. AUTOUR DE LA NOTION D'ALGORITHME

La notion de temps de calcul, dégagée au plan théorique doit être aussi appliquée au niveau pratique. Une revue de certains algorithmes permet alors d'une part d'acquérir une certaine pratique pour la conception des programmes, d'autre part de s'entraîner à des calculs pratiques de complexité, aussi bien en moyenne que dans le plus mauvais cas. C'est l'occasion aussi de présenter dans un cadre assez général, certaines techniques de combinatoires (cf. Knuth [1973]) : statistique des permutations et des distributions, séries génératrices, analyse asymptotique.

Par ailleurs, au fur et à mesure qu'on présente les algorithmes, on peut également introduire les structures de données de l'informatique : piles, files, listes, arbres, graphes et leurs diverses représentations.

Ce qui suit constitue une liste non exhaustive d'algorithmes qu'on peut présenter.

1. ALGORITHMES DE TRI

Tri par insertion,
Tri bulle,
Heapsort,
Quicksort.

La présentation de ces algorithmes de tri, conduit déjà à de nombreuses observations instructives : on doit justifier le fait qu'on fait essentiellement le décompte des comparaisons, on doit introduire des structures de données originales (dans heapsort en particulier)... Les exemples choisis illustrent également la différence entre complexité moyenne et complexité dans le plus mauvais cas.

2. ALGORITHMES DE RECHERCHE

Recherche séquentielle
Utilisation d'arbres de recherche binaires.
Utilisation d'arbres AVL ou d'arbres 2-3.
Hachage.

Là encore, des structures de données originales sont présentées.

3. RECONNAISSANCE DE MOTIFS

Algorithme de Knuth Morris et Pratt
Algorithme de Rabin Karp

On peut faire ici le lien avec les automates finis.

4. ALGORITHMES DES GRAPHES

Arbres de recouvrement maximaux
Plus court chemin
Clôture transitive.

A propos de ces algorithmes, on peut exposer le principe et les avantages de la recherche en profondeur.

Il est naturellement possible de parler aussi de multiplication de matrices, de transformation de Fourier rapide, etc. Cependant, il est peut être dangereux de multiplier les exemples, en particulier ceux qui ne mettent pas en jeu des concepts informatiques nouveaux.

Il convient à ce point d'aborder le sujet des algorithmes non polynomiaux, par exemple à travers l'algorithme de résolution pour le calcul propositionnel, ce qui conduit au problème SAT de Cook [1971].

La classe NP et la notion de problème NP complet peuvent être présentées sans difficulté à partir des machines de Turing ou des machines à accès direct fonctionnant en mode non déterministe. Le principal travail consiste à établir le théorème de COOK [1971], après quoi, on peut se doter d'une première panoplie de problème NP complets (cf. Garey, Johnson [1978]), par exemple :

Problème SAT,
Problème du voyageur de commerce,
Problème du circuit hamiltonien,
Problème des cliques,
Problème du sac à dos.

On peut, pour finir, donner quelques indications sur la façon d'aborder les problèmes NP-complets par exemple présenter une heuristique pour le problème du voyageur de commerce à partir d'arbres de recouvrement.

§3. AUTOUR DE LA LOGIQUE : SYNTAXE ET SÉMANTIQUE

Une première approche des problèmes de syntaxe est fournie par la théorie des langages algébriques développée à partir des grammaires algébriques. Naturellement, le lien avec une approche liée à la calculabilité est fait par l'intermédiaire des automates à pile. Il est bien clair qu'il convient ensuite de présenter l'utilisation des grammaires dans l'analyse syntaxique.

La notion d'arbre de dérivation pour les grammaires algébriques constitue également une préparation pour aborder les concepts de la logique en particulier les règles de déduction. En effet, de façon surprenante, les concepts de base de la logique sont quelquefois un motif de panique pour les mathématiciens. La présentation de la logique et en particulier du théorème de complétude doit naturellement être constructive et adaptée à l'informatique. Les fonctions de Skolem permettent de ne considérer que des formules $\forall\exists$. Par le théorème de Herbrand, on se ramène à des conjonctions de clauses auxquelles on peut appliquer l'algorithme d'unification ; ceci conduit à l'algorithme de résolution de Robinson [1965]. Il convient bien sûr de ne pas masquer le phénomène d'indécidabilité. La procédure de Herbrand ne se termine pas forcément. Cela dit, on peut souligner l'utilité de la résolution en évoquant le langage PROLOG.

Ces préliminaires de logique étant acquis, on peut brièvement introduire deux sujets assez délicats.

1. La sémantique des procèdures récursives et l'approche "point fixe" des programmes.

2. La vérification de programmes par assertions et les règles de Hoare [1969].

CONCLUSION : On a essayé de montrer dans ce qui précède, la logique qui sous tend la délimitation des sujets proposés comme bases mathématiques de l'informatique. Il est clair que le contenu décrit plus haut est appelé à varier très rapidement, compte tenu des développements de l'informatique. Peut-être faudra-t-il par exemple y incorporer des outils théoriques pour l'étude des bases de données ou des circuits VLSI. Quoi qu'il en soit, sous une forme ou sous une autre, il devrait s'introduire progressivement dans l'enseignement mathématique de nos Universités.

BIBLIOGRAPHIE

AHO A.V., J.E. HOPCROFT, J.D. ULLMAN [1974]. The design and Analysis of computer Algorithms, Addison Wesley, Reading, Mass.

AHO A.V., J.E. HOPCROFT, J.D. ULLMAN [1983]. Data Structures and Algorithms, Addison Wesley, Reading, Mass.

AHO A.V., J.D. ULLMAN [1977]. Principles of compiler Design, Addison Wesley, Reading, Mass.

COOK S.A. [1971]. The complexity of theorem proving procedures. Proc. Third Annual ACM Symposium on the Theory of Computing, 29-33.

COOK S.A., R.A. RECKHOW [1973]. Time bounded random access machines, J. Computer and Systems Science 7, 354-375.

GAREY M.R., D.S. JOHNSON [1978]. Computers and Intractability, a Guide to the theory of NP-completeness, H. Freeman, San Francisco.

HOARE C.A., [1969]. An Axiomatic Basis of Computer Programming, C.ACM 12, 576-580.

KLEENE S.C. [1936]. General recursive functions of natural numbers, Mathematische Annalen 112, 727-742.

KNUTH D.E. [1973]. The Art of Computer Programming Vol III : Sorting and Searching, Addison-Wesley, Reading, Mass.

MANNA Z. [1974]. Mathematical Theory of Computation, Mc Graw Hill, New York

Mc CULLOCH W.S., W. PITTS [1943]. A logical calculus of the ideas immanent in nervous activity, Bull Math. Biophysics 5, 115-113.

POST E. [1936]. Finite combinatory processes - formulation I, J. Symbolic Logic 1, 103-105.

ROBINSON J.A. [1965]. A Machine-oriented Logic Based on the Resolution Principle J. ACM 12, 23-41

ROGERS H. Jr [1967]. Theory of Recursive Functions and Effective Computability, Mc Graw Hill, New York.

SEDGEWICK R. [1983]. Algorithms, Addison Wesley, Reading, Mass.

WIRTH N. [1976]. Algorithms + Data Structures = Programs, Prentice Hall, Englewood Cliffs, N.J.

INTERRELATIONS BETWEEN COMPUTERS, STATISTICS, AND TEACHING MATHEMATICS

Rolf Biehler, IDM Universität Bielefeld

1. Introduction

This paper will discuss some of the issues raised by the ICMI background paper on "The Influence of Computers and Informatics on Mathematics and Its Teaching" (ICMI 1984) from the perspective of statistics, data analysis and its teaching.

The change in statistics in connection with the availability and development of the modern computing technology is certainly relevant for the statistics curricula at the secondary and tertiary level although a direct projection of new developments into curricula, in particular at the secondary level, is neither feasible nor desirable. Furthermore, mere imitation of computer use in a scientific context in the classroom neglects the possible advantages of using computers to support the learning and teaching of statistics (see RÅDE 1983). But there is also a somewhat more fundamental reason for analyzing the relationship between computers and statistics. The background paper discusses the effect of computers on mathematics but conveys the impression that there may have been a certain bias towards pure mathematics in discussing this topic. The influence of computers on applied mathematics seems to be not only older than its influence on pure mathematics, but perhaps even more profound.

To include applied mathematics, the mathematical sciences and the diversity of uses of mathematics in other sciences and in society in a broad picture of mathematics is necessary, at least where mathematics education at the pre-university level is being discussed. In most European countries, the pre-university level has to provide a type of general education. Even university-bound courses have to lay foundations not only for studying mathematics and computer science, but also for all other sciences where new efforts of mathematizaton go hand in hand with "computerization". Therefore the image of mathematics underlying those curricula has to be somewhat more general than the understanding of mathematics shared by pure mathematicians. In this sense I disagree with the ICMI paper, because it seems to assume that the reference to "the discipline of mathematics" provides a unifying factor for curricula in some direct way. (cf. ICMI 1984, 165).

In the following, I shall discuss algorithmics, the emergence of new mathematics, visualizations, experimentation and the status of proof. These are topics which are mentioned in the ICMI paper. Although they will be discussed mainly from the perspective of statistics, it is hoped that some of the aspects can be generalized to other domains, as well.

2. Algorithmics

The ICMI paper emphasizes "the influence of fundamental concepts of informatics, in the forefront of which is found algorithmics" (p. 162). It would be very interesting to analyze more closely in which respect "algorithm" is an old mathematical concept, and in which respect informatics has contributed new aspects to this concept. In any case, however, the central importance of algorithms for computers and computer science has influenced mathematical research and applications in two ways which in part contradict one another: On the one hand, it has led to a new emphasis on mathematical research on the design and analysis of algorithms (e.g. numerical analysis, computer algebra; complexity theory, probabilistic analysis of algorithms). On the other hand, the availability of computer software, i.e. the possibility of making machines execute already constructed algorithms, has led to deemphasize algorithms, insofar as executing algorithms by hand has to a large extent become superfluous. For different people, these two aspects are of different importance. With regard to curricula, it is far from being clear at present, how these two aspects should be balanced out. ENGEL's (1985) suggestions for an integrated course in statistics and informatics emphasize algorithms very much, whereas JOINER (1983), e.g., suggests using statistical software in the curriculum to get leisure to concentrate on other aspects of the statistical analysis process, such as learning to judge when certain statistical procedures are appropriate, which assumptions are critical, how to check for violations of assumptions, how to interpret results etc.

Although the availability of statistical software may, in principle, provide leisure for such aspects, it is frequently not used in this way. At the beginning of the computerization of statistics, YATES (1966, 250) gave severe warnings against the concept of "data processing", which certainly stems from computers and computer science. This concept was associated with the idea that having data processed by a computer will suffice to get meaningful results. It would seem that two factors which still prevail supported this "philosophy" in statistics. Historically, batch processing fostered an understanding of the statistical analysis process as input - processing (algorithm) - output. Although the technological development now, in principle, allows a more interactive approach, this is not always put into practice. But there seems to be a more fundamental reason. YATES refers to a fairly wide spread understanding or ideal of statistics (which has also been critizised by the late R.A. FISHER), saying, that it is possible to model pratical situations before looking at data such that the analysis of data will be reduced to mere application of an algorithm selected in advance.

It is not by chance, that TUKEY (1972), while discussing the relationship between data analysis and computation, emphasizes the distinction between "data processing" and "data investigation". TUKEY is one of the promoters of the so-called Exploratory Data Analysis (EDA) (cf. e.g. TUKEY 1977, BIEHLER 1982, HOAGLIN et al. 1983), which emphasizes an interactive style of data analysis. This style says that it is not reasonable to make the process of analysis completely automatic, the human analyst has to make decisions in many stages of the process, he must decide which algorithm is to be applied next on the basis of data plots, for instance, (see below). Such an interactive style was much supported and extended because of the opportunities provided by modern computing facilities. To practice this style with large and high-dimenstional data sets would otherwise be not possible. In this sense, statistics has also become "less algorithmic" due to the "influence" of computers.

To sum up, two things may be learned from this development. First, even if algorithms are executed by machines, the emphasis on data processing may lead to an inadequate shift of emphasis to processing, which is only part of the whole of analysis process. This may be even more so if the algorithmic standpoint in statistics is stressed in an one-sided manner. Second, there is no unidirectional influence of computers on statistics (mathematics). Rather, it depends on the understanding of statistics (mathematics) how computer may influence the subject, just as the possibilities of computers may influence the understanding or philosophy of the subject.

3. The Emergence of New Mathematics

The ICMI paper says that one "can expect" that the new methods of computation will lead to the emergence of new mathematical concepts (p.162). This has already occured to a large extent. The development of statistics during the last, say, 25 years, is a very rich source of examples for this proposition; take, for instance, topics concerned with robustness and all the techniques and concepts for the analysis of multivariate data. EFRON (1979) gives further examples concerning nonparametric methods for the analysis of one-dimensional data. All this provoked new theorems and concepts that meet every standard of pure mathematics. But it is highly questionable whether this is the only or the main perspective from which these developments should be viewed.

First of all, consciously realizing the revolutionary impact of computers on statistics seems to be rather old as compared to other domains of mathematics (see e.g. TUKEY 1965, YATES 1966). This is due to the fact that statistics and other fields of applied mathematics always have been computationally intensive, i.e. tools for computation have always been an important aspect.

YATES (1966), who views computers as the second revolution in statistics, reminds of the desk calculator, the first one. YATES says that not only new statistical theory emerged in connection with computers, but that a change of statistical methodology took place as well. He mentions, among other things, the following types of "progress": speed of analysis (more analyses in a given time), comprehensiveness of analysis (using all available data, comparing with similar data), easy reanalyses with transformed data, calculation of residuals to check for violations of assumptions and their correction, editing of data and error correction, using more adequate methods for a problem not used before because of large computational cost. YATES expected further progress from devices of automatic data recording and from display facilities, which "may revolutionize the presentation of much statistical material" (YATES 1966, 249). The change YATES described 20 years ago has become even more profound than expected. MENGES (1980, 182) summarises that the computer has contributed towards overcoming "the unholy trinity of independence, homoscedasticity and Gaussianity (assumption of normal distribution)" of traditional statistics, i.e. new methods were developed to meet the challenges to statistics raised by other disciplines and by practice more adequately. This seems to be similar in other parts of applied mathematics. EFRON (1979, 439) characterizes the qualitative nature of the change by pointing out that the computer has led to a redefinition of what is "simple", and not just to an extension of relevant concepts and methods. Although it is not reasonable to equate simplicity on a scientific level with simplicity at a curricular level, it is worthwhile to explore whether there could be a gain with regard to the contents and the sequence of topics to be taught in school. ENGEL's (1985) suggestions for including nonparametric methods at an early stage will have to be considered in this context.

The beginning change of statistical methodology which YATES pointed out 20 years ago has led, among other things, to the already mentioned Exploratory Data Analysis (EDA). The movement for "l'analyse des données" in France (cf. BENZECRI 1980) goes in a similar direction.

While computer-influenced, robust, and non-parametric methods remain, to a large extent, within the traditional paradigm of inference statistics, i.e. testing hypotheses, estimating parameters and confidence intervals on the background of probabilistic models, EDA breaks with this paradigm. Its aim is to obtain insight into structures and anomalies of data, to discover new phenomena in data, to aid in the generation of new hypotheses. In order to reach this aim, the data are transformed, represented in a variety of graphical displays, compared to several models, etc. This strategy is adequate, for instance, in situations where little is known about the data and the underlying system and only vague questions can be asked. The application of EDA is not confined to random samples. It gives more freedom for exploring data than classical methods; the price paid is that the amount of uncertainty of its results, e.g. their stability or error characteristics of the procedure usually cannot be calculated. A direct controlled inference from sample to population like in traditional statistics is not possible. Results can stimulate subject matter considerations, instead, or have to be tested by traditional methods on other data. (A more detailed analysis of EDA's methodology is given in BIEHLER 1982.)

It is worth mentioning that the techniques of EDA extend from complex displays and algorithms for multidimensional data to very simple displays and techniques for the analysis of one- or two-dimensional data. The simplicity of some of its techniques has attracted the interest of mathematics educators who see several good reasons for including elements of EDA, its style and its displays, in curricula on statistics. Among others, the enrichment of curricula by analyses of real data, the enrichment of descriptive statistics, which is usually taught in a boring and formal way, are emphasized, as well as the possible gains for a better understanding of probability and traditional statistical methods, i.e. their specific advantages and problems in real applications (cf. GNANADESIKAN et.al. 1983, BIEHLER 1983, 1984).

Such a type of exploratory mathematics, which calls for more computation time and more time for constructing displays, to the extent of several orders of magnitude, than traditional methods, seems to be a typical child of the computer age. Although the computer is used for exploration, and to aid discovery in other parts of mathematics, too, the results of EDA usually are not conjectures which can and should be proved mathematically. Rather, they are more directly related and relevant to subject matter fields outside mathematics, i.e. they have to be handled and tested like other conjectures in empirical sciences.

One final remark to avoid misunderstanding is necessary. A certain practice in empirical sciences of feeding data into a computer, runing a battery of tests and then picking out only those results which are statisticaly significant is often correctly critizised with vehemence. This is not EDA! On the one hand, EDA is a result of the realization that in many situations it is reasonable to look at data first in order to generate hypotheses. The classical norm that hypotheses must be formulated before data are analyzed cannot be universally applied. On the other hand, EDA does not assign the stamp of statistical significance to its results, i.e. does not give them wrong authority. In this sense, the methodology of EDA, which cannot be described in detail here, tries to regulate an otherwise unregulated "data snooping" with computers. In this sense there may be some similarity to R.A. FISHER's important contributions to statistics, which also provided guidance to an formerly rather arbitrary practice with desk calculators in statistics (cf. YATES 1966, 235).

4. Visualizations

Geometrical diagrams and graphical displays have always played a certain role in mathematical research and in communicating mathematics to other mathematicians and to people outside mathematics. Their importance has varied with the type of mathematics and the social context in which mathematics is used. The graphical capabilities of modern computers have brought new life to the use of visualizations in research and communication.

Statistics, too, has seen a renaissance of graphical methods during the last, say, 15 years (cf. BENIGER/ROBYN 1978, WAINER/THISSEN 1981, CHAMBERS 1983, BIEHLER 1985). The use of graphics to communicate statistical information effectively to a broad public has a longstanding tradition, but the increasing amount of information available in science and society together with the new technological opportunities has led to many efforts to invent new representations and to use them more effectively. It would, however, be a serious misunderstanding to view graphical displays still merely as a tool for "showing the obvious to the ignorant". The most important change is a new appreciation of graphical displays as a genuine research tool in statistics, especially in EDA.

In EDA, plots of data, or of results after an algorithm has been applied, are the most important method for revealing structure and anomalies (e.g. "outliers") in data. It is more than 20 years ago that TUKEY (1962, 49) formulated the thesis that graphical displays are the most important tools for "revealing the unexpected". In the meantime a huge number of new displays have been invented, designed for the specific purposes of exploring data and taking into account the specific features of the human eye-brain system. The speed of picture generation on computers and the facility of changing them quickly makes it possible to construct a large variety of plots for one data set to explore it form several points of view. Also, in many cases, graphical displays are the only means for communicating the features discovered, which often cannot be described by analytic models and equations.

From a more general point of view, graphical displays can be viewed as a means for "keeping in live rapport with the data" (cf. Pearson 1956, 135). This aspect has become more and more impportant since feeding data into computers has led to a certain alienation between data and data analyst.

Although the use of visualizations in statistics and EDA may have several specific features, the use of graphical displays for digesting large amounts of numbers is of more general importance, as is exemplified by their use for studying solutions of differential equations, for analyzing Julia sets, and for studying results of simulations of any kind. This may be interpreted as the necessary "continuous" complement to the "discrete" digital computer. Also, the "holistic" features of displays counterbalance the linear-sequential thinking of algorithmics.

With regard to curricula, some general consequences may be drawn. Visualizations have always played a role in the everyday practice of mathematics teaching, as they are considered as a means to facilitate the learning of abstract mathematics. The new developments present a challenge to this practice in several respects. First, the computer increases the possibilities for visualizations in the classroom but, perhaps more important, allows interaction with displays which is very difficult without computers in many situations. Visualizations can become a tool for active learning, rather than being illustrative material presented by the teacher. Second, graphical displays should not only be seen as means for better teaching and learning, but the new role of graphics as research tools of mathematics should be reflected in the curricula to a certain extent. In most cases, graphs are no substitute for abstract mathematics, but serve somewhat different objectives than other representation systems in mathematics. Third, the partly high complexity of modern computer graphics calls for a more theoretical understanding as a prerequisite for an adequate use of displays. This supports the results of educational research into the students' difficulties even with presumably simple graphs of school mathematics, which show that most of the graphs are not self-evident or self-explanatory. Therefore, curricula should provide a graphical education, an education for "graphicacy", as it is sometimes called. This has become even more desirable because of the increasing importance of graphs and visualizations outside school (for a more detailed discussion, cf. BIEHLER 1984a)

5. Experimentation and the Status of Proof

The ICMI paper well states that computers have greatly increased the possibilities of experimentation in mathematics and affected the status of proof. EULER has been quoted to make clear that pure mathematics in its research practice has always been a mixture of inductive and deductive reasoning resp. observation and demonstration. I agree, but I claim that the new developments cannot fully be described on these lines, at least not if one includes applied mathematics in a broad picture of mathematics.

First of all, the status of proof in applications of mathematics or parts of applied mathematics has always differed from that in pure mathematics for two reasons. Proofs are not sufficient to substantiate a statement. In applications, one always must give additional reasons for the assumptions on which a proof is based. Additionally, proofs, i.e. complete formal deductions from presuppositions, are sometimes considered to be unnecessary, if there are other means for evaluating statements in the context of a field of application.

I think, it cannot be denied that the use of computers has increased the amount of "mathematics without proof". How this phenomenon is judged depends on the point of view. Let me take one example from statistics: the case of robust estimators. The famous Princeton Robustness Study (ANDREWS et.al. 1972) analyzed sets of estimators under a system of different modelling assumptions by means of Monte Carlo simulation on a computer. Although the results also stimulated new mathematical concepts and theorems, the results themselves have not been "reproduced" or validated by mathematical proof. Although one may wish or hope that this will be done some time, the most important point is that the results have already influenced the practice of analyzing data. The results are important in themselves and can, to a certain extent, be validated by their success in applications outside mathematics. A result like estimator A has a lower variance than estimator B under some assumptions (and therefore should be preferred in practice) is not proved to be true. Monte Carlo simulations only give results which are, loosely speaking, true with a very high probability. From a practical standpoint this adds only a marginal amount of uncertainty in an application situation where one never can be certain that modelling assumptions are true. Similar arguments are put forward in discussions about the problems of "probabilistic proofs". But the problem is still more complicated. For one thing, the question what constitutes an empirical validation of statistical methods has to be clarified and is already debated; see STIGLER (1977) and the related discussion, in which ANDREWS (1977, 1079) and HOGG (1977, 1089) point out in which sense robust estimators have "proved" to be superior in the exploration of multi-dimensional data. MALLOWS (1979) gives several instructive examples illustrating this claim. There is another aspect of the problem that can be related to the discussion of the computer-proof of the 4-colour-theorem. ATIYAH (1985) makes the point that the function of proof is not only to make sure that a statement is true but also to give insight into why a statement is true. Also, a proof usually establishes relations between a statement and an existing body of knowledge, thereby providing some kind of "understanding". TUKEY (1979, 106) discusses this problem in relation to robust estimators, pointing out that understanding is good when one can get it, but from a pragmatic standpoint, understanding cannot be the only relevant aspect. But even if proofs are given like in the asymptotic theory of robust estimators, they do not provide sufficient guidance for practical applications because the assumptions usually do not cover completely the cases met in practice. Therefore, the experience with procedures on selected examples and the experiences with methods under realistic conditions are a necessary supplement. This seems to be the case in numerical analysis as well, where algorithms are applied often without complete proof, and where their performance on well-chosen "exemplars" is equally important for the user as are some mathematically proved results. In this sense, the German mathematician H. WERNER (1982, 23), an expert in numerical methods, speaks of a "spectrum of types of verification for numerical methods". TUKEY (1983, 5), discussing the future relationships between statistics and computers, puts forward the general thesis: "We will, however, have to admit how often and how far we go beyond the CERTAINTY of MATHEMATICS." This seems to be an unavoidable feature rather than a relapse to a "prescientific mathematics".

It can be supposed that the phenomena described are not specific for statistics. In various sciences, the relative importance of studying models by computer simlation has increased dramatically. The change is sometimes interpreted as being so fundamental that one can compare it to the Galileian revolution: the two principal research methods of natural science (experimental and theoretical) have now been supplemented by a third: the use of methods from computational and information science (including simulation). This thesis is put forward by, N.F. LANE, the chairman of the Advisory Committee for Advanced Scientific Computing of the National Science Foundation in the United States (see Physics Today, May 1984, 61). It is not unplausible that the already existing large gap between those phenomena discovered by simulation and those which have been proved mathematically will become larger and larger. Although this situation is a real challenge for mathematical theory to catch up, the simulation results have often stimulated insight and theory building in the natural and social sciences without having been proved mathematically. Certainly, one could define mathematics in a way to exclude such activities and results from "proper" mathematics, but for one thing, the objects (models) studied can often be defined in a precise mathematical way and do not refer just to one discipline like biology or sociology but have a similar general range of applicability just as other parts of mathematics. And, more importantly, as has been pointed out in the introduction above, the definition of mathematics, the drawing of borderlines, as seen from the perspective of general mathematical education, should not be as narrow as in selfreflections of some mathematicians.

What do these ideas imply for secondary mathematics education? I think that the role of proof in mathematics education has always been different from that in pure mathematics, too, e.g. because of the closer relation of school mathematics to applicatios and because of the impossibility of sticking to a pure and "complete" logical deductive style. Relying on intuitions and experiences of different kinds has always been necessary to a certain extent. Now the computer may give more help and more reason for such an approach, while a more experimental or empirical approach to mathematics may simultaneously provide a more adequate background for reflecting on the status, usefulness, limits and advantages of proof in selected situations as well, instead of presupposing the "necessity for proof" as an unquestioned ritual of mathematics.

6. Teaching Statistics and Computers

As has been pointed out at the beginning, the reflections on the interrelations between statistics and computers may be more generally relevant then only for statistics education. Nevertheless, possible challenges for redesigning the teaching of statistics have been mentioned in this text several times. But some aspects have to be added. Because of its importance it has to be repeated that a clear distinction between the (upper) secondary level and the tertiary level (including college level in Nothern America) must be made. Both the motivational and cognitive prerequisites of students and the educational objectives of the two levels are too different to be treated alike. The following refers to the secondary level.

Any innovation in teaching statistics has to take into account the level of teacher knowledge and competence. In the Federal Republic of Germany, it is still a problem that the majority of mathematics teachers in schools had no courses in probability and statistics at the university level. And if they did, aspects of applied statistics (let alone of EDA) played only a minor role. Therefore, even if a certain shift towards applied statistics with computers seems to be desirable, only the planning of a gradual cautious change is reasonable. Teaching experiments with accompanying educational research are most necessary. Even if curricula on statistics should provide a better preparation for the computer-influenced change of statistics and its applications in science and society, it would be a fallacy to simply equate the goal to be reached with the path leading to this goal. More precisely, even if most aplications of statistics use computers and are concerned with large and high-dimensional data, it should be very carefully analyzed which amount of practice on small data sets with efficient paper-and-pencil methods is necessary to develop a basic understanding of statistical techniques and problems. This may also contribute to a counterweight against blind data-distant use of computers in statistics. This argument is related to a problem of presumably more general nature. Think, e.g., of the controversial discussion of the question how much paper-and-pencil practice with symbolic manipulations is a necessary prerequisite for understanding or effectively using computer algebra systems.

Students should perhaps also experience specific limitations of prefabricated statistics software as compared to the higher flexibility and adaptability of the "paper-and-pencil-technology". This presupposes a relatively independent view of what is desirable. Let me give one illustrative example. At first sight, it is an astonishing fact that TUKEY's (1977) famous book on EDA which certainly was developed with a view towards the possibilities of the modern computing technology, contains only methods and displays which are consciously adapted to paper-and-pencil use. TUKEY's interactive style of data analysis has since become a model for requirements asked of software and its handling and it is still the case that even highly sophisticated software on large computers is partly less flexible than TUKEY's paper-and-pencil approach. HUBER (1983, 3) gives the following example: "How many packages allow you to freely compose a nice and informative picture from its components (e.g., can you paste two histograms back-to-back to facilitate comparison?" In summary, it may sound paradoxical to some ears, but I think, one influence of computers on curricula may or, better, should be, to use the paper-and-pencil-technology more consciously.

The last example also reminds us of the fact that actually there is, or should be, an interrelation between statistics and computing technology (software). To speak only of influence of computers on mathematics, as the title of the ICMI paper does, emphasizes only one direction of the relation. As the social control of technology in general, and of the computer technology in particular is a very important challenge to our societies, the educational system has to make an adequate contribution to reach this aim. Therefore, mathematics and statistics education should not primarily adapt students to the new technology, but rather a critical appreciation should be provided of what the new technology may contribute, or has already contributed, to the science of mathematics and its applications for solving relevant problems. The computer should not become the master of what is and can be done in curricula, but rather serve goals that cannot be deduced from the new technological possibilities although these developments have to be seriously taken into account when new curricula are discussed.

References

Andrews, D.F.: 1977. Comment on Stigler. In: Ann. Statist. 5, 1078 - 1079

Andrews, D.F./Bickel, P.J./Hampel, F.R./Huber, P.J./Rogers, W.H./Tukey, J.W.: 1972. Robust Estimation of Location. Princeton: Princeton University Press

Atiyah, M.: 1985. Mathematics and the computer revolution. In: ICMI/CIEM (ed.): The Influence of Computers and Informatics on Mathematics and Its Teaching. Document de travail. Strasbourg

Beniger, J.R./Robyn, D.L.: 1978. Quantitative graphics in statistics: a brief history. In: Amer. Statist. 32, 1 - 11

Benzécri, J.P.: 1980. L'Analyse des Données. Vol I: La Taxonomie; Vol. II: L'Analyse des Données. Paris: Dunod

Biehler, R.: 1982. Explorative Datenanalyse - Eine Untersuchung aus der Perspektive einer deskriptiv-empirischen Wissenschaftstheorie. Bielefeld: IDM Materialien und Studien Bd. 24

Biehler, R.: 1983. Exploratory Data Analysis - a new field of applied mahematics. In: Berry, J.S. et al. (eds.): Teaching and Applying Mathematical Modelling. Chichester: Ellis Horwood, 117 - 130

Biehler, R.: 1984. Neuere Entwicklungen in der Datenanalyse und Statistik - ihr wissenschaftstheoretisches Verständnis und Implikationen für die Fachdidaktik. In: Zum 10-jährigen Bestehen des IDM. Schriftenreihe des IDM Bd. 30, 116 - 154, or as IDM Occasional Paper No. 48. Universität Bielefeld

Biehler, R.: 1984a. Graphische Darstellungen. IDM Occasional Paper No. 50. Universität Bielefeld (to appear in: mathematica didactica 8 (1985))

Biehler, R.: 1985. Die Renaissance graphischer Methoden in der angewandten Statistik. IDM Occasional Paper No. 65. Universität Bielefeld

Chambers, J.M./Cleveland, W.S./Kleiner, B./Tukey, P.A.: 1983. Graphical Methods for Data Analysis. Duxbury: Wadsworth

Efron, B.: 1979. Computers and the theory of statistics: thinking the unthinkable. In: SIAM Review 21, 460 - 480

Engel, A.: 1985. Algorithmic aspects of stochastics. In: ICMI/CIEM (eds.): The Influence of computers and informatics on mathematics and its teaching. Document de Travail. Strasbourg

Hoaglin, D.C./Mosteller, F./Tukey, J.W. (eds.): 1983. Understanding Robust and Exploratory Data Analysis. New York: J. Wiley & Sons

ICMI (ed.): 1984. The Influence of computers and informatics on mathematics and its teaching. L'Enseignement Mathématique 30, 159 - 172

Joiner, B.L.: 1983. The case of using computers in teaching statistics. In: Proceedings of the First International Conference on Teaching Statistics, ed. by D.R. Grey et al. Sheffield: University of Sheffield Printing Unit, Vol. 1, 307 - 312

Gnanadesikan, R./Kettenring, J.R./Siegel, A.F./Tukey, P.A.: 1983. Symposium on Exploratory Data Analysis. In: Zweng, M. et al. (eds.): Proceesings of the Fourth International Congress on Mathematical Education. Boston: Birkhäuser, 344 - 357

Hogg, R.V.: 1977. Comment on Stigler. In: Ann. Statist. 5, 1088 - 1090

Huber, P.J.: 1983. Statistical Graphics: History and Overview. Paper presented at the NCGA-Conference Chicago

Mallows, C.L.: 1979. Robust methods - some examples of their use. In: Amer. Statist. 33, 179 - 184

Menges, G.: 1980. Adaptive Statistik. In: Statistische Hefte 21, 182 - 208

Pearson, E.S.: 1956. Some aspects of the geometry of statistics. In: J. Roy. Statist. Soc. Ser. A 119, 125 - 146

Råde, L.: 1983. Statistics at the school level in the age of the computer. In: Proceedings of the First International Conference on Teaching Statistics, ed. by D.R. Grey et al. Sheffield: University of Sheffield Printing Unit, Vol. 1, 19 - 33

Stigler, S.M.: 1977. Do robust estimators work with real data? In: Ann. Statist. 5, 1055 - 1078 (with discussion 1078 - 1098)

Tukey, J.W.: 1962. The future of data analysis. In: Ann. Math. Statist. 33, 1 - 67

Tukey, J.W.: 1965. The technical tools of statistics. In: Amer. Statist. 19, 23 - 28

Tukey, J.W.: 1972. Data analysis, computation and mathematics. In: Quarterly of Applied Mathematics 30, 51 - 65

Tukey, J.W.: 1977. Exploratory Data Analysis. Reading (Mass.): Addison-Wesley

Tukey, J.W.: 1979. Robust techniques for the user. In: Launer, R.L./Wilkinson, G.N. (eds.): Robustness in Statistics. New York: Academic Press, 103 - 106

Tukey, J.W.: 1983. Another look at the future. In: Heiner et. al. (eds.): Computer Science and Statistics: Proceedings of the 14th Symposium on the Interface. New York: Springer, 2 - 8

Wainer, H./Thissen, D.: 1981. Graphical data analysis. In: Annual Review of Psychology 32, 191 - 241

Werner, H.: 1982. Zur Methodologie der Numerik. In: Landesinstitut für Curriculumentwicklung, Lehrerfortbildung und Weiterbildung (Hrsg.): Numerische Mathematik in der Sekundarstufe II. Curriculum Heft 29. Neuss

Yates, F.: 1966. Computers, the second revolution in statistics. In: Biometrics 22, 233 - 251

ADDRESS OF AUTHOR:

Rolf Biehler
Institut für Didaktik der Mathematik (IDM)
Universität Bielefeld
Postfach 8640
D - 4800 Bielefeld 1

ALGORITHMIC ASPECTS OF STOCHASTICS

Arthur Engel, University of Frankfurt

I will treat in some detail one topic: stochastics, that is probability and statistics from an algorithmic standpoint. It is an extract from a book due to appear at the end of 1985 [1]. In this way I will give implicit answers to many issues raised in the ICMI report on the influence of computers and informatics on mathematics and its teaching.

Stochastics has always been computationally intensive. Due to a lack of computational power probabilists of the past mostly turned their attention to elegant results requiring little computation. By making strong and often dubious assumptions about the data statisticians could get by with a couple of small tables for Φ, t, χ^2, F.

Now the computer has revolutionized statistical practice. Theory is lagging behind, but it is catching up. In a series of examples we treat the most important statistical problems for introductory high school and college courses in the computer age. We assume that every student owns a personal computer or at least a BASIC programmable pocket calculator, but has no formal training in computer science. In our treatment statistical tables play no role. All numbers are generated by algorithms which are constructed during the course. In this way the necessary computer science topics are learned via statistics. By the way, it is also possible to develop an integrated course STATISTICS AND COMPUTER SCIENCE, but I will not go into details, since I have treated the subject elsewhere [5].

Example #1. Bold Gamble

I start with the only artificial but interesting problem from pure probability theory. It will lead to a functional equation, that will be solved by means of the computer.

Define the bold gamble:
My current fortune is $x \leq 1$, my goal is 1.
If x=1 then I quit.
If $0 < x < 1/2$ then I stake x, all I have.
If I win my fortune will become 2x, a loss ruins me.
If $1/2 \leq x < 1$ then I bet 1-x.
If I win then I have fortune x+(1-x)=1 and I quit.
If I loose I still have x-(1-x)=2x-1 left.
Suppose in one play my chance of winning is p and my chance of loosing is q. Let

f(x)=probability of eventual success under bold play starting with fortune x.
Then we have

$$f(x)=pf(2x), \qquad 0<x<1/2$$
$$f(x)=p+qf(2x-1), \qquad 1/2 \leq x<1$$
$$f(0)=0, \quad f(1)=1$$

The function f is of a strange type, called singular. It is almost everywhere differentiable with f'(x)=0, yet it is continuous and increasing. For any rational x I can actually find f(x), although sometimes with considerable effort, depending on the period of the binary expansion of x. In spite of this the following LOGO program quickly finds the result f(x) for any x and p from [0,1] . Here OP stands for OUTPUT.

```
TO F :X :P
IF :X=0 OP 0
IF :X=1 OP 1
IF :X <.5 OP :P*F 2*:X :P
OP :P+(1-:P)*F 2*:X-1 :P
END
```

Fig.1

```
FUNCTION F(X,P),
  WHEN X=0, 0 EXIT,
  WHEN X=1, 1 EXIT,
  WHEN X<1/2, P*F(2*X,P) EXIT,
  P+(1-P)*F(2*X-1,P),
ENDFUN;
```

Fig.2

p	0.4	0.3	0.25	0.1	$5\ 10^{-2}$	0.9	0.8
x	0.6	0.9	0.5	0.9	0.95	0.1	0.1
f(x)	0.465195	0.67347	0.25	0.272396	0.185693	~~0.1~~ 0.727603	0.504433

Table 1

Table 1 shows a few examples. For $p \leq 1/2$ bold gamble is an optimal strategy. For $p > 1/2$ one should play as timidly as possible.

The closely related muSIMP program in Fig.2 gives exact results for dyadic rationals i.e. for $x < 1$ and $x=n/2^p$ with nonnegative integers n,p. But for other x it never ends and eventually prints out the message ALL SPACES EXHAUSTED. The program does not end because muSIMP makes exact computations. It is unable to neglect unless told so.

Example #2. The Binomial Distribution

The formula

$$b(x) = \binom{n}{x} p^x q^{n-x}$$

for the binomial distribution is obviously useless. The recursion

$$b(0)=q^n, \quad b(x)=b(x-1)\cdot\frac{n-x+1}{x}\cdot\frac{p}{q}$$

is not completely useless, but its utility is severely limited because of underflow. For example, the Apple gives
0.5↑127=5.87747176E-39 (correct to 9 significant digits).
But we get abruptly 0.5↑128=0.
To prevent underflow we use logarithms. The following program BIN is extremely reliable and gives for input n,p,c,d the output

$$s=b(c)+b(c+1)+\ldots+b(d).$$

This suffices to solve all sensible probability problems involving the binomial distribution. In fact, the normal approximation is no more needed.

```
10 INPUT N,P,C,D:Q=1-P:R=LOG(P/Q)
20 L=N*LOG(Q): IF C=0 THEN 60
30 FOR X=1 TO C                         <---- computes ln b(c)
40   L=L+R+LOG((N-X+1)/X)
50 NEXT X
60 S=EXP(L):IF D=0 THEN 110  <---- computes b(c)
70 FOR X=C+1 TO D                       <---- computes s=b(c)+b(c+1)+..+b(d)
80   L=L+R+LOG((N-X+1)/X)
90   S=S+EXP(L)
100 NEXT X
110 PRINT S
```

Fig.3. Programm BIN

Example #3. Fisher's exact test. Hypergeometric Distribution

It is no more necessary to use artificial data. We can use real life data that are lifted from research literature, even the most extensive data for the problem at hand.

DOES VITAMIN C HELP IN PREVENTING COMMON COLD ?

For 2 months 407 persons took a Vitamin C pill per day, while 411 took a placebo pill per day. Table 2 shows the result of this Toronto Study, a double-blind randomized controlled experiment, the only type of experiment, that makes sense in medicine.

	common cold		
Vitamin C	yes	no	sum
yes	302	105	407
no	335	76	411
sum	637	181	818

Table 2. Source: Canad. Med. Ass. J., Sept. 1972, 503-508.

x	407−x	407
637−x	x−226	411
637	181	818

Table 3.

226+x	181−x	407
411−x	x	411
637	181	818

Table 4.

We want to find the observed significance level or P-value for the following hypothesis H and alternative A:

H: Vitamin C does not help. It is just a random fluctuation.

A: Vitamin C helps.

If H is true then 637 would have caught a cold anyway. By randomization only 302 turned up in the experimental group instead of the expected value

$$\frac{\text{column sum}\times\text{row sum}}{\text{grand total}}=\frac{637\cdot 407}{818}=316$$

Thus

$$P=P(x\le 302|H)=\sum_{x=226}^{302}\frac{\binom{407}{x}\binom{411}{637-x}}{\binom{818}{637}}$$

If we use Table 4 instead of Table 3, we get

$$P=P(x\le 76|H)=\sum_{x=0}^{76}h(x)=\sum_{x=0}^{76}\frac{\binom{411}{x}\binom{407}{181-x}}{\binom{818}{181}}$$

The computational effort to find this number is beyond human patience, yet a simple task even for a programmable hand calculator, which uses the program LEFT TAIL in Fig.4. The input N=818, S=181, R=411, X=76 gives the result

P=7.42356802E-03

Vitamin C definitely helps, but very little indeed.

Comments: The program uses the recurrence

$$h(x)=h(x-1)\cdot\frac{(r-x+1)(s-x+1)}{x(n-r-s+x)}$$

We used logarithms to prevent underflow. Interchange of r and s does not alter the result.

```
10 INPUT N,S,R,X
20 FOR I=1 TO S                              <---- computes ln h(0)
30   L=L+LOG((N-R-I+1)/(N-I+1))
40 NEXT I
50 P=EXP(L):IF X=0 THEN 95  <---- computes h(0)
60 FOR I=1 TO X                              <---- computes
70   L=L+LOG(((R-I+1)*(S-I+1))/(I*(N-R-S+I)))      p=h(0)+h(1)+...+h(x)
80   P=P+EXP(L)
90 NEXT I
95 PRINT P
```

Fig.4. Program LEFT TAIL

Example #4. Wilcoxon's Two-Sample-Test

Table 5 shows the frequency of use of one letter words per 1000 words in 8 essays of A. Hamilton and 7 essays of J. Madison.

Hamilton	24	21	23	24	33	28	28	37
Madison	20	27	19	30	11	17	27	

Table 5. Source: F.Mosteller and D.L. Wallace. Inference and Disputed Authorship: The Federalist,1964,p.248.

Could these be random samples from the same distribution ?

Let us sort both sets of data increasingly and then denote Madison's data by 0 and Hamilton's data by 1. Then we get the binary word

W=000011110011011

Below each 0 we write the number of ones to the left of it. We get

U=6+4+4=14

For the reflected word

W'=110110011110000

we get

U'=8+8+8+8+4+4+2=42

Later we will use the number U of inversions to derive a simple recursion . But for simulation purposes we use the equivalent rank sum RS of the zeros:

RS(W)=1+2+3+3+9+10+13=42

For the reflected word W' we get for the rank sum of zeros:

RS(W')=3+6+7+12+13+14+15=70

For symmetry reasons we have

$P(U \leq 14) = P(U \geq 42)$

and

$P(RS \leq 42) = P(RS \geq 70)$

In the first months of an introductory statistics course we give for each problem several solutions:

a) straightforward simulation

b) exact solution

c) sophisticated simulation giving ultimate power.

We always present a) and b) but only sometimes c).

a) Let us find the probability $P(RS \leq 42)=P(RS \geq 70)$ by simulation.

```
10 INPUT M,N,A,B:S=M+N:DIM X(S):DEF FNR(X)=1+INT(S*RND(1))
20 FOR K=1 TO 1000:RS=0
30   FOR I= 1 TO S:X(I)=0:NEXT
40   FOR I=1 TO M
50     R=FNR(1):IF X(R)=1 THEN 50
60     RS=RS+R:X(R)=1
70   NEXT I
80   IF RS <=A OR RS>=B THEN T=T+1
90 NEXT K
95 PRINT T
```

Fig.5

Fig.6 shows a stem and leaf plot of 20 simulation runs of this program with input m=7, n=8, a=42, b=70.

```
 9|7
10|12
10|89
11|0341
11|7
12|01
12|897
13|3
13|8868
```

Fig.6

We get for P an estimate

$P=P(RS \leq 42) \approx 5.9675\%$

b) There is a one-to-one correspondence between words with m zeros and n ones and partitions into at most m parts with each part at most n. For instance, with m=7 and n=8 the binary word

100001010111011

determines uniquely the partition

U=6+3+2+1+1+1+1

and vice versa. The possible cases are easy to enumerate. They are all

$\binom{15}{7} = 6435$

binary words with 7 zeros and 8 ones. The favorable cases for our event $U \leq 14$ are those partitions of the numbers0 to 14 into at most 7 parts with each part at most 8.

We will solve the general problem. Let w(u,m,n) be the number of partitions of 0,1,...,u into at most m parts with each part at most n. It is easy to derive a recursion for w. A partition either contains n or it does not contain n. If it contains n we must partition u-n into at most m-1 parts with each part at most n. If it does not contain n, we must partition u into at most m parts, with each part at most n-1. That is

w(u,m,n) = w(u-n,m-1,n) + w(u,m,n-1)

with the boundary conditions

w(0,m,n)=w(u,0,n)=w(u,m,0)=1, w(u,m,n)=0 for $u<0$.

We have the theorem

w(u,m,n)=w(u,n,m)

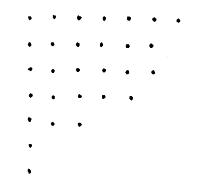


Fig.7

The proof is immediate by reading the partition in Fig.7 first rowwise and then columnwise.

LOGO is a computer language of miraculous simplicity and power. It understands the recursion for w, if you type in the program in Fig.8.

```
TO  W :U :M :N
IF :U< 0 OP 0
IF :U=0 OP 1
IF :M=0 OP 1
IF :N=0 OP 1
OP (W :U :M :N-1) + (W :U-:N :M-1 :N)
END
```

Fig.8

If you now type in

W 14 7 8

then after 36 seconds of computation you will get the result 388. That is, w(14,7,8)=388. The computatin of the same number w(14,8,7) takes somewhat less time.

So we get finally

$P = P(U \leq 14) = \frac{388}{6435} = 6.0\%$

Simulation gave us $P \approx 5.9675\%$, a remarkably good approximation. For large u,m,n the BASIC program in Fig.9 is very much faster then the LOGO program, but its construction requires more effort.

```
10 INPUT U,M,N:DIM W(U,M,N)
20 FOR Z=0 TO U
30   FOR Y=0 TO N
40     W(Z,0,Y)=1
50   NEXT Y
60 NEXT Z
70 FOR Z=0 TO U
80   FOR X=0 TO M
90     W(Z,X,0)=1
100  NEXT X
110 NEXT Z
120 FOR X=0 TO M
130   FOR Y=0 TO N
140     W(0,X,Y)=1
150   NEXT Y
160 NEXT X
170 FOR X=1 TO M
180   FOR Y=1 TO N
190     FOR Z=1 TO U
200       IF Z<Y THEN W(Z,X,Y)=W(Z,X,Y-1):GOTO 220
210       W(Z,X,Y)=W(Z,X,Y-1)+W(Z-Y,X-1,Y)
220     NEXT Z
230   NEXT Y
240 NEXT X
250 PRINT W(U,M,N)
```

Fig.9

c) The Bootstrap Method: the ultimate method in nonparametric statistics

The mean number of one letter words per 1000 words is for Hamilton

A=27.25

and for Madison

$B=21\frac{4}{7}$

We have

A-B=5.67857143.

H: The two samples come from the same distribution.

A: Hamilton uses on the average more one letter words.

The Bootstrap Method uses the idea that every sample carries its own internal yardstick of variability that can be extracted by drawing (for example) 1000 artificial samples from the given sample of 15 data. The program in Fig.10, which carries out this idea, is of utmost simplicity. For input m=7 and n=8 it draws from the 15 data a sample of 7 and a sample of 8, each time with replacement, and finds their averages A and B. This is repeated 1000 times and a counter T counts how often $|A-B| \geq 5.67857143$.

```
10 INPUT M,N:S=M+N:DIM X(S):DEF FNR(X)=1+INT(S*RND(1))
20 FOR I=1 TO S:READ X(I):NEXT
30 FOR J=1 TO 1000:A=0:B=0
40   FOR I=1 TO M:R=FNR(1):A=A+X(R):NEXT
50   FOR I=1 TO N:R=FNR(1):B=B+X(R):NEXT
60   A=A/M:B=B/N:IFABS(A-B)>=5.67857143 THEN T=T+1
70 NEXT J
80 PRINT T
90 DATA 24,20,21,27,23,19,24,30,33,11,28,17,28,27,37
```

Fig.10

```
6|49
7|11888
8|01277
9|267
```

Fig.11

The stem and leaf plot in Fig.11 shows 15 runs of this program. It yields

$P = P(A-B \geq 5.67857143) \approx 4.0\%$

Example 5. Matched Pairs

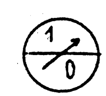
a) The data in Table 6 are from the first controlled marijuana study. They show for 9 subjects the changes X and Y in mental performance 15 minutes after smoking an ordinary cigarette and a marijuana cigarette, respectively. The data are scores from each subject's base level. We have also tabulated the difference D=Y-X and $|D|=d_k$.

Subject No.	1	2	3	4	5	6	7	8	9	
Ordinary cigarette X	-1	-1	-3	3	-3	-3	2	4	10	$\sum X = 8$
Marijuana cigarette Y	1	-3	-7	-3	-9	5	-6	-7	-17	$\sum Y = -46$
Difference D=Y-X	2	-2	-4	-6	-6	8	-8	-11	-27	$\sum D = -54$
Absolute value $\|D\|=d_k$	2	2	4	6	6	8	8	11	27	

Table 6. Source: Science 162, 1234-1242

Before computers were available the differences d_k were replaced by their ranks in order to use precomputed tables. This habit still persists. But actually the computer makes tables obsolete. We will work more exactly with the differences instead of their ranks. This is in fact somewhat easier since any combination of ties is allowed.

As test statistic we use the random variable



$T = d_1 I_1 + d_2 I_2 + \ldots + d_9 I_9$,

where the I_k are tosses of a fair coin with outcomes 0 and 1 (Fig. 12).

In our example the sum of the positive differences is T=2+8=10. We want to find $P(T \leq 10)$. The possible and equiprobable cases are all 2^9 subsets of the last row in Table 6. The favorable cases are those subsets with sum 10 or less. These can be found in a few minutes:

8,8,8+2,8+2,8+2,8+2,6,6,6+4,6+4,6+2,6+2,6+2,6+2,6+2+2,6+2+2,4,4+2,4+2,4+2+2,2,2,2+2,0.

Thus

$P(T \leq 10) = \frac{24}{512} = \frac{3}{64} = 4.6875\%$

b) Does maternal malnutrition retard the mental development of a child? In particular, does the better nourished (in utero) of two identical twins usually develop a higher IQ than the other?

The IQ of 14 pairs of identical twins was measured years after birth and compared with the weight at birth. Table 7 shows the result for the 12 pairs with different IQ's. In this table X and Y are the IQ's of the heavier twin and lighter twin, respectively.

X	100	124	108	91	100	91	79	80	95	104	100	119
Y	101	123	106	97	106	84	70	70	84	92	85	104
$\lvert Y-X\rvert = d_k$	1	1	2	6	6	7	9	10	11	12	15	15

Table 7. Source: Churchill and Willerman: Intelligence and birth weight in identical twins. Child Development, vol.38, No.3, 623-629.

The sum of the positive differences Y-X is here T=1+6+6=13. As test statistic we use again

$T = d_1I_1+d_2I_2+\ldots+d_{12}I_{12}$.

To find P(T 13) we need all subsets of the d_k's with sum 13 or less. These are

12,12+1,12+1,11,11+1,11+1,11+2,11+1+1,10,10+2,10+1,10+1,10+2+1,10+2+1,10+1+1,9,9+2, 9+1,9+1,9+2+1,9+2+1,9+1+1,9+2+1+1,7,7+6,7+6,7+2,7+1,7+1,7+2+1,7+2+1,7+1+1,7+2+1+1, 6,6,6+6,6+2,6+2,6+1,6+1,6+1,6+1,6+6+1,6+6+1,6+2+1,6+2+1,6+2+1,6+2+1,6+1+1,6+1+1, 6+2+1+1,6+2+1+1,2,2+1,2+1,2+1+1,1,1,1+1,0 .

There are 60 favorable cases and 2^{12} or 4096 possible cases. Thus

$P(T \leq 13) = \frac{60}{4096} = \frac{15}{1024} = 1.465\%$

This is strong evidence for the hypothesis that the better nourished twin develops a higher IQ. But we had to pay a stiff computational price for the answer. We have done this to get familiar with the computational aspects of the problem before writing a computer program.

c) Should hospitals allow parents of sick children to live in the clinic together with the children (rooming-in model) in order to reduce time spent in the hospital?

50 children were matched in relevant variables to form 25 pairs. From exactly one child of each pair the parents were allowed to live in the hospital. Table 8 shows the time spent in the hospital in days, X without rooming-in and Y with rooming-in. The six pairs with Y-X=0 are not listed.

X	15	7	18	11	11	11	12	22	14	15	15	29	23	8	18	16	44	28	31
Y	14	6	16	8	8	8	7	17	19	8	8	21	15	17	8	6	32	12	8
$\lvert Y-X\rvert = d_k$	1	1	2	3	3	3	5	5	5	7	7	8	8	9	10	10	12	16	23

Table 8. Source: Fernandez-Jung: Auswirkung elterlicher Mitaufnahme (rooming-in Modell) auf das Verhalten stationär behandelter Kinder. Diss. FU Berlin 1983

Our test statistic is again

$T = d_1I_1+d_2I_2+\ldots+d_nI_n$ with n=19.

The sum of the positive differences Y-X ist T=5+9=14. Thus we are interested in the number of favorable cases for the event $T \leq 14$. More generally, let

$q(t,n)$ = # of favorable cases for the event $T \leq t$ = # of representations of $0,1,2,\ldots,t$ in the form $T=d_1I_1+\ldots+d_nI_n$.

We have immediately

$q(t,n) = q(t,n-1) + q(t-d_n,n-1)$

with the boundary conditions

$q(t,n)=0$ for $t<0$, $q(0,n)=q(t,0)=1$.

Indeed, there are $q(t,n-1)$ such representations without the maximal element d_n and $q(t-d_n,n-1)$ with element d_n.

Fig.13 shows an elegant LOGO program. Instead of the counting variable n it uses the list :D of the absolute values of the differences.

```
TO Q :T :D
IF :T<0 OP 0
IF :T=0 OP 1
IF :D=[]OP 1
OP (Q :T BL :D)+(Q :T - LAST :D BL :D)
END
```

Fig.13

The input

Q 14 [1 1 2 3 3 3 5 5 5 7 7 8 8 9 10 10 12]

yields after 1 minute the result 537. Thus we get

$P(T \leq 14) = \frac{537}{2^{19}} = 1.02425\cdot 10^{-3} \approx 10^{-3}$

Since the computation time is roughly proportional to the number of favorable cases the LOGO program cannot be used for considerably larger data.

The BASIC program in Fig.14 can be understood without comment. It is fast, but amazingly enough it causes memory problems on a microcomputer since all nt numbers q(t,n) are stored. The program in Fig.15 is almost as fast and never causes any memory problems, except over and/or underflow. It uses only onedimensional arrays. P(T) and S(T) stand

stand for preceding and succeeding row. It computes the numbers q(t,n) rowwise, but keeps only the preceding row.

```
10 INPUT T,N:DIM D(N),Q(T,N)
20 FOR I=1 TO N: READ D(I): NEXT
30 FOR I=0 TO N: Q(0,I)=1:NEXT
40 FOR I=0 TO T: Q(I,0)=1:NEXT
50 FOR J=1 TO T
60   FOR I=1 TO N
70     IF J<D(I) THEN Q(J,I)=Q(J,I-1)
75     ELSE Q(J,I)=Q(J,I-1)+Q(J-D(I),I-1)
80   NEXT I
85 NEXT J
90 PRINT Q(T,N)/2 N
95 DATA 1,1,2,3,3,3,5,5,5,7,7,8,8,9,10,10,12,16,23
```

Fig.14. Program MATCHED PAIRS

```
10 INPUT T,N:DIM D(N),P(T),S(T)
20 FOR I=1 TO N:READ D(I):NEXT
30 FOR I=0 TO T:P(I)=1:NEXT
40 FOR I=1 TO N
50   FOR J=0 TO T
60     IF J<D(I) THEN S(J)=P(J)
70     ELSE S(J)=P(J)+P(J-D(I))
75   NEXT J
80   FOR K=0 TO T:P(K)=S(K):NEXT
85 NEXT I
90 PRINT S(T)
95 DATA 1,1,2,3,3,3,5,5,5,7,7,8,8,9,10,10,12,16,23
```

Fig.15. program MATCHED.PAIRS

d) Let us now go back to the marijuana example. By plotting Y versus X we observe that there seems to be one "outlier" present, the point (10,-17). With so few data at hand we cannot afford to throw away a single point. So we decide to keep it. In addition we observe that Y does not seem to depend on X. That is, instead of 9 pairs we have 18 independent data -3,5,10,-17,-3,-7,3,-3,4,-7,-3,-9,2,-6,-1,1,-1,-3 . To these data we apply the Bootstrap Method. That is, we draw from this sample at random with replacement 9 numbers X and 9 numbers Y, and we find their sums S and T. This is repeated 1000 times and we count with the variable C, how often $|S-T| \geq 54$, as in TABLE 1. The BASIC program can be found in Fig.15a. 32 runs of this program yielded the C-values

```
2|2
2|66788999
3|000112234
3|555667899
4|01123
```

The median of these data is $\tilde{C} = (32+33)/2 = 32.5$. Thus

$P(|T-S| \geq 54) \approx 3.25\%$

$P(T-S \leq -54) \approx 1.6\%$

This is considerably better than the 4.7% we got earlier.

```
10 READ N,D:DIM X(N):DEF FNR(X)=1+INT(N*RND(X))
20 FOR I=1 TO N:READ X(I):NEXT
30 FOR J=1 TO 1000:S=0:T=0
40   FOR I=1 TO N/2
50     R=FNR(1):S=S+X(R):R=FNR(1):T=T+X(R)
60   NEXT I
70   IF ABS(S-T)>=D THEN C=C+1
80 NEXT J
90 PRINT C
95 DATA 18,54,-3,5,10,-17,-3,-7,3,-3,4,-7,-3,-9,2,-6,-1,1,-1,-3
```

Fig.15.a

e) Permutation Test

By means of the computer we can traverse all $\binom{18}{9} = 48\,620$ 9-subsets of our 18 data, and we find that exactly 766 subsets have sum -46 or less. Hence the "exact significance level" is

$$P = \frac{766}{48620} = 1.5755\%$$

The corresponding program has exponential complexity. If we traverse the subsets in lexicographic order then it will take on an apple II about 1.5 hours. This is not very efficient, but it has the advantage of a simpler program.

Example #6. Kendall's Rank Correlation

Fig.16 shows the US draft lottery for 1970. We want a quick test to decide, if it is a random distribution of the numbers 1 to 366 over the days of the year. For this purpose we find the median for each month in Fig.15 and get the result in Table 9.

Table 1. 1970 Random selection sequence, by month and day

	Jan.	Feb.	Mar.	Apr.	May	June	July	Aug.	Sep.	Oct.	Nov.	Dec.
1	305	086	108	032	330	249	093	111	225	359	019	129
2	159	144	029	271	298	228	350	045	161	125	034	328
3	251	297	267	083	040	301	115	261	049	244	348	157
4	215	210	275	081	276	020	279	145	232	202	266	165
5	101	214	293	269	364	028	188	054	082	024	310	056
6	224	347	139	253	155	110	327	114	006	087	076	010
7	306	091	122	147	035	085	050	168	008	234	051	012
8	199	181	213	312	321	366	013	048	184	283	097	105
9	194	338	317	219	197	335	277	106	263	342	080	043
10	325	216	323	218	065	206	284	021	071	220	282	041
11	329	150	136	014	037	134	248	324	158	237	046	039
12	221	068	300	346	133	272	015	142	242	072	066	314
13	318	152	259	124	295	069	042	307	175	138	126	163
14	238	004	354	231	178	356	331	198	001	294	127	026
15	017	089	169	273	130	180	322	102	113	171	131	320
16	121	212	166	148	055	274	120	044	207	254	107	096
17	235	189	033	260	112	073	098	154	255	288	143	304
18	140	292	332	090	278	341	190	141	246	005	146	128
19	058	025	200	336	075	104	227	311	177	241	203	240
20	280	302	239	345	183	360	187	344	063	192	185	135
21	186	363	334	062	250	060	027	291	204	243	156	070
22	337	290	265	316	326	247	153	339	160	117	009	053
23	118	057	256	252	319	109	172	116	119	201	182	162
24	059	236	258	002	031	358	023	036	195	196	230	095
25	052	179	343	351	361	137	067	286	149	176	132	084
26	092	365	170	340	357	022	303	245	018	007	309	173
27	355	205	268	074	296	064	289	352	233	264	047	078
28	077	299	223	262	308	222	088	167	257	094	281	123
29	349	285	362	191	226	353	270	061	151	229	099	016
30	164		217	208	103	209	287	333	315	038	174	003
31	211		030		313		193	011		079		100

Source: Selective Service System, Office of the Director, Washington, D.C.

Fig.16

month	1	2	3	4	5	6	7	8	9	10	11	12
median of the ranks	211	210	256	225	226	207.5	188	145	168	201	131.5	100

Table 9

If we replace each median by its rank we get the permutation

9 8 12 10 11 7 5 3 4 6 2 1

Is this a "random" permutation ? There are 66 pairs altogether, of which 11 are rising (concordant pairs) and 55 are falling (discordant pairs). In a random permutation we expect 33 rising and just as many falling pairs. Discordant pairs are called inversions.

a) Let us generate random 12-permutations, count the inversion with the variable INV, and test if INV$\leq$11 or INV$\geq$55.

```
 10 INPUT N,L,H:DIM X(N)
 20 FOR M=1 TO 1000:INV=0
 30 FOR I=1 TO N:X(I)=I:NEXT
 40 FOR I=N TO 2 STEP -1
 50   K=1+INT(I*RND(1)):C=X(I):X(I)=X(K):X(K)=C
 60 NEXT I
 70 FOR I=1 TO N-1
 80   FOR J=I+1 TO N
 90     IF X(I)>X(J) THEN INV=INV+1
100   NEXT J
110 NEXT I
120 IF INV<=L OR INV>=H THEN T=T+1
130 NEXT M
140 PRINT T
```

Fig.17

The T-values of 10 simulation runs are 0, 1, 2, 2, 1, 2, 1, 2, 3, 2. This gives for $P(INV \leq 11) = P(INV \geq 55)$ the estimate

$P \approx 0.0008$

b) To find the exact value of this probability we must find the number of 12-permutations with at most 11 inversions and divide this number by 12!.

Let <u>p(n,k)</u> be the <u>number of n-permutations with at most k inversions</u>. Our aim is to derive a computer friendly recursion formula for p(n,k).

Suppose I have an (n-1)-permutation $X_1X_2\ldots X_{n-1}$ of $\{1,2,\ldots,n-1\}$. I can make of it an n-permutation by inserting element n into one of n places numbered 0 to n-1, as indicated by squares below

$\boxed{n-1}X_1\boxed{n-2}X_2\ \ldots\boxed{i}X_{n-i}\ldots\boxed{2}X_{n-2}\ \boxed{1}X_{n-1}\boxed{0}$

If I place element n into place i, it contributes i inversions. To get at most k inversions altogether the (n-1)-permutation $X_1X_2\ldots X_{n-1}$ must contribute at most k-i inversions. There are p(n-1,k-i) (n-1)-permutations with this property. By inserting element n successively into places 0 to n-1 we get

$$p(n,k) = \sum_{i=0}^{n-1} p(n-1,k-i)$$

This recursion is not yet computer friendly. But let us replace k by k-1:

$$p(n,k-1) = \sum_{i=0}^{n-1} p(n-1,k-1-i)$$

Subtracting we get

$$p(n,k) = p(n,k-1) + p(n-1,k) - p(n-1,k-n)$$

If we take into account the boundary conditions

$p(n,0)=p(1,k)=1$ for $n\geq 1$, $k\geq 0$ and $p(n,j)=0$ for $j<0$

then we get

$k<n \Longrightarrow p(n,k)=p(n,k-1)+p(n-1,k)$

$k\geq n \Longrightarrow p(n,k)=p(n,k-1)+p(n-1,k)-p(n-1,k-n)$

$p(n,0)=p(1,k)=1$

Let us translate this recursion into LOGO:

```
TO P :N :K
IF :K=0 OP 1
IF :N=1 OP 1
IF :K<:N OP (P :N-1 :K)+(P :N :K-1)
OP (P :N :K-1)+(P :N-1 :K)-(P :N-1 :K-:N)
END
```

Fig.18

This program is quite slow. It does find p(9,6) in 3.5 minutes, but p(12,11)=431886 is too much for it. This value was found by the fast BASIC program in Fig.19. Thus we get the exact value

$P(INV\leq 11) = P(INV\geq 55) = 431886/12! = 0.00090$

```
10 INPUT N,K:DIM P(N,K)
20 FOR I=1 TO N:P(I,0)=1:NEXT
30 FOR I=0 TO K:P(1,I)=1:NEXT
40 FOR I=2 TO N
50   FOR J=1 TO K
60     IF J<I THEN P(I,J)=P(I,J-1)+P(I-1,J):GOTO 80
70     P(I,J)=P(I,J-1)+P(I-1,J)-P(I-1,J-I)
80   NEXT J
90 NEXT I
95 PRINT P(N,K)
```

Fig.19

Example #7. Random Sampling: A Paradigmatic Example

You have a set $\{1,2,\ldots,n\}$. Choose a random subset of size s.
This simple and important example is an excellent topic to learn enough probability and computer science to carry you very far.

The program in Fig.20 comes immediately to your mind. A number R between 1 and N is chosen at random. Then one tests with X(R)=1 if it has already occurred. If so the the random choice is repeated. If not R is printed and we set X(R)=1 to remind us that R has already occurred.

```
10 INPUT N,S:DIM X(N):DEF FNR(X)=1+INT(N*RND(X))
20 FOR I=1 TO S
30   R=FNR(1):IF X(R)=1 THEN 30
40   PRINT R:X(R)=1
50 NEXT I
```

Fig.20

This method has the disadvantage that it requires space N and the sample is not sorted. How much time does it take on the average ? This is the classic coupon collector's problem. In Feller's Volume 1 it is already treated in the context of random sampling. The answer is well known to be

$$E(T) = N\left(\frac{1}{N} + \frac{1}{N-1} + \cdots + \frac{1}{N-S+1}\right) \approx N \cdot \ln \frac{N}{N-S+1}$$

For $S \leq N/2$ it is quite efficient. For larger S one chooses the numbers to be rejected.

```
10 INPUT N1,S1:N=N1:S=S1
20 FOR I=1 TO N1
30   IF N*RND(1)<S THEN S=S-1:PRINT I
40   N=N-1
50 NEXT I
```

Fig.21

The program in Fig.21 has the disadvantage that it requires time N and it is difficult to understand. It has the advantage that it requires constant space and gives the sample in sorted order.

Why does the second algorithm work ?
There is a nice, short but subtle argument, not at all trivial. Show by means of a tree that the algorithm works for S=2 and S=3.
In future there must be more probability problems suggested by algorithms. Writers of textbooks on stochastics have thus far ignored this inexhaustible source of new and interesting problems. Fortunately D. E. Knuth's "Computer Science Bible" is a good source of such problems, at least for the teacher.

Now take the following problem:
From the phone book of New York with N=2000000 entries you are to select at random S=2000 entries for a phone interview. At your disposal is a microcomputer.
With the second algorithm you need time N.
With the first algorithm you need space N. You can save space by storing the selected entries in an array. To prevent repetitions you will have to make (S-1)(S-2)/2 comparisons which is again about 2000000. Thus we simply traded in space N for time N.

Introduce hashing with complexity (Time,Space)=(CS,CS), a vast improvement, but the output is not sorted. Hashing is a useful technique every mathematics or computer science student should be familiar with. See [2] for details.

Another approach would be to store the selected elements in a binary tree. The complexity now becomes (CS·ln S,CS). Binary trees are also a fundamental data structure that should be familiar in the future to anyone, who wants to use the computer in a nontrivial way. See [3] for details.

Let X(S,N) be the random variable that counts the number of records to skip over before selecting the next record for the sample. Here the parameter S is the number of records remaining to be selected and N is the total number of records left in the file. J.S. Vitter [4] uses X to develop three new algorithms A,C,D. The last one has time complexity CS and constant space, and it gives in addition a sorted output. The paper [4] contains a lot of elementary nontrivial probability.

References:

1. Arthur Engel, Stochastik, Band 1, Klett Studienbücher Mathematik 1985.

2. H. Wilf, Combinatorial Algorithms, 1st edition 1975, Academic Press.

3. H. Wilf A.M.M., January 1979,30-36 and correction on page 571 (1979).

4. J.S. Vitter, Faster Methods for Random Sampling. CACM, July 1984, 703-718.

5. Arthur Engel, Statistics and Computer Science: An Integrated High School Course. Due to appear 1985 in Lennart Rade,ed., Proceedings of the ISI Round Table Conference on the Impact of Calculators and Computers on Teaching Statistics in Canberra in August 1984.

Arthur Engel
University of Frankfurt
Department of Mathematics
Senckenberganlage 9
6000 Frankfurt
West Germany

LES CALCULATEURS ET L'ENSEIGNEMENT DES PROBABILITES

J.P. CONZE
Université de RENNES

INTRODUCTION

Théorique et appliqué, intuitif et formaliste, discret et continu, le calcul des probabilités offre un champ idéal d'expérimentation des moyens micro-informatiques.

Dans cette note, nous rassemblons quelques remarques sur les modifications de l'enseignement des probabilités qui pourraient résulter à la fois de l'expérimentation sur machine, et de l'introduction de l'informatique théorique dans les cursus de mathématiques.

1 - PROBABILITES ET CALCUL

Le mariage de la théorie des probabilités et du calcul ne date pas de la micro-informatique.
Dès 1971, le livre de FREIBERGER et GRENANDER propose un cours de probabilités calculatoires (*), auquel le développement de la micro-informatique redonne aujourd'hui toute son actualité.
En fait, il faut remonter au formidable développement de la méthode de Monte-Carlo, après guerre, et à la création d'une nouvelle discipline, la simulation, qui fournit les éléments d'une théorie des probabilités expérimentée à la machine.
On trouve dans les ouvrages consacrés à la simulation un matériel pédagogique riche : visualisation de théorèmes limites par tirages d'échantillons, écritures d'algorithmes, application de tests et possibilité d'en discuter la validité (ce qui est impossible avec des données concrètes) etc... De cette façon, la simulation n'est plus seulement un moyen d'appliquer les probabilités, mais aussi un moyen d'expérimenter la théorie.
Bien entendu la statistique est largement présente dans cette expérimentation, dans tous ses aspects d'estimation, de tests d'hypothèses qu'il ne nous paraît pas possible de séparer d'un cours de probabilités.

(*) : A short course in computational Probability and Statistics (Springer, Appl. Math. Sc. V6).

S'il est ainsi relativement facile de concevoir des travaux pratiques de probabilités sur machine (*), cela ne résoud pas la question du contenu du cours théorique. Nous allons voir qu'en effet ce contenu risque d'être influencé par l'expérimentation.

2 - L'ENSEIGNEMENT TRADITIONNEL DES PROBABILITES

Cet enseignement se présente essentiellement comme une extension de la théorie de l'intégration et de la mesure. Les outils de base, convolution, transformation de Fourier, ensembles mesurables sont ceux de l'analyse. Ce qui est propre au calcul de probabilités, c'est le vocabulaire (indépendance, variable aléatoire, évènement...), introduit pour sous-tendre des notions intuitives issues de l'observation de phénomènes aléatoires, mais réduits par la modélisation à des objets mathématiques standard : mesure produit, fonction ou ensemble mesurable,...
C'est l'axiomatisation proposée par KOLMOGOROV et adoptée à partir des années 30; La plupart des cours de probabilités commencent donc par un long préliminaire sur la théorie de la mesure.
Si elle est justifiée par la cohérence de la construction mathématique, cette démarche n'en présente pas moins de sérieux inconvénients pédagogiques : aridité et difficulté intrinsèque de la théorie de la mesure, et surtout "escamotage" de concepts, puisque, partant d'objets d'expériences qui sont des échantillons numériques doués de propriétés statistiques, on aboutit à la notion de variable aléatoire "statique", vue comme une fonction définie sur un espace probabilisé idéal.
Comme on le sait, ces deux points de vue se réconcilient dans l'énoncé de la loi des grands nombres, qui justifie, à posteriori, la modélisation utilisée, en montrant que la fréquence d'une suite d'évènements indépendants de même probabilité est presque sûrement égale à cette probabilité.

(*) Sa mise en oeuvre est moins aisée, et grandement facilitée par la disponibilité de logiciels adaptés.

L'expérimentation sur machine risque cependant d'avoir des effets importants sur le contenu du cours :

Déplacement des points de vue

Les objets représentés sur l'écran sont les échantillons numériques obtenus par simulation, les tracés des lois de probabilité empiriques ou théoriques, etc., mais jamais, bien évidemment l'espace $(\Omega, \mathcal{A}, P)$ des probabilistes, et les variables aléatoires considérées.
Notons que la théorie de la mesure est confrontée à la même difficulté de représentation concrète de certains concepts.
Tout naturellement, la notion de fréquence, et donc de mesure de Radon, redevient prépondérante au détriment des notions d'intégration abstraites.

Importance critique du "presque-sûr"

La "pathologie" cachée par la théorie de la mesure dans le presque-partout ou le presque-sûr ne peut plus être masquée ici, puisque l'expérimentateur parcourt, une fois pour toutes, la même trajectoire (il a choisi un point ω de l'espace Ω) avec les risques que cela comporte en cas de "mauvais choix".
Une analyse plus approfondie de la notion même de probabilité devient nécessaire.

Rôle des méthodes de simulation

La méthode utilisée pour le générateur aléatoire doit être expliquée aux étudiants. Il n'est pas possible en effet, sauf au niveau élémentaire, de la considérer comme une "boîte noire". Il est alors nécessaire de disposer dans le cours des outils permettant de justifier la méthode, au moins dans son principe.

3 - DES SUGGESTIONS POUR LE CONTENU DU COURS

Les quelques remarques précédentes nous conduisent à une série de suggestions :

Place de la statistique

Comme nous l'avons vu plus haut, une part importante de la statistique a sa place dans un cours de probabilités calculatoires.

De plus, au niveau élémentaire, une initiation à la statistique permet de bien dégager les concepts de fréquence, de loi limite qui seront ensuite repris dans un cours plus avancé de probabilités.
Les points que nous allons aborder maintenant concernent les cours de probabilités avancés.

Un vieux débat

Les problèmes posés par l'expérimentation à la théorie raniment le débat ouvert par VON MISES en 1919, et nous ramènent aux travaux qui se sont poursuivis jusqu'à ces dernières années (WALD, VILLE, CHURCH, KOLMOGOROV, MARTIN-LÖFF...), pour donner un statut mathématique correct à la notion de suite de nombres au hasard, c'est-à-dire à l'objet fourni par la simulation et observé à l'écran.

La formalisation de la notion fait un large emprunt à la théorie des fonctions récursives, à la théorie de la complexité. Elle a sa place, au niveau des cours avancés, d'une part parce qu'elle résoud certaines difficultés mentionnées plus haut, d'autre part, parce qu'elle permet aux étudiants d'appliquer les concepts qu'ils ont rencontrés dans les cours d'informatique théorique (*), quand ces cours sont intégrés au cursus de mathématiques.

(*) il faut mentionner les pages consacrées par KNUTH à ce sujet dans son ouvrage "The art of computer programming".

Stationnarité et systèmes dynamiques

La place importante de la notion de fréquence devrait conduire à développer plus que par le passé la notion de processus stationnaire, et à aborder par là, avec la théorie ergodique, l'étude des systèmes dynamiques.
Ce point de vue permet de traiter plus à fond le problème du presque-partout ou du presque-sûr, en conduisant à la notion de point générique pour un système dynamique, au sens d'une mesure invariante.
La description des systèmes dilatants et de leurs nombreuses mesures invariantes est nécessaire pour expliquer l'apparition, dans divers domaines, de comportements aléatoires, et la signification du presque-partout.
De façon générale, la présentation du comportement "chaotique" des itérées de certaines transformations a sa place dans l'enseignement, d'autant que les tracés graphiques sont là particulièrement éclairants.

Présentation des générateurs de nombres au hasard

L'introduction des systèmes dynamiques dilatants permet de faire comprendre le sens de la méthode des congruences linéaires, utilisée pour engendrer des suites aléatoires.
Malheureusement une analyse plus poussée des différentes méthodes proposées pour fabriquer des générateurs aléatoires est difficile, et reste un sujet largement ouvert à la recherche.

Place de la théorie de la mesure

Il n'est, bien entendu, pas question de renoncer aux développements classiques des cours de Probabilités (en se préoccupant peut-être plus des questions de vitesse de convergence).
Mais il est clair que l'adjonction des différentes théories mentionnées plus haut ne peut se faire qu'en allégeant d'autres domaines : ceci pourrait conduire à réduire le temps consacré à la théorie de la mesure, et, dans l'ensemble du cursus, à la construction de l'intégrale de Lebesgue.

CONCLUSION

L'usage des ordinateurs dans l'enseignement des mathématiques, en permettant une expérimentation des mathématiques , déplace les centres d'intérêts, pose des problèmes mathématiques nouveaux et difficiles, oblige à une réflexion vivifiante sur des thèmes considérés comme traditionnels.
L'enseignement des probabilités, théorie au confluent d'idées et de préoccupations très diverses, a tout à gagner à cette confrontation avec la machine.

UNE THEORIE DU CONTINU ET QUELQUES APPLICATIONS

par

Jacques HARTHONG

Depuis Cantor, Dedekind, et Hilbert, la conviction qu'aucune théorie mathématique du continu n'est possible sans recourir aux ensembles infinis est partagée par la quasi-totalité des mathématiciens.(*) Cette conviction est devenue un dogme d'autant plus prégnant que l'enseignement contemporain de la mathématique met le plus grand soin à évacuer systématiquement la seule connaissance qui eût permis de le relativiser : celle de l'histoire de la mathématique, qui répond à la question comment et pourquoi en est-on arrivé là ?

Il faut donc rappeler certains faits bien oubliés. Lorsque Hilbert a inventé la mathématique formelle, c'était pour pouvoir conserver la théorie cantorienne des ensembles infinis, qui lui paraissait indispensable pour fonder la géométrie et le calcul infinitésimal (dont il ne peut être question de priver la mathématique) et qui sous sa forme primitive était inconsistante (voyez les célèbres paradoxes). L'idée de Hilbert, qui constitue le principe fondamental de la mathématique formelle, fut de nier tout caractère objectal aux ensembles ; seul un système de propriétés détachées de toute intuition resterait objectif, à condition d'être dépourvu de contradiction interne [4] . Oublier cela et croire naïvement que les termes de la théorie formelle des ensembles infinis <u>sont</u> les objets que notre intuition tente de concevoir (comme le continu) est non seulement une erreur que Hilbert n'a pas commise, mais mène tout droit au cercle vicieux dénoncé par le mathématicien L.E.J. Brouwer :

(*) On notera cependant qu'il y a des exceptions, des mathématiciens qui ne sont pas dupes de ce préjugé ; exemple : Takeuti [2] .

J. HARTHONG
Institut de Recherche Mathématique Avancée
Laboratoire Associé au C.N.R.S. n° 01
7, rue René Descartes
67084 STRASBOURG CEDEX

c'est que la construction de la mathématique formelle se fait dans le cadre d'une métamathématique qui présuppose au moins la suite intuitive (ou intuitionniste) des entiers naturels. Si après cela la mathématique formelle prétend avoir construit ces mêmes objets qu'elle n'a fait que restituer, elle mystifie. Par contre, si elle prétend fournir une théorie scientifique sur ces objets qui, eux, nous sont donnés en quelque sorte par la nature, elle rend un très grand service à la mathématique. La théorie formelle des ensembles est donc une invention remarquable, mais il faut la prendre pour ce qu'elle est : une théorie, c'est à dire un système logique fabriqué de toutes pièces par l'esprit, et qui permet, à partir d'un petit nombre de principes de base et de concepts, de retrouver par déduction toutes les propriétés observables des objets auxquels elle s'applique. Et puisque les ensembles infinis non dénombrables ne sont certainement pas des objets (Hilbert dixit) c'est à son adéquation avec les propriétés observables du continu par exemple qu'on pourra tester la théorie des ensembles. Notons bien que sur ce point, elle a donné jusqu'ici entière satisfaction, sauf qu'elle est bien compliquée.

Ainsi, vue de l'extérieur, par un observateur impartial, mais nécessairement intuitionniste (du seul fait qu'il s'est placé à l'extérieur), la théorie des ensembles n'a pas la même apparence que si on la regarde de son intérieur : elle est relativisée. C'est de l'extérieur qu'on peut poser la question célèbre : "les entiers naïfs remplissent-ils $\mathbb{N}$?"(1) à laquelle Reeb a répondu par la négative.

Je voudrais montrer dans cette communication que cette réponse est liée à un enjeu de taille : si on admet que les entiers naïfs ne remplissent pas $\mathbb{N}$, la seule théorie des ensembles finis suffit à rendre compte de toutes les propriétés du continu, et il est inutile de recourir à des ensembles non dénombrables.

Au départ nous admettrons donc l'arithmétique formelle de Peano, comportant les concepts bien connus $\mathbb{N}$, $\mathbb{Z}$, et $\mathbb{Q}$; en outre, nous admettrons tout théorème portant sur les ensembles finis de la théorie formelle des ensembles, c'est à dire l'analyse combinatoire. Et bien entendu, la clé sera l'élément ω de $\mathbb{N}$, non naïf.

1. Comment définir des nombres vraiment très grands

Parmi les opérations élémentaires de l'arithmétique, c'est l'élévation à une puissance qui donne les plus grands nombres. Si on veut avoir des nombres vraiment très grands, il faut élever à beaucoup de puissances. Pour cela on emploie la notation de Knuth : au lieu d'écrire a^b on écrira $a \uparrow b$, puis $a \uparrow\uparrow b = = a \uparrow (a \uparrow b)$, $a \uparrow\uparrow\uparrow b = a \uparrow^3 b = a \uparrow\uparrow (a \uparrow\uparrow b)$, et par récurrence

$$a \uparrow^n b = a \uparrow^{n-1} (a \uparrow^{n-1} b)$$

On peut ainsi construire une suite de nombres prodigieusement grands (procédure de Graham)

$G_0 = 10 \uparrow 10 = 10$ milliards

$G_1 = 10 \uparrow^{G_0} 10$

$G_2 = 10 \uparrow^{G_1} 10$

....

$G_n = 10 \uparrow^{G_{n-1}} 10$

Le nombre $\omega = G_{10}$ est si prodigieusement grand qu'il suffira pour la suite.

La procédure de Graham utilisée pour définir ω pourra vous sembler bizarre ; le fait est qu'on ne recontre jamais de procédure aussi délirante en mathématiques usuelles. Supposons qu'on s'interdise (2) (par discipline librement consentie) justement le recours à la notation de Knuth, on aurait une arithmétique bâtie à partir des opérations élémentaires +, x, et puissance. La notation a^b ne permettrait pas de désigner des nombres aussi grands que G_1. Mais elle permettrait d'englober toutes les fonctions qui ont jamais été introduites depuis que la mathématique existe : les polynômes, e^x, e^{x^2}, $e^{x^2 e^x}$ etc. Nous pouvons dire que la procédure de Graham, ou toute procédure qui utilise la notation de Knuth, est externe (elle n'intervient pas dans les mathématiques usuelles). Par contre, les procédures internes sont les procédures usuelles (addition, multiplication, exponentiation).

Tout nombre explicitement spécifié par une procédure interne (algorithme) est nécessairement inférieur à G_1 ; nous appellerons standard un tel nombre). De même, tout nombre défini par une procédure interne contenant la variable G_1 est inférieur à G_2 (car le passage de G_1 à G^2 est impossible sans faire appel à la notation de Knuth). Le nombre G_1, et plus encore $G_{10} = \omega$ est un nombre inaccessible pour la mathématique usuelle.

2. Une théorie du continu

Les procédures externes ne peuvent pas être exécutées par un ordinateur (essayez de faire exécuter la procédure de Graham, même si le nombre de base est 2 au lieu de 10). C'est pourquoi la meilleure définition qu'on puisse donner des procédures externes est : "procédure non exécutable par un ordinateur".

Cela n'empêche pas les procédures externes de fournir de nombreux concepts mathématiques parfaitement efficaces et opératoires. Comme premier exemple, voici trois notions qui en dérivent, puisqu'elles utilisent le concept de nombre standard, lui-même dérivé de la notion de procédure externe.

Soit Z l'ensemble des entiers relatifs.

1° Nous dirons que deux éléments k et l de Z sont équivalents (noté $k \simeq l$) si pour tout nombre standard n, $n\,|k-l| \leqslant \omega$. Cela signifie que $|k-l|$ est petit par rapport à ω, mais il peut être bien plus grand qu'un nombre standard.

Exemples : si n est standard, alors $n \simeq 0$; mais aussi $G_1 \simeq 0$, $G_2 \simeq 0$, ... $G_g \simeq 0$, $\log \omega \simeq 0$ (comme nous ne travaillons que dans le cadre strict de l'arithmétique, nous appelons $\log \omega$ le nombre N tel que $\omega = 10^N$). C'est une relation d'équivalence externe (i.e. réflexive, symétrique, et internalement transitive) les classes d'équivalence sont appelées les halos.

2° Nous dirons qu'un élément k de Z est limité s'il existe un nombre standard n tel que $|k| \leqslant n.\omega$.

Exemples :

ω, 2ω, 3ω sont limités ; mais pas ω^2, et encore bien moins G_{11}!

Théorème 1

a) Tout nombre inférieur à un nombre limité est lui-même limité.

b) Tout nombre $\leqslant$ à un nombre $\simeq 0$ est $\simeq 0$.

Démonstration

a) Si B est limité et $|A| \leqslant |B|$, il existe un standard n tel que $|B| \leqslant n\omega$; donc $|A| \leqslant n\omega$ CQFD

b) si $B \simeq 0$ et $|A| \leqslant |B|$: pour tout n standard, $n|B| \leqslant \omega$; donc $n|A| \leqslant \omega$ CQFD

3) Nous appellerons nombre réel le halo d'un élément limité de Z.

Exemples :

a) ω, 2ω , 3ω sont limités ; donc leurs halos $\widehat{\omega}$, $\widehat{2\omega}$, $\widehat{3\omega}$,... sont des nombres réels.

b) (moins trivial) Soit l'ensemble fini $E = \{(k,l) \in Z \times Z, k^2+l^2 < \omega\}$ et a son cardinal ; d'après un théorème connu de la théorie des ensembles finis, $a \in N$, donc à Z. De plus :

$$E \subset \{k \in \mathbb{Z},\ |k| < \sqrt{\omega}\} \times \{\ell \in \mathbb{Z},\ |\ell| < \sqrt{\omega}\}$$

(ω est un carré parfait !)

Donc d'après un autre théorème de la théorie des ensembles finis :

$$a \leqslant \left(\text{card}\,\{k \in \mathbb{Z},\ |k| < \sqrt{\omega}\}\right)^2 = (2\sqrt{\omega} - 1)^2 < 4\omega$$

cette inégalité signifie que a est limité. Donc son halo $\widehat{a}$ est un nombre réel, qu'on appelle π .

c) ω^ω et $(\omega+1)^{\omega+1}$ sont deux entiers naturels ; on peut donc faire la division euclidienne du second par le premier :

$$(\omega+1)^{\omega+1} = b\,\omega^\omega + r \quad , \text{ avec } \quad r < \omega^\omega$$

en appliquant la formule du binôme à $(\omega+1)^{\omega+1}$, on voit facilement que $b \leqslant 3\omega$, donc b est limité. Son halo $\widehat{b}$ est alors un réel appelé e.

Les nombres relatifs (les éléments de Z) ont une écriture décimale ; on peut en dériver une écriture décimale pour les nombres réels. Ainsi par exemple l'entier a défini ci-dessus (dont le halo est π) a une écriture décimale ; nous avions montré que $a \leqslant 4\omega$; avec des arguments similaires on pourrait aussi montrer que $a \geqslant 3\omega$, d'où on déduit que a est un nombre de N + 1 chiffres (rappelons que $\omega = 10^N$) dont le premier (le plus à gauche) est 3 ; une étude plus serrée permettrait de prouver que le second chiffre est 1, le troisième 4, etc... Bien entendu, il est de plus en plus difficile de calculer ces chiffres et certainement impossible de les connaître tous (on connaît actuellement les 8 millions premiers chiffres, ce qui est littéralement rien par rapport aux $N = \log \omega$).

Définition :

pour $k > 0$ nous appellerons mantisse de k la liste des N chiffres les plus à droite, éventuellement complétée par des 0 (du Nième à partir de la droite au 1er) et caractéristique la liste de tous les autres.

La mantisse est donc un nombre toujours compris entre 0 et $\omega - 1$, la caractéristique est un nombre qui peut être quelconque et on a toujours $k = m + 10^N c$.

Théorème 2

a) Pour qu'un nombre soit limité, il faut et il suffit que sa caractéristique soit un nombre standard.

b) Pour que deux nombres soient équivalents il faut et il suffit que leurs caractéristiques soient égales et que leurs mantisses soient identiques à partir de la gauche à tout ordre standard, à condition d'identifier des écriture telles que 1000...0 et 099...9.

Démonstration

a) Soit k limité et > 0 ; par définition cela veut dire qu'il existe un standard n tel que $k < n\omega$. Or cette inégalité veut dire que la caractéristique de k est $\leqslant n$ donc standard.

b) Soit $k = \sum_{j=0}^{M} \delta_j 10^j$, avec $\delta_j \in \{0, 1, \dots, 9\}$ l'écriture décimale de k. La condition b) du théorème se traduit ainsi

$$k \simeq 0 \iff \forall n \text{ standard},\ \forall j \geqslant N - n,\ \delta_j = 0$$

Or :

- n standard 10^n standard car 10^n est une procédure interne donc :
- $k \simeq 0 \iff \forall n \text{ standard},\ 10^n k < \omega \iff \forall n \text{ standard},\ k < 10^{N-n}$.

Remarque :

Ce théorème montre que si d'un nombre on ne retient que sa caractéristique et les chiffres de la mantisse accessibles à partir de la gauche, on obtient un résultat qui ne dépend que du halo de ce nombre ; ceci permet donc la :

Définition

On appelle écriture décimale d'un nombre réel la liste (illimitée vers la droite) de chiffres obtenus ainsi : on trace une virgule ; à gauche de cette virgule on inscrit la caractéristique d'un représentant quelconque, et à droite de la virgule les chiffres de la mantisse aussi loin qu'il est accessible. Ainsi pour π:

$$\underbrace{3,}_{\text{caractéristique}}\ \underbrace{1\,4\,1\,5\,9\,2\,6\,5\,3\,5\ \ldots}_{\text{mantisse (inépuisable)}}\quad .$$

3. Fonctions continues

Nous allons considérer toutes les fonctions f de Z dans Z, c'est-à-dire l'ensemble usuellement noté Z^Z (Mais on pourrait se contenter sans aucune perte de E^E où E serait $\{k \in Z,\ |k| \leq G_{11}\}$ et qui, lui, serait fini).

Définition :

Nous appellerons fonction continue au point k (de Z dans Z ou de E dans E) une fonction f ayant la propriété :

$$\forall\, l \simeq k, \quad f(l) \simeq f(k)\ .$$

Théorème 3 (des valeurs intermédiaires)

Soient a et b deux éléments limités de Z, telle que $f(b) > 0$ et $f(a) < 0$ mais $f(a) \not\simeq 0$ et $f(b) \not\simeq 0$, et f une fonction continue dans $[a,b]$; alors il existe $k_0 \in [a,b]$ tel que :

$$\forall j \simeq k_0, \quad f(j) \simeq 0\ .$$

Démonstration

Soit $F = \{k \in [a,b],\ f(k) > 0\}$.

F n'est pas vide car b F ; donc F a un plus petit élément k_0 ; alors $k_0 \in F$, mais $k_0 - 1 \notin F$,
c'est-à-dire que $f(k_0) > 0$ et $f(k_0 - 1) \leq 0$; or $k_0 \simeq k_0 - 1$, donc $f(k_0) \simeq f(k_0-1)$ et cela n'est compatible avec les deux inégalités précédentes que si $f(k_0) \simeq 0$ CQFD

4. Fractions

La considération de fonctions à valeurs entières n'est pas a priori la plus naturelle, car les fonctions sont définies par des algorithmes qui donnent souvent des valeurs fractionnaires.

Exemples

$$\exp(k) = \left(1 + \frac{1}{\omega}\right)^k$$

$$\sin(k) = \mathcal{I}m\left(1 + \frac{i}{\omega}\right)^k$$

$$\mathrm{Log}(k) = \sum_{j=\omega}^{k} \frac{1}{j} \qquad \left(= -\sum_{j=k}^{\omega} \frac{1}{j} \quad si \quad k < \omega\right)\ .$$

C'est pourquoi la considération de fonctions à valeurs dans Q est plus commode, mais voici des considérations qui montrent l'équivalence entre les deux points de vue :

Définition

Soit une fraction irréductible $\frac{p}{q} = r$ et $N(r)$ le quotient euclidien de ωp par q : $\omega p = N(r)q$ + reste ; on appelle $N(r)$ la représentation entière de r ; on dit que r est limité si sa représentation entière est limitée, et que $r \not\simeq 0$ si $N(r) \simeq 0$; si $f : Z \to Q$ on lui associe $\hat{f} : Z \to Z$, $k \mapsto N(f(k))$.

Exercice

Montrer que

a) pour que r soit limité, il faut et il suffit qu'il existe un entier standard n tel que $|r| \leq n$.

b) pour que $r \not\simeq 0$ il faut et il suffit que pour tout n standard $n\,|r| \leq 1$.

c) si $f : Z \to a$, f est continue en k si et seulement si :
$l \simeq k \Rightarrow f(l) \not\simeq f(k)$.

Définition

Si r est limité, $\widehat{N(r)}$ est un réel appelé l'ombre de r et noté ^{o}r ou $St(r)$.

Théorème 4

Pour tout $k \in Z$ limité, $\left(1+\frac{1}{\omega}\right)^k$ est un élément limité de Q.

Démonstration

D'après la formule du binôme de Newton :

$$\left(1 + \frac{1}{\omega}\right)^k = \sum_{j=0}^{k} \binom{k}{j} \frac{1}{\omega^j} \quad ;$$

le terme général de cette somme peut s'écrire :

$$\binom{k}{j}\frac{1}{\omega^j} = \frac{k(k-1)\ldots(k-j+1)}{j!\ \omega^j} = \frac{1}{j!}\ \frac{k}{\omega}\ \frac{k-1}{\omega} \cdots \frac{k-j-1}{\omega}\ .$$

Par hypothèse il existe un entier standard N tel que $\left|\frac{k}{\omega}\right| \leq N$, donc on peut majorer le terme général ainsi :

$$\binom{k}{j}\frac{1}{\omega^j} \leq \frac{N^j}{j!}\ .$$

Notre somme peut alors se décomposer en deux morceaux

$$\sum_{j=0}^{k}\binom{k}{j}\frac{1}{\omega^j} = \sum_{j=0}^{2N-1}\binom{k}{j}\frac{1}{\omega^j} + \sum_{j=2N}^{k}\binom{k}{j}\frac{1}{\omega^j}$$

$$\leqslant \sum_{j=0}^{2N-1}\frac{N^j}{j!} + \sum_{j=2N}^{k}\frac{N^j}{j!}$$

dans le deuxième morceau on a toujours $j \geqslant 2N$, donc $j! \geqslant (2N)!\,(2N)^{k-2N}$

ce qui permet d'écrire

$$\sum_{j=0}^{k}\binom{k}{j}\frac{1}{\omega^j} \leqslant \sum_{j=0}^{2N-1}\frac{N^j}{j!} + \frac{N^{2N}}{(2N)!}\sum_{i=0}^{k-2N}\frac{N^i}{(2N)^i}$$

$$\leqslant \sum_{j=0}^{2N-1}\frac{N^j}{j!} + 2\,\frac{N^{2N}}{(2N)!} \quad ;$$

comme N est standard, il est clair que tous les termes de cette dernière expression sont limités, d'où la conclusion.

5. Dérivées

Soit $f : Z \to Q$. On appelle dérivée de f la fonction :

$$\partial_+ f \quad \text{ou} \quad f' : \quad \begin{array}{l} Z \longrightarrow Q \\ k \longmapsto \omega\,(f(k) - f(k-1)) \end{array} ,$$

il y a aussi la rétrodérivée

$$\partial_- f \quad \text{ou} \quad f'_R : \quad \begin{array}{l} Z \longrightarrow Q \\ k \longmapsto \omega\,(f(k) - f(k-1)) \end{array} ,$$

et la dérivée symétrique

$$\partial_o f \quad \text{ou} \quad f'_s : \quad \begin{array}{l} Z \longrightarrow Q \\ k \longmapsto \frac{\omega}{2}(f(k+1) - f(k-1)) \end{array} ;$$

bien entendu f'_s est la moyenne de f' et f'_R :

$$f'_s = \frac{f' + f'_R}{2} \quad ;$$

les dérivées d'ordre supérieur s'obtiennent en itérant les opérateurs $\partial_{+,-,o}$; exemples :

$$\partial_+^2 f(k) = \omega^2\,(f(k+2) - 2f(k+1) + f(k)) \quad ,$$

$$\partial_{+-}^2 f(k) = \partial_+\partial_- f(k) = \omega^2\,(f(k+1) - 2f(k) + f(k-1)) \quad , \text{ etc.}$$

N.B Une fonction a toujours des dérivées.

6. Equations différentielles

Définition

On appelle équation différentielle ou relation de récurrence une équation de la forme :

$$\partial_+ f(k) = F(k, f(k)) \, ,$$

où $F(k,r)$ est une fonction $Z \times Q \to Q$.

Théorème 5 (de Cauchy, d'existence et d'unicité des solutions d'une équation différentielle)

Si F est lipschitzienne, c'est-à-dire vérifie les inégalités

$$|F(k,r) - F(k,s)| \leqslant M\,|r-s| \quad (1) \qquad \text{et} \qquad |F(k,r)| \leqslant N \quad (2) \, ,$$

avec $M, N \in Q$ limités pour tout k limité, alors pour toute valeur initiale $f(0) = r_0$ limité, $f(k)$ reste limité tant que k est limité (c'est-à-dire que $f(k)$ a une ombre), et si f et g sont deux solutions telles que $f(0) \# g(0)$ alors $f(k) \# g(k)$ pour tout k limité.

Démonstration

Ecrivons l'équation sous la forme

$$f(k+1) = f(k) + \frac{1}{\omega}\,F(k, f(k)) \quad ;$$

remarquons que si $f(0)$ est connu, $f(k)$ est univoquement déterminé pour tout $k \in Z$

Supposons d'abord que $f(0) \# g(0)$.

Alors :

$$f(k+1) = f(k) + \frac{1}{\omega}\,F(k, f(k))$$

et

$$g(k+1) = g(k) + \frac{1}{\omega}\,F(k, g(k)) \quad ;$$

d'où en retranchant membre à membre :

$$f(k+1) - g(k+1) = f(k) - g(k) + \frac{1}{\omega}\left(F(k,f(k)) - F(k,f(k))\right) ,$$

et en appliquant l'hypothèse (1)

$$|f(k+1) - g(k+1)| \leqslant |f(k) - g(k)|\left(1 + \frac{M}{\omega}\right) .$$

On en déduit par récurrence que $\forall k \in N$,

$$|f(k) - g(k)| \leqslant \left(1 + \frac{M}{\omega}\right)^k |f(0) - g(0)| \, .$$

D'après le théorème 4, $\left(1+\frac{M}{\omega}\right)^k$ est limité si k est limité, donc $\left(1+\frac{M}{\omega}\right)^k\,|f(0) - g(0)|$ reste $\# 0$, et par conséquent aussi $f(k)-g(k)$.

De la même façon, la condition (2) implique que $f(k)$ reste limité.

7. Vers d'autres applications

De la même façon que précédemment, on peut envisager de recourir à des définitions dans un univers fini (mais très vaste) pour tous les concepts courants de l'analyse. Ainsi l'intégrale définie peut être très bien exploitée à partir de la définition suivante : soit $f : Z \to Q$; on dit que f est <u>intégrable sur Z</u> si $\frac{1}{\omega} \sum_{|k| \leq N} |f(k)|$ est limité pour tout $N \gg \omega$. Ce qui est particulièrement intéressant, c'est la proximité remarquable des concepts ainsi présentés, et mis en oeuvre, avec les "objets" que l'on est amené à faire fonctionner dans un travail effectif, sur ordinateur par exemple.

NOTES

(1) On peut considérer que cette question demeure posée depuis Brouwer, quoique sous une forme moins directe (ou moins brutale).

(2) <u>Note</u> (F. Pluvinage): On pourrait s'imaginer que la barrière indiquée par J. Harthong disparaîtrait moyennant l'usage de notations appropriées. Il n'en est rien: Même en s'autorisant des écritures à la Knuth, on bute sur des obstacles insurmontables; il est possible d'affirmer que les entiers trop grands sont <u>ontologiquement</u> intraitables.

Considérons à titre d'exemple la fonction d'Ackermann $A : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$, déterminée par

$$\begin{cases} A(0,n) = n+1 \\ A(m,0) = A(m-1,1) \text{ , si } m>0 \\ A(m,n) = A(m-1,A(m,n-1)) \text{ , si } m>0 \text{ et } n>0 \text{ .} \end{cases}$$

Il est connu que cette fonction, définie récursivement, atteint très vite des valeurs très grandes; ainsi:

$$A(4,4) = 2^{2^{2^{65536}}} - 3 \text{ .}$$

Espérer simplement comparer par exemple $\omega = G_{10}$ et $A(G_0,G_0)$ est sans doute déjà une manifestation d'optimisme excessif.

Bibliographie

[1] Georges REEB : <u>La mathématique non-standard, vieille de soixante ans ?</u> Publ. IRMA, 1978.

[2] G. TAKEUTI : <u>Two applications of Logic to Mathematics</u> (Princeton University Press, 1976).

[3] <u>Les entretiens de Zürich sur les fondements des sciences mathématiques.</u> Editeur : S.A. Lehmann frères & Cie, Zürich, 1941 .

[4] D. HILBERT : <u>Ueber das Unendlichen</u> . Mathematische Annalen, vol.95,1926, p 161 - 190 .

A FUNDAMENTAL COURSE IN HIGHER MATHEMATICS
INCORPORATING DISCRETE AND CONTINUOUS THEMES

by

Stephen B. Seidman and Michael D. Rice
Department of Computer and Information Sciences
George Mason University
Fairfax, VA 22030, USA

Paper prepared for presentation to the Symposium on
the Influence of Computers and Informatics on
Mathematics and its Teaching
Strasbourg, March 25-30, 1985

I. The Current Mathematics Curriculum: Traditions and Concerns

For many years, a crucial place in the mathematics curriculum of the last year of secondary school or the first year of university studies has been occupied by the differential and integral calculus. The calculus can be seen both as the culmination of the secondary school mathematics curriculum and as the beginning of the serious study of mathematics in the university. In some sense, the study of calculus has become synonymous with the serious study of mathematics. The central and essential position occupied by calculus can be traced to at least two interrelated causes.

For mathematicians, calculus represents the methodology and techniques needed for the study of functions, first defined on the real line, then on higher-dimensional Euclidean spaces, and finally on the complex plane. Thus the study of the calculus allows students for the first time to acquire the formal and abstract tools that are essential for the further study of higher mathematics.

On the other hand, calculus provides the foundation for the application of mathematics to the physical sciences and engineering. These applications date back to Newton's original development of the calculus in the seventeenth century, and since that time they have been wildly successful across a vast collection of disciplines, even including (in recent years), the biological sciences and economics. All of the calculus-based applications are based on mathematical models that can be regarded as being <u>continuous</u>; that is, the quantities being modeled are real numbers (or elements of some Euclidean space R^n).

Given both the central mathematical position of the calculus and its vital role in applications (not to speak of the interaction between these two features), it is easy to see why the calculus has occupied such a fundamental and unassailable position in mathematics curricula. During the past several decades, however, the central role of calculus has been seriously questioned, and the questions have been repeated with particular emphasis during the last few years ([3]). Just as a major motivation for the predominance of calculus in the curriculum has been the wide range of the applications of continuous mathematics, the challenge to that predominance has arisen from the steadily increasing interest in the applications of discrete mathematics in many disciplines.

This increasing interest in discrete mathematical applications can be primarily attributed to the widespread use of computers. Computers are essentially discrete machines, and the mathematics that is needed to describe their functions and develop the algorithms and software needed to use them is also discrete. As a consequence, the discipline of computer science is heavily dependent on a wide variety of discrete mathematical ideas and techniques. Furthermore, the easy availability of computers has encouraged the use and development of discrete mathematical models in many disciplines. For one example, operations research models (linear programming, integer programming, etc.) are widely used and are based on a discrete mathematical perspective.

It is natural to expect that the rapid growth of interest in discrete mathematics and its applications, fueled by the explosive developments associated with computers, should have an impact on the mathematics curriculum. Although this impact would have been significant under any circumstances, its effect has been magnified by other questions that have been raised in the United States in recent years about the teaching of calculus. Widespread dissatisfaction has been reported with the nature of the calculus courses and the knowledge of the students that have completed them ([2],[5]). The computer is also directly influencing the content of the calculus course itself, both by encouraging the inclusion of numerical methods and by suggesting that symbolic manipulation software may make emphasis on techniques of differentiation and integration obsolete ([1],[6]).

In summary, both the nature of the calculus course and the fundamental position that calculus has occupied in the mathematics curriculum for more than a century have come under serious challenge. These challenges have come both from within and outside the community of mathematicians, and they can primarily be attributed to the increasingly broad role that computers are playing in the various scholarly disciplines represented within the university and in the wider world. In the next section of this paper, we will look at the responses that have been proposed to these challenges.

II. Responses to the Challenge of Discrete Mathematics

When any curriculum is confronted by a new topic that should be included, there are essentially two potential responses. The new topic can either be encapsulated in a course that is added to the curriculum, or it can be incorporated as a fundamental constituent of a revised curriculum. Most topics that have been added to the mathematics curriculum in recent decades have been added as new courses (e.g. abstract algebra and topology).

It was therefore natural that when mathematics faculties were asked to include discrete mathematics in the curriculum, this was most commonly done by developing new courses in discrete mathematics. Such courses were designed primarily for students of computer science. There were two fundamental problems with this approach. First, the discrete mathematics courses were usually taken by third-year students, so that the material was learned too late to be of use in the data structures courses taken by first and second year students of computer science. Second, when students were expected to use their discrete and continuous mathematical skills in fourth-year computer science courses (for example, in the analysis of algorithms), most have found it very difficult to combine these skills effectively. Many students do not see any connections between discrete and continuous mathematics, and are unable, for example, to apply calculus techniques to estimate growth rates of discrete functions or to estimate the size of discrete sums. This inability to combine discrete and continuous skills is also found in students of probability, operations research and signal processing.

Both of the above reasons suggest that discrete mathematics should be incorporated as a component of the fundamental mathematics course that is offered to all students in their first two years of university study. This suggestion was first made by Ralston ([3]), who proposed that the study of discrete mathematics <u>precede</u> the study of calculus. He argues that such an organization would benefit virtually all students of mathematics, and not just those students concentrating in computer science. Ralston's proposal has led to substantial discussion in the United States on the proper place of discrete mathematics in the curriculum ([4]). The debate has focused on whether discrete mathematics should <u>precede</u> or <u>follow</u> the calculus in the curriculum of the first two years. Many of the arguments advanced on either side are administrative in nature, dealing either with the demands of other curricula (such as physics or engineering) or with articulation with other institutions (such as high schools, junior colleges or universities that have retained the standard curriculum).

Whether calculus is placed before or after discrete mathematics, it is by no means clear that students who have completed both courses will be able to combine their discrete and continuous mathematical skills in an effective manner. This problem has been recognized by some designers of proposed curricula, and consequently their calculus proposals generally include some discrete aspects, such as extended discussion of numerical methods and substantial use of sequences (see for example [1]).

Another possibility, which is rarely given serious attention, would be to develop a new, unified curriculum that would interweave discrete and continuous themes throughout its courses. While the first year of the curriculum would correspond to the calculus course, its real thrust would be the study of functional behavior and functional representation. The course would consider discrete functions (sequences) along with continuous functions, and would constantly emphasize analogies and parallels between discrete and continuous situations. Thus the first year of the curriculum would be primarily continuous, but with a strong discrete flavor. The second year of the curriculum would focus on <u>structure</u>, and would be primarily discrete, but with a strong continuous flavor.

This paper will argue that a curriculum unifying discrete and continuous themes is not only feasible, but has the potential of providing students with a broad, powerful perspective embracing the mathematical ideas and techniques that are needed for the study of computer science. This perspective would also yield a strong mathematical foundation for the study of engineering, the physical sciences, and indeed for the study of higher mathematics itself.

Furthermore, the development of such a curriculum will force a reexamination of the topics taught in the conventional calculus course. As mentioned above, various recommendations have been made to remove or include particular topics. Although each such recommendation has been solidly grounded, no consistent rationale has been given for the collection of topics that together make up the proposed calculus course. The first-year course outlined below has a consistent theme - functional behavior and representation - and each topic to be included in (or excluded from) the course should be judged on the degree that it matches the course's perspective.

In the following section, a detailed outline and discussion will be given only for the first year of the proposed two-year curriculum. At the conclusion of the paper, we will return to the second year of the curriculum, as well as to the larger issues raised by the question of articulation with other curricula.

III. <u>A First-Year Curriculum Incorporating Discrete and Continuous Themes</u>

The fundamental thrust of the proposed first-year curriculum is the <u>behavior</u> <u>and</u> <u>representation</u> <u>of</u> <u>functions</u>. Roughly, the first semester is devoted to tools for the description and analysis of functional behavior, with the focus shifting to representation of functions in the second semester. Before presenting a more extended discussion of the benefits to be achieved by including both discrete and continuous topics, it will be useful to give an annotated outline of the first semester curriculum.

A. FUNCTIONS

1. Numbers and Relations

A knowledge of set concepts and notation is assumed. Inequalities will be emphasized.

2. Functions and Operations

Function concept and functional notation will be introduced, stressing the algorithmic interpretation of the function symbol f. Discussion will include domain and range, operations on functions (arithmetic operations, composition, translation), and graphs of functions. Useful functions will be introduced (polynomials, rational functions, exponential functions (defined on the integers), absolute value, floor, ceiling).

3. Models

Algorithms and elementary complexity analysis will be introduced (including binary search). This will allow discussion of the function $\lceil \lg(n) \rceil$. Models demonstrating the need to construct functions and to perform curve fitting will be included.

B. BEHAVIOR OF DISCRETE FUNCTIONS

1. Sequences: Iteration and Recursion

This section will include a discussion of geometric series. Examples will include the Fibonacci numbers and the greatest common divisor function.

2. Difference Operators

The difference operator Δ will be introduced as a function on sequences. The recursion scheme $u_{k+1}-u_k = Au_k$ will be treated in order to emphasize the special role of exponential functions (defined on the integers). Formulas for higher differences will be discussed.

3. Summation

The primary topic here will be the binomial theorem, both in its standard form and in the expression for $(1+\Delta)^n$. The second form will allow various formulas for finite sums to be presented.

4. Landau Notation (O,o) and Limits of Sequences

C. BEHAVIOR OF CONTINUOUS FUNCTIONS

1. Limit Heuristics

Limits of functions will be discussed only in terms of limits of sequences. The continuity concept will be introduced. The operator $\Delta_h f = (f(x+h)-f(x))/h$ will be introduced. Analogies to the discrete difference operator discussed above will be pursued.

2. First Derivative

The derivative will be defined, and interpreted using tangent lines. It will be shown that differentiable functions are continuous.

3. Differentiation Rules

Powers and roots; product and quotient rules.

4. Monotone Functions and Local Extrema

A rigorous treatment will be postponed. Curve sketching will be introduced here.

5. Second Derivative

Concavity will be discussed and applied to curve sketching.

6. Extreme Values

Maximum-minimum problems will be solved. Examples will also demonstrate the use of piecewise linear functions.

7. Related Rates

The chain rule will be presented, and related rate problems will be solved.

D. ESTIMATION AND ERROR

1. Mean Value Theorem

Monotone functions will be discussed more rigorously, and the MVT will be applied to global estimation of functions.

2. Solution of Equations

Newton's method will be discussed from both geometric and iterative perspectives. An elementary treatment of error estimation will be given, and critical values will also be estimated.

3. Interpolation

Interpolation of functions by lines and parabolas will be discussed, using the difference operators developed above.

4. Approximation

Second-order Taylor polynomials will be used to approximate functions, and the estimated error will be computed. Analogies will be drawn between interpolation and approximation and between differences and derivatives.

E. INTEGRATION

1. Introduction

The summation operator for sequences will be introduced. Its relation to the difference operator will be discussed. It will be treated as an aggregation operator, and used to motivate the discussion of area.

2. The Definite Integral

This will first be introduced using a piecewise linear definition. This definition will then be applied to step functions. The area definition will then be presented, and applied to parabolas using the results on finite sums obtained above. Some elementary properties of the definite integral will be presented, including the mean value theorem for definite integrals.

3. The Indefinite Integral

This will be explicitly computed for step functions, piecewise linear functions and parabolas.

4. The Fundamental Theorem of Calculus

This will be derived from the mean value theorem for definite integrals. The chain rule will be applied to investigate some properties of the integral of $1/x$.

5. Evaluation of Integrals: Analytic Techniques

Substitution techniques will be discussed, as well as the use of integral tables.

6. Evaluation of Integrals: Numerical Techniques

The trapezoidal rule and Simpson's rule will be discussed. It will also be shown how integrals can be estimated using inequalities, and how sums can be estimated using integrals.

7. Applications of Integration: Aggregation

The applications to be treated include work and volume.

8. Applications of Integration: Modeling

The primary theme here will be the recognition of Riemann sums in differing situations. Examples will be taken from arclength and fluid flow. The basic point will be that a model generates a discrete (Riemann) sum, which can then be approximated by a definite integral.

Although this annotated outline gives a good overview of the first semester of the proposed course, it is too brief to show how the interweaving of discrete and continuous themes can lead to major benefits. The following examples are meant to be typical of the perspective that will be possible within this course structure.

<u>Example 1</u>: At the beginning of the course, the discrete exponential function, $f(n) = 2^n$, will be introduced, along with its one-sided inverse, $g(n) = \max\{ k \mid 2^k \leqslant n\}$. The function $g(n)$ is vitally important in computer science; for example, $g(n)+1$ is the worst-case number of comparisons in a binary search of a list of length n. The growth rate of $g(n)$ is important, and is usually treated (via calculus) using l'Hospital's rule. We suggest a discrete approach, based on the binomial theorem. Clearly $2^{g(n)} \leqslant n$, so that $g(n)/n \leqslant g(n)/2^{g(n)}$, and $g(n)$ approaches ∞ with n since $g(2^L) = L$. Thus it is only necessary to look at the behavior of $k/2^k$ as $k \to \infty$. By the binomial theorem $2^k = (1+1)^k \geqslant k(k-1)/2$, and hence $k/2^k \leqslant 2k/k(k-1) = 2/(k-1)$, which gives the desired result. The simplicity of the discrete argument should aid the student in learning, understanding and assimilating the growth rate of the <u>continuous</u> logarithm.

<u>Example 2</u>: The syllabus outline has referred to analogies between the discrete difference and summation operators on the one hand, and differentiation and integration on the other. For one example, the difference operator is defined on the sequence $\{u_n\}$ by $\Delta u_n = u_{n+1} - u_n$. If we define a function on the integers by $x^{(m)} = x(x-1)\ldots(x-m+1)$, then it is easy to see that $\Delta x^{(m)} = mx^{(m-1)}$, $\Delta^2 x^{(m)} = m(m-1)x^{(m-2)}$, and finally that $\Delta^m x^{(m)} = m!$ and $\Delta^{m+1} x^{(m)} = 0$. Thus the behavior of the difference operator (and its iterates) on the polynomials $\{x^{(m)}\}$ is strongly analogous to the behavior of the differentiation operator (and its iterates) on the polynomials $\{x^n\}$. Furthermore, since each collection of polynomials provides a basis for the vector space of polynomials of degree at most n, an example has been introduced which will be useful in a later course in linear algebra.

One further benefit of the use of difference operators is the natural observation that $\Delta 2^n = 2^n$, or more generally that $\Delta k^n = (k-1)k^n$. This suggests that exponential functions, whether discrete or continuous, may have a special role to play with respect to difference or derivative operators, and serves to motivate the later observation that $d/dx(e^x) = e^x$.

<u>Example 3</u>: The first two examples used discrete ideas to motivate continuous concepts that are to be introduced later. In this example, continuous techniques are used to obtain a discrete result. The identity giving the sum of a geometric progression, $\sum_{k=0}^{n-1} x^k = (x^n-1)/(x-1)$ can be differentiated using the quotient rule to obtain the identity

$$\sum_{k=1}^{n-1} kx^k = ((n-1)x^{n+1} - nx^n + x)/(x-1)^2.$$

Using this identity, it is immediate that $\sum_{k=1}^{n-1} k2^k = (n-2)2^n+2$ and that $\sum_{k=1}^{n-1} k2^{-k} = 2 - (n+1)/2^{n-1}$. The last result yields $\sum_{k=1}^{\infty} k2^{-k} = 2$, since it has already been shown that $k/2^k \to 0$ as $k \to \infty$. This example serves to remind students that continuous techniques can be important in discrete situations.

These examples demonstrate that the proposed course does not merely insert a collection of important discrete topics into the calculus course, but rather expresses a consistent approach to all of the subject matter. The fundamental perspective is the study of functional behavior, and both discrete and continuous functions are treated throughout. Each class of functions is used to develop tools and suggest analogies that will be useful for the study of functions of the other class.

The second semester of the course further elaborates its functional perspective. Rather than give a detailed, annotated outline, we will discuss the topics to be covered and describe how they relate to the themes developed during the first semester. The second semester is primarily devoted to material taken from two broad categories: special functions and representation of functions.

Exponential and logarithmic functions will be treated in depth. The natural logarithm will be introduced using the definite integral, and its properties will be investigated. The <u>inverse</u> of the logarithm will be motivated using growth models and the differential equation $dy/dx = ky$, and the relationship of this inverse to the exponential function will be motivated using difference equations and the <u>discrete</u> logarithm. Finally, the properties of the function e^x will be developed. Numerical estimates for exponential and logarithmic functions will be used throughout the discussion.

The next major topic will be trigonometric functions. Here the primary motivation will come from the geometry of the circle and from models of circular and harmonic motion, although discrete periodic functions, such as <u>mod n</u>, will also be used. The properties of the trigonometric functions will be developed. Integration by parts will be introduced and applied to the special functions. The special integrals leading to the inverse trigonometric functions will be introduced here. Mathematical models suggesting the use of trigonometric polynomials will also be used.

Now that the special functions have been treated, it will be possible to discuss infinite limits and improper integrals. L'Hospital's rule will be treated, and more discussion of Landau's (O,o) notation will be presented.

At this point, the focus will shift somewhat from functional <u>behavior</u> to functional <u>representation</u>. Thus the next major topic will be infinite series, with particular emphasis on the use of Taylor series to represent functions. Generating functions for simple recursions will be discussed, as well as connections with improper integrals. Serious attention will be paid to computational issues and to the estimation of error terms. The constant theme will be the use of Taylor series as function approximations to obtain information about functional behavior that would otherwise be very difficult to obtain.

The final major topic will be trigonometric series, with particular emphasis on the representation of functions using Fourier series. It will be emphasized that the choice of representation implies a decision about the type of functional behavior that is to be represented. The treatment of Fourier series at this early point will require the introduction of complex numbers, which will reinforce the students' geometric understanding of trigonometric functions. Furthermore, the availability of Taylor series will permit analytic as well as geometric discussion of the identity $e^{ix} = \cos x + i \sin x$. Finally, the early introduction of Fourier series will make it possible to discuss discrete Fourier series and the fast Fourier transform at a far earlier point in the curriculum than is presently possible.

Clearly, the focus on functional behavior and representation has produced a first-year course that is rather different from what is currently taught. The essential core of the current calculus course has been retained, but it is always made clear that it is there because it throws a powerful spotlight on functional behavior and representation. Conversely, many traditionally taught topics have been removed. This pruning was only possible because the developers approached each topic with the same question: how does this topic impact on the main theme of the course?

Now that the course has been outlined, it remains to be seen how it will fit into the curriculum. We will also have to pay some attention to the second-year course that will follow this course, and also to the political and institutional problems that its adoption would pose.

IV. Implications for the Curriculum

The first question to be addressed is the audience to be served by the proposed course. It is clearly ideally suited for students of computer science, since it merges themes from continuous and discrete mathematics in a synergistic manner. Students who have successfully completed the course can be expected to handle the mathematics arising (for example) in the analysis of algorithms. It can also be argued that this course would be well suited as a first course for students of mathematics, the physical sciences and engineering. For these disciplines, the major omission has been vector geometry and multivariate calculus. In many universities, a large proportion of this material is treated in the second year, and it is not unreasonable to suppose that even more could be shifted to a third-semester course designed for those students.

Although much vitally important mathematics can be subsumed under the general heading of "functions" , an equally important heading is that of "structure". While the proposed course is intended to give students the most important tools that come under the "function" heading, it does not address "structure". For students of computer science, "structure" and "function" are equally important, and thus an important place in their education must be found for "structure". Much of the debate summarized above on the place of discrete mathematics in the curriculum can be seen as a debate on the place of "structure" in the curriculum. Following on the first-year course that has been outlined above, it is reasonable to believe that a second-year course focusing on "structure" can be developed.

Such a course will not be described here, but it is possible to discuss briefly what general topics might be included. The primary strands might be discrete mathematics, linear algebra and probability theory. Discrete mathematical topics could include relations, graphs, Boolean algebras and formal languages. The discussion of linear algebra could include some multivariate calculus, which could then be applied in the probability portion of the course. Just as with the first-year course, the topics included in the second-year course should be chosen because they illustrate vital structural themes or because they are motivated by or permit the development of important applications.

The introduction of courses designed along these lines will not be a simple matter. The obstacles that will be found will range from the need for new textual materials to the difficulty of articulating the new courses with other institutions on all levels. It would be an unfortunate mistake, however, to conclude that because of the certainty of encountering what seem to be insuperable obstacles to the introduction of a truly new curriculum, the only possible strategy is one of incremental change. The development and introduction of a curriculum integrating discrete and continuous ideas is an exciting challenge, and one that is sure to be taken up in several places. What is really needed is a collection of design and development experiments, performed in out-of-the-way "protected" environments. Once a new curriculum has proven its viability and worth in one or more of these experimental environments, it will be time to address the structural and institutional issues involved in transplanting the successful curriculum to less protected situations.

References

1. Donald Bushaw, A Two-Year Lower-Division Mathematics Sequence, in The Future of College Mathematics, Anthony Ralston and Gail S. Young, eds., Springer-Verlag, New York, 1983, pp. 111-118.

2. Jack Lochhead, The Mathematical Needs of Students in the Physical Sciences, in The Future of College Mathematics, Anthony Ralston and Gail S. Young, eds., Springer-Verlag, New York, 1983, pp. 55-69.

3. Anthony Ralston, Computer Science, Mathematics and tne Undergraduate Curriculum in Both, American Mathematical Monthly, 88, 1981, pp. 472-485.

4. Anthony Ralston and Gail S. Young, eds., The Future of College Mathematics. Springer-Verlag, New York, 1983.

5. Lynn Arthur Steen, Developing Mathematical Maturity, in The Future of College Mathematics, Anthony Ralston and Gail S. Young, eds., Springer-Verlag, New York, 1983, pp. 99-107.

6. Herbert S. Wilf, Symbolic Manipulation and Algorithms in the Curriculum of the First Two Years, in The Future of College Mathematics, Anthony Ralston and Gail S. Young,eds., Springer-Verlag, New York, 1983, pp. 27-40.

DISCRETE MATHEMATICS

- Two years experience with an introductory course -

T. A. Jenkyns and E. R. Muller

Brock University, St. Catharines, Ontario, Canada, L2S 3A1

Introduction

The computer, more than any other scientific or technological development, is raising concerns about the undergraduate mathematics curriculum [1]. Many of us have grown up with the present first two years of the undergraduate curriculum where some gradual changes have been introduced, mainly on the Algebra side. It is therefore not surprising that we are nervous about a rapidly growing area of knowledge gnawing at these very courses. What makes things even more difficult is that the computer technology is developing so fast that we cannot afford to sit back for long waiting for the discipline to stabilize. If we do, we will probably lose students in mathematics and will also find a proliferation of mathematics courses in computer science departments. We are of the opinion that mathematics should be taught by mathematicians who in turn should be aware of developments around them and, where appropriate, the needs of their clientele [2].

Generally the mathematics curriculum has been developed with an eye on applications in the physical sciences. Lip service has been paid to applications in other areas with changes in the undergraduate curriculum coming mainly in the senior years. Such developments are not surprising as the sciences have traditionally required a strong component of mathematics within their programs. We have argued [3] that the calculus course, with some different emphases, should remain as a core course in the first two years of the undergraduate program. What concurrent core mathematics course should now be developed as we keep the other eye on the computer? At this time it appears that Computer Scientists are prepared to also require a strong component of mathematics within their program. To meet this demand and to offer our own students a broad mathematics education we must survey the field of mathematics and search out the appropriate areas. We follow others, e.g. Ralston [4], in isolating two fundamental areas that an undergraduate should be exposed to early. The first is an algorithmic way of thinking, the other, less well defined, are the mathematical concepts presently applicable in computer science which we group under the term "discrete mathematics". The first year course we have taught at Brock for the past two years, and to which 500 students have been exposed, emphasizes and explores these two areas. A detailed course outline is presented in Appendix I.

Course Philosophy

Algorithms are fundamental to much of the arithmetic and algebra taught in school. Unfortunately, the algorithms are rarely made explicit or discussed. They are the basis of learning by rote and the teacher's method becomes the only acceptable method, however inefficient or cumbersome it may be. These familiar algorithms form a valuable source of examples which can be used to motivate the fundamental characteristics of algorithms, for example, that for all inputs they terminate in a finite number of steps. One is naturally led to proving that they are effective, i.e., they produce the correct answer where a "correct solution" is often an approximate solution with some bound on the error. These algorithms can also be used to motivate initial discussions on efficiency. Furthermore, introductory ideas of iterative and recursive procedures are easily motivated, for a simple example consider the Short Division Algorithm stated traditionally as

Given integers $D > d$ and $d > 0$ there exists unique integers $Q > 0$ and R such that

$$D = Q*d + R \text{ and } 0 \le R < d.$$

Of interest is the algorithm which for a given D and d will generate Q and R, i.e. for an Input of D and d will Output Q and R.

Two contrasting algorithms are:

```
(i)  Iterative
     Proc     sd(D, d, R, Q)
              Q ← 0; R ← D
              Repeat
                   Q ← Q + 1
                   R ← R - d
              until [R < d]

(ii) Recursive
     Proc     SD(D, d, R, Q)
              If D < d then Q ← 0; R ← D
              else SD(D - d, d, R, Q); Q ← Q + 1
```

Stepping through these algorithms for D = 27 and d = 7 we find

```
(i)  Q      R     R < d      (ii)  SD(27, 7, , )D < d No
     0     27                      SD(20, 7, , )No
     1     20     No               SD(13, 7, , )No
     2     13     No               SD( 6, 7, , )Yes
     3      6     Yes              So we build back up through
     Output Q = 3, R = 6           the stack.
                                   SD( 6, 7, 6, 0)
                                   SD(13, 7, 6, 1)
                                   SD(20, 7, 6, 2)
                                   SD(27, 7, 6, 3)
                                   and Output R = 6 and Q = 3.
```

From the beginning of the course students develop algorithms, step through them for different inputs and follow through proofs of their effectiveness.

Deductive logic can be introduced with computer language constructs and it forms a useful base for a discussion of the structure of proofs. Students in introductory courses are rarely exposed to a discussion of mathematical proofs. We somehow assume that they will learn by repeated exposures as proofs are presented in their various courses. We review traditional proof constructs, mathematical induction playing a major role, and compare them with proofs for algorithms where the algorithm logic is used in the proof. We have found very little on this, in the literature, which is of use to an introductory course. The following proof constructs for the iterative and recursive Short Division Algorithms is what we are looking for.

(i) The iterative algorithm is effective, i.e.

(a) it stops for all inputs (with correct values for Q and R)
(b) the output is unique for any given input.

For ease of proof we rewrite the algorithm as follows

1. Set $Q_0 = 0$ and $R_0 = D$
   (then $R_0 \geq 0$ and $D = Q_0*d + R_0$)

2. If $R_j < d$ then stop

   else $Q_{j+1} = Q_j + 1$ and $R_{j+1} = R_j - d$

   (then $D = Q_{j+1}*d + R_{j+1}$

   $= Q_j*d + d - R_j - d$

   and $R_{j+1} = R_j - d \geq d - d = 0$)

a) The procedure must stop since

$$Q_j*d \geq Q_j*1 = j$$

and if it didn't, we would have after D iterations

$$D = Q_D*d + R_D \geq D + R_D \geq D + d > D.$$

b) Suppose the algorithm stops after k iterations with

$$D = Q_k*d + R_k \text{ and } 0 \leq R_k < d$$

and suppose that Q_k is not unique, then there exists a q which either

(i) $0 \leq q < Q_k$

then

$$D = Q_k*d + R_k = q*d + (Q_k - q)*d + R_k$$

$$= q*d + r \text{ hence } r \geq d$$

or (ii) $Q_k < q$ then $D = q*d + r$

$$\geq Q_k*d + d + r$$

$$> Q_k*d + R_k + r$$

$$= D + r \text{ hence } r < 0.$$

Thus Q_k with $0 \leq R_k < d$ is unique.

(ii) The Recursive Algorithm is effective.

a) The procedure must stop since successive values of D form a strictly monotonic decreasing nonnegative finite sequence with difference

$$D_k - D_{k-1} \geq 1$$

b) By induction on D we prove that the algorithm does always produce the correct result.

1. If D = 0 then D < d and result would be correct.

2. Assume correct for

   $D = 0, 1, 2, \ldots, k.$

3. Then for $D = k + 1$

   if $k + 1 < d$ then result is correct, and

   if $k + 1 \geq d$ since $d > 0$

   $0 \leq D - d \leq k$

   and therefore by assumption (2) the result is correct, i.e.

   $$D - d = Qd + R \text{ and } 0 \leq R < d$$

   $$D = (Q + 1)d + R$$

   with unique $(Q + 1)$ and R.

The initial and continuing effort has been to use the algorithmic way of thinking as an underlying theme. Clearly for some topics, e.g. graphs, counting and generating, etc. this is not difficult while for others, e.g. sequences and series, it is not so natural and perhaps not desirable. These can be used to contrast and underline the importance of the non algorithmic mathematics. As we review the course we are also finding common links (for example, graphs, recurrence relations) between what were initially separate topics. We have found it useful to conclude the course with an introduction to automata as we can return to the fundamental properties of algorithms.

DISCUSSION. This course is a required course for Computer Science and Mathematics/Computer Science combined majors. The latter take a full year Calculus course concurrently and a half course in Linear Algebra in the second year. The former take a half course in Calculus normally combined with a half course in Linear Algebra in the second year. We have, up to now, had close to 500 students through this course and we intend to present at the conference correlations between performances in this course and the Calculus and Linear Algebra courses. We are fortunate that a large proportion of the students in this course take a Pascal course concurrently. We still regard the content and emphasis of this courses as experimental. Even if there was an agreed set of topics the content within each has so many possibilities that we have not reviewed all the alternatives. For example in the section in graphs we have concentrated on paths and trees, in the section on counting and generating, we have selected algorithms which reflect lexicographic ordering. The section on probability and discrete random variables is well motivated by average case analysis of algorithm efficiency.

We are encouraged by the support we are receiving both from our Mathematics Department and the Department of Computer Science to continue to develop this course. We look forward to sharing the experience of other mathematicians who are attempting similar changes to the mathematics curriculum of the early undergraduate years.

Bibliography

[1] for e.g. "The Future of College Mathematics" ed. A. Ralston and G.S. Young - Springer Verlag, New York, 1983.

[2] "Motivating non-mathematics majors through discipline-oriented problems and individualized data for each student" J.W. Auer et al, Int. 3. Math. Educ. Sci. Technol., 13, 1982, 221-225.

[3] B.R. Hodgson et al., paper sent to this I.C.M.I. symposium.

[4] "Computer Science, Mathematics and the Undergraduate Curricula in Both" A. Ralston, Amer. Math. Mon., 88, 1981, 472-485.

APPENDIX

SECTION I - ARITHMETIC ALGORITHMS AND THEIR ANALYSIS

Basic definition of algorithms. Algorithms for multidigit addition and subtraction, long and short division; intuitive introduction to effectiveness, approximation, iteration and recursion. Primality and Factorization - intuitive introduction to efficiency - worst case analysis. Highest Common Factor (Euclid's algorithm and its efficiency). Square roots (Doubling of digits method and its rationale, more general methods for solving $x^2 = A$, bisection method and Newton's method (without proof), stopping rules and rounding).

SECTION 2 - NUMBERS AND MACHINE ARITHMETIC

Computer representation of numbers. Review of Sets (basic definitions including partitions). Binary Relations (basic definitions and examples of Cartesian Product, Relation, Function, Characteristic function, Partial and Total Ordering, Equivalence, matrix and graphical representations of relations). Real Numbers (review of definitions). Positional Representation of Numbers (the importance of positional representation, introduction to different bases). Round-off errors (errors and their behaviour under the four arithmetic operations). Bases, Conversion and Arithmetic (simple algorithms for converting from one base to another and arithmetic in any given base). Modulus Function, Floating point representations, Complement Arithmetic (basic definitions and an application to complement addition for subtraction).

SECTION 3 - FORMAL LOGIC AND PROOF TECHNIQUES

Formalizing arguments, axioms and rules of inference. Propositional calculus (definition of Boolean variables and the operations, $\sim, \cap, \cup, +, \rightarrow, \leftrightarrow$. Conventions for evaluating complex expressions and other basic definitions required for proof techniques). Is it correct? -- or how to prove it (proof techniques, including 1. Valid argument forms. 2. Direct proof. 3. Indirect proof. 4. Construction. 5. Counter - example. 6. Mathematical Induction (1st and 2nd principles)).

SECTION 4 - SEARCHING AND SORTING

Application of proof procedures and further efficiency analysis. Searching Lists (algorithms for searching random and sorted lists with their analyses). Sorting a List (Minsort, Bubblesort and Mergesort and their analyses - concept of a recursive algorithm). Slow Algorithms (Examples of the Tower of Hanoi and Travelling Salesman problem). Complexity of Algorithm (summary of complexity of algorithms studied to date with a graphical representation of the growth curves of these functions).

SECTION 5 - SEQUENCES AND SERIES

Concepts have been introduced intuitively before - arithmetic algorithms produce either sequence or series approximations. Sequences (definitions, limits of infinite sequences). Series (definitions, convergence). Some discussion on order of convergence,numerical limit computations and round off errors.

SECTION 6 - RECURRENCE RELATIONS - DIFFERENCE EQUATIONS

Basic definitions, applications. Homogeneous with Constant Coefficients (theory and solution of first order - Example [Fiobanacci] of second order with solution. First Order Nonhomogeneous with Constant Coefficients (solution procedure and applications). Examples of other types of Recurrence Relations (Pascal's triangle, calculator functions, deterministic simulation).

SECTION 7 - ITERATIVE AND RECURSIVE ALGORITHMS

Review and definitions. Iterative procedures (Examples of iterative algorithms - zeros of functions by fixed point methods). Programming recursion (examples of recursive procedures).

SECTION 8 - COUNTING AND GENERATING SETS AND SEQUENCES

Examples where previously counting and generating procedures were used - what procedures to look for. Four Basic Principles of Counting (addition and multiplication, inclusion-exclusion, pigeonhole principles. Sequences of length n from set X with $|X| = n$ (procedures for counting and generating sequences of length n - extensive section which discusses various algorithms both iterative and recursive, defines new concepts such as permutations, natural order permutations, signatures, minimal differences, etc.) Counting and Generating Sets (similar in style to the previous section it brings in new concepts - Gray code, next in lexicographic order, etc.). Sequences with limited repetition of elements (small section on counting). Counting and Recurrence Relations (examples of Recurrence Relations in counting). Binomial and Multinomial Theorems.

SECTION 9 - PROBABILITY AND RANDOM VARIABLES ON DISCRETE SAMPLE SPACES

Difference between deterministic and probabilistic systems. Probability for discrete sample spaces (Basic definitions and axioms, equally likely and not equally likely simple events - Application of counting principles in the former - development of further probability axioms, conditional probability, independence). Random Variables and Probability Distributions (definitions, applications, basic discrete distributions, uniform, binomial, geometric, Poisson [the last two as examples of infinite sample space]). Expectation value and Average Case Analysis (definitions and applications to the basic probability distributions of the previous section, also to sequential and binary search algorithms). Random number generation (pseudo random numbers generation using the linear congruential method - an application to simulation).

SECTION 10 - GRAPHS AND TREES

Definitions and Examples (definitions, algorithms for generating a simple path, theorems on simple paths, Euler paths). Forest and trees (definition, algorithm, on connectedness, shortest path, minimum connector problem, depth-first vs breadth-first transversal).

SECTION 11 - ELEMENTARY AUTOMATA

Turing machines (examples, definitions, and simulations). A universal machine. The Halting Problem.

What Should a Discrete Mathematics Course Be?

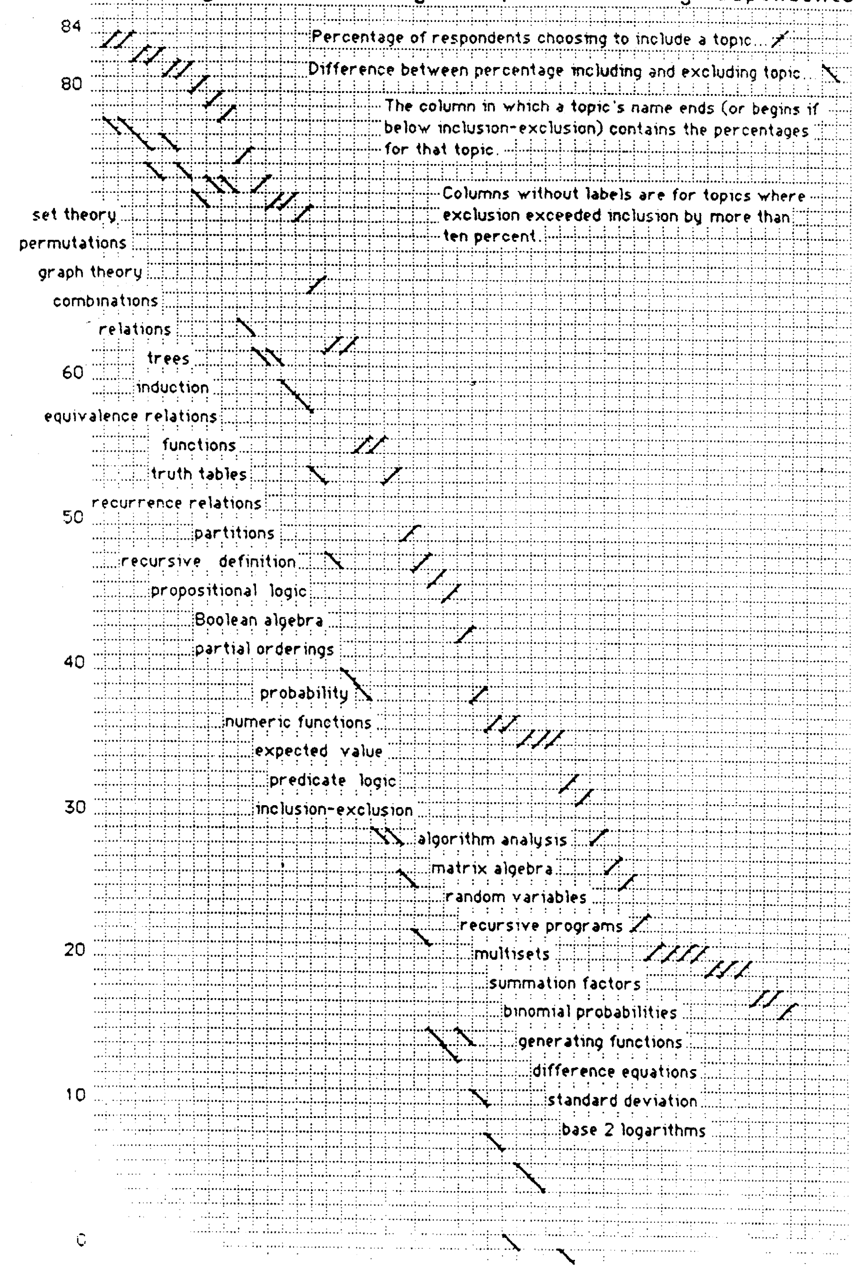
An answer to question 5 of ICMI study document "The influence of computers and informatics on mathematics and its teaching"

Kenneth P. Bogart

Introduction. In "What is a Discrete Mathematics Course?" [1], the author reported with Kathy Cordiero and Mary Lu Walsh on a joint survey research project to determine the extent and nature of freshman to sophomore level discrete mathematics courses. Virtually all our respondents (13% of the 3600 questionaires were returned) indicated that their institution has or contemplates a course in discrete mathematics. Most courses described have at most one programming course or one or two calculus courses as prerequisite, confirming that they may be regarded as freshman-sophomore level courses. Though some of the courses are aimed at a fairly broad audience, most are aimed at computer science majors (or perhaps both computer science and mathematics majors.) The typical course described is a one semester course meeting three hours a week. (Seventy per cent of the respondents reported a one semester course, ten per cent reported a two semester course and 16 percent reported a one quarter course, four percent reported a two quarter course.) These courses are likely no more sophisticated or proof-oriented in their approach than freshman calculus, and at least somewhat algorithmic in flavor.

Projected content of discrete math courses. We asked respondents to choose from a menu of topics chosen to typify the contents of current books in discrete mathematics, applied algebra, combinatorics and probability. We also asked respondents to indicate which topics on the menu should <u>not</u> be included in such a course. We analyzed the results separately for courses taught in mathematics departments, courses taught in computer science departments and courses taught in joint departments. With the exception of probability topics, there was fairly general agreement as to what would be taught regardless of where it would be taught. The overall results are summarized in Figure 1. This Figure shows more detail than was possible in our earlier report. The topics on the menu correspond (as shown) to the columns of the Figure; the percentage of respondents favoring a topic is graphed in that column with a "/" and the difference between the percentage favoring and the percentage excluding a certain topic is graphed with a "\". Both graphs have fairly evident points of inflection; these points show where the degree of consensus on what

Figure 1. Summary of topics chosen by respondents



should be in a course is changing most rapidly. A plausible generalization is that the topics in the "left hand group" are likely to appear in the majority of discrete mathematics courses while the topics on the right may appear in a significant minority of discrete mathematics courses. If a topic was excluded by significantly more respondents than included it (more than ten per cent), then the column for that topic was not labelled with the topic. The topics not included were topics in abstract algebra, abstract linear algebra, further probability, the general theory of difference equations and combinatorial designs.

It is possible to organize the topics shown in Figure 1 into natural groupings according to how they might appear now in various courses. They are organized in this way below.

Sets, relations, functions, equivalence and ordering relations, multisets

Permutations, combinations, partitions, recurrence relations, inclusion-exclusion, generating functions and difference equations

Graphs, digraphs and trees

Induction and recursion

Truth tables, propositions, Boolean algebra, predicate logic

Probability, expected value, random variables, binomial probabilities, standard deviation

Matrix algebra

Comparison with the MAA Discrete Math Panel course. For the sake of comparison we list here the major topics but not the detailed subtopics of the one year course outlined in the preliminary report of the Panel on Discrete Mathematics of the Mathematical Association of America [2].

1. Sets	5. Functions and Relations	9. Trees and order
2. The number system	6. Recursion	10. Algebraic Structures
3. The nature of proof	7. Combinatorics	11. Algorithmic Linear Algebra
4. Formal logic	8. Graphs	

The overlap among the outline proposed by the panel, the outlines proposed to the panel by correspondents (and included as appendices in their report), and the survey responses indicates a growing consensus on the content of discrete mathematics courses. (Although the panel had access to details of our survey they considered a great deal of other data as well.) We comment below on the differences between the survey results and the panel outline.

First we did not include topics in number systems in our menu; this decision means we cannot comment on the percent of our sample which would include a unit on the number system.. From the answers to our open ended questions we have no evidence that respondents thought we should have included such topics. Despite the fact that most of this material is supposed to be covered in high school algebra, American students are often deficient in their understanding of number systems. Because the panel is recommending a one year freshman course (which means that weak students might have the time to take college algebra first) I believe the panel's recommendations in this area are realistic. In our survey results I find considerable sentiment for discussing the nature of proof in an integrated fashion rather than as a separate unit. Still it appears to me that most of the Panel's recommendations for Unit 4 will appear somewhere in a discrete mathematics course. The majority of the material in the algorithmic linear algebra unit is matrix algebra: the survey leads us to conclude about half of the courses will have some of this material. The survey did not ask about linear programming, so we can make no comment on this topic. One point of significant difference between the panel outline and the survey results is in the unit on algebraic structures. Our survey suggeests that Boolean algebra is the only topic in that unit likely to appear in a significant number of courses.

There is one other area where the Panel's outline and the survey results do not agree, namely probability. There was also a clear difference between courses taught by mathematics departments and courses taught by computer science departments in the depth of coverage of probability. This is also the one area where I hope the panel will reconsider its recommendations before its final report. The Panel recommends probability as a topic in the combinatorics unit while I argue on the basis of both the survey results and the needs of computer science students that a separate unit on probability is quite important. The topic of expected value was included by more than half of our respondents. It is natural to assume that this is because of the importance to computer science students of being able to analyze the expected run time of algorithms. One may argue that some institutions will require separate courses in probability and statistics of their computer science majors and that these courses will cover expected values. Many institutions, mine included, will not, however

Further the usual calculus-based probability course treats continuous random variables rather than discrete random variables as its main topic, so the tools needed to analyze expected run time of algorithms will be touched on indirectly if at all. Finally, the time for a student to understand the probabilistic background for an analysis of quick-sort or tree-sort is before they come up in computer science courses, not after!

A one semester course. The only other major divergence between the panel recommendations and the survey results is the duration of the course. Our survey data indicate that the typical course will be a one semester course rather than a two semester course. The remainder of this paper is devoted to giving an outline for a one semester course. Since the outline below is in close agreement with with the MAA panel recommendations it is natural to ask "How is the course to be squeezed into one semester? Is the instructor to talk twice as fast? Are the students supposed to think twice as fast?" I have four answers to these questions. First, I believe that the course could quickly evolve to a four semester hour course as calculus courses have. Second, by requiring only one semester of discrete mathematics, a department opens up one additional semester in which it may require students to take a course in abstract algebra, combinatorics, graph theory, linear algebra and linear programming, logic, or probability and statistics. Thus it can leave one or two units in the outline to one of these courses. Third, this course outline is based on the assumption that the student has mastered high school algebra or will take college algebra before discrete mathematics. In fact half the institutions surveyed (Dartmouth included) have or intend one or two courses in calculus as a prerequisite. This experience with a substantial college course will accustom the student to a faster pace and develop a student's manipulative skills so less time need be spent on them in discrete mathematics. Finally, a one semester course aimed at students who are unsophisticated in high school (or college) algebra would almost surely not cover units 5, 9, and 10. Especially for such students the panel's recommended two semester sequence is the approach I would recommend, though an alternate approach would be a substantial college algebra course followed by a semester of discrete mathematics.

In the course outlined below, I have marked with an asterisk those topics an institution might choose to leave to later courses. Of course leaving all these topics to later courses is undesirable and probably unfeasible.

Outline for a One Semester Discrete Mathematics Course

(Topics marked with (*) are included as time and local circumstances dictate)

Unit 1 **Sets and Logic** Sets as truth sets of statements, logical connectives and set operations, circuits to test the truth of statements, conditional statements. Equivalence and implication and their relation to truth sets. Equivalence of a statement and its contrapositive as the basis of an indirect proof.

Unit 2 **Functions and Relations** Relations and digraphs, transitivity and reachability, transitive closure, partial orderings. Symmetric relations and graphs, connectivity and equivalence relations. Functions, one to one and onto functions, review of logartithmic and exponential functions, "big OH" notation.

Unit 3 **Mathematical Induction** The principle of mathematical induction. Examples of divide and conquer algorithms and inductive proofs that they work. Inductive (recursive) definition of functions. (Recursive definition of sets and applications to context free languages)*

Unit 4 **Basic Combinatorics** The sum and product principles, permutations as one to one functions, combinations as subsets and multisets. Pascal's triangle and the binomial theorem. (The principle of inclusion and exclusion)*

Unit 5 **Advanced Combinatorial Analysis** Recurrence relations, first order linear recurrence relations (constant coefficient case), reduction of recurrences from divide and conquer algorithms to first order linear. (Second order linear homogeneous recurrence relations and Fibonacci style problems)* (Generating functions, product principle for generating functions, the extended binomial theorem, application to second order recurrence relations)*

Unit 6 **Trees** Trees as connected acyclic graphs, spanning trees. Rooted and Binary trees. Binary trees as data structures, tree traversal. (Breadth and depth first search trees)* (Minimum total cost and minimum total path length spanning trees)*

Unit 7 **Graphs*** Multigraphs, isomorphism, polynomial time verification algorithms, the travelling salesman and Chinese postman problems.

Eulerian and Hamiltonian graphs, mention of concepts of NP hard and NP complete. Colorability, planarity, Euler's formula. Digraphs as models of finite state machines.

Unit 8 **Probability** Sample spaces, probability measures, conditional probabilities and independent events, expected values, (binomial probabilities)* (Standard deviation and its interpretations)*

Unit 9 **Matrix Algebra*** Matrix operations, matrix equations and systems of linear equations, inverse matrices, determinants, applications of matrix powers to graphs and Markov Chains, Hamming codes.

Unit 10 **Symbolic Logic*** The language of the propositional calculus, truth assignments and satisfiability, Boolean algebra, conjunctive and disjunctive normal form, Boolean algebras as lattices, unique decomposition into atomic elements. The language of the predicate calculus, quantifiers, prenex form, database query languages. The ideas of inference and their importance in artificial intelligence.

The Experience at Dartmouth. The discrete mathematics course at Dartmouth College is a course for students in their first year of university level mathematics. The course has one semester of calculus as a prerequisite and is a prerequisite to both the computer science and the mathematics majors. Virtually all the advanced computer science courses have it as a prerequisite. The majority of the advanced mathematics courses outside analysis have it as a prerequisite. In twenty four lectures of 65 minutes duration the course covers Units 1-4, half each of Units 5 and 6, and most of Unit 8 and Unit 9. In addition there are 6 hours per week of tutorials available to the students. In as many as four more lectures of 65 minutes duration the students are introduced to BASIC programming with conditional instructions, loops, strings, recursive programming and rudiments of file handling. Additional computer science topics such as binary trees, binary search and sorting algorithms are covered in the main lectures. Weekly programming assignments illustrate the computer science topics and applications of computing in probability and matrix algebra. The main themes of the course are mathematical induction, algorithm analysis, systemmatic enumeration, and graphs and relations.

Based on student performance on computer assignments and examinations, I believe most students are mastering the material. (Of course some master it at a much higher level of sophistication than others.) On the other hand I also believe the students find this course just as challenging as calculus.

Conclusions. There is, at least among institutions in the United States, a reasonably well defined body of material that is important to introduce early in a computer science student's carreer. This is leading to a discrete mathematics survey course taught to students in their freshman or sophomore year. Although the content of this course is reasonably well established, it will vary on the basis of local needs. There is not a general consensus in the community on the status of algebraic structures and probability in such a course, or on the duration of and prerequisites to such a course.

References

1. Bogart, K.P., Cordiero, K. and Walsh, M.L. "What is a Discrete Mathematics Course?" SIAM News, January, 1985.

2. Panel on Discrete Mathematics. "Preliminary Report" American Mathematical Association, November, 1984

Introductory Calculus in 1990

a paper for the I.C.M.I. Symposium on Computers and Mathematics*

B.R. Hodgson (Université Laval)
E.R. Muller (Brock University)
J. Poland (Carleton University)
P.D. Taylor (Queen's University)

1) Introduction

In this article we propose ways in which the introductory Calculus curriculum might respond to the recent and rapidly changing computer resources. Our purpose is not to describe how such computer resources might be used most effectively in the learning of the Calculus but rather to examine the impact of the existence of such resources as computer programs to perform differentiation and definite and indefinite integration.

Our main points are

- it is counterproductive to train our students to perform calculations that they know a microcomputer can do far more accurately and quickly;

- consequently a major reorientation in the style and content of the introductory Calculus course is needed, away from the performance of algorithms and towards a more meaningful and thoughtful experience;

- the spirit of this change calls for presenting the Calculus as one of mankind's finest intellectual achievements, more valuable than ever in its recent applications, and demanding of more interactive classroom teaching.

In a sense, we are entering a golden age of mathematics teaching, in which the deemphasis upon paper-and-pen performance of algorithms frees us to teach in ways that respect what we each feel are the true goals of mathematics education.

*The authors would like to thank SSHRC of Canada for a grant to hold the 1984 Annual Meeting of the Canadian Mathematics Education Study Group where this paper was prepared, and the hospitality of the University of Waterloo, site of the Meeting.

2) The computer resources

Always in the background will be the ongoing convergence in technology of the handheld calculator and the microcomputer that may well lead to a microcomputer the size of a pencil case, folding open to display a low-power, high-resolution graphics screen above and keyboard below, possibly with portable-telephone capabilities to hook into the classroom microcomputer or to home. The demand for such technology is very widely based in society and could well result in such developments. In terms of software, the MACSYMA computing package now incorporates the ability to find derivatives of a function given in closed form, and through the work of R. Risch [11] to also find its indefinite integral where this exists in closed form or to indicate that no such closed form exists. Some of this is presently available in software for present microcomputers (example: MuMath [12]) and more extensively for microcomputers of the immediate future (example: Maple [2]). Many similar software developments are underway, incorporating symbolic manipulation systems such as factoring polynomials, and offering quick, accurate graphing of functions [9].

3) The Calculus: intellectual challenge and routine manipulation

Although many individuals have questioned the central position which the Calculus enjoys in post-secondary education, quite recently M.E. Rayner at the session "Is Calculus Essential?" at I.C.M.E. IV [10] underlined the fact that the Calculus provides a rich source of concepts which are part of mathematics and mankind's cultural history. The problems originally addressed by the Calculus, and more recent applications, are to situations that are challenging and varied and require great intellectual skill to solve. The Calculus is particularly rich in ideas to motivate and direct students to think in a mathematical way, because it is here that the unreasonable effectiveness of mathematics is so clear. On the other hand, the Calculus as its name suggests and its long and rich history fully illustrates, was invented to allow us to handle the calculation of such entities as enclosed area and instantaneous velocity through the application of simple algorithms. But the invention of computers calls into question the necessity of training our students to be capable primarily of using and viewing the calculus as a method to calculate. The various techniques of integration take on an entirely different perspective, we argue, when such routines as MACSYMA are readily available (see Steen [12] for similar remarks). For, make no mistake here, the vast majority of our students after graduating will use these routines whenever faced with any problem involving the calculus as a tool. No more hunting through a table of integrals, the engineer, scientist or social scientist will use some prepackaged variant of Risch's algorithm on a computer for the indefinite integral, or a variant of Simpson's Rule to calculate the definite integral without even finding the indefinite integral. Unfortunately many introductory Calculus courses only pay lip service to the most important, the mathematical intellectual pursuit, and spend most of the time on calculations which are easier to use as testing materials and the assignment of grades.

4) The pressure for change

In the near future, we can expect the students arriving at our introductory Calculus courses to have substantial experience with microcomputers and software. The ability of microcomputers to quickly calculate values, plot points and thereby sketch curves will lead, undoubtedly to our students having a greatly increased ability to visualize the graph of a function. In particular we would expect them to be familiar with using the computer and calculators to solve problems, to have had access to many types of microcomputers, and to know that there exists a wide range of inexpensive mathematics software. The knowledge of the existence and widespread availability of microcomputer packages to do differentiation and integration will quickly spread through our classroom. If we continue to base our teaching and student evaluation primarily on the paper-and-pen execution of the algorithms of the Calculus, then we and our textbooks run the risk of our students perceiving mathematics as an archaic branch of science superceded by the new, challenging and prestigious frontier of computing (see [5]).

Just now a similar revolution is taking place in the teaching of arithmetic (see Anderson [1], or [8]). Handheld calculators have been accepted in many arithmetic classes, where the teaching emphasizes learning what can be done quickly and mentally and deemphasizes pen-and-paper execution of the algorithms. Also the skills of using basic number facts to make estimates for the reasonableness of an answer are more important today. Students in arithmetic are intended to have as great a facility with simple mental calculations and an intuition for arithmetic operations as they ever have in the past. Is this being accomplished? Can it be accomplished in the calculus? Certainly our students will expect similar adjustments in the style and content of our calculus courses to those they have seen in arithmetic.

5) Identifying essential aspects of the Calculus curriculum

It should be clear that much of the calculation aspect of the curriculum can now be omitted. How should the curriculum respond?

Clearly the notion of limit remains an important conceptual tool, and we should explain the derivation of various differentiation rules. But our task is no longer to teach them how to differentiate but rather when, where and why. In consequence, our new textbook and our lectures will concentrate on the minimum number of concepts and derivations to cover the meaning of differentiation and impart the necessary ability to mentally deal with simples cases and check the reasonableness of more complicated ones, but all this imbedded in a different context, as we shall explain in the next section. Similarly, some techniques of integration will be presented, when appropriate, for their conceptual value. For example, substitution techniques are fundamental to further study and will remain in the introductory curriculum we feel, whereas integration by parts and use of partial fractions need only be taught in later courses, for example, in the context of complex integration, the basis for the computer's indefinite integration routine. The ability of microcomputers to quickly sketch curves greatly reduces the necessity of teaching the detailed graph-sketching techniques in present introductory calculus courses, and the basic underlying theory can be taught again as a conceptual tool, pathological examples not withstanding.

6) How should the free time be spent?

It has been estimated (Wilf [16]) that about 15 to 20% of classroom time will become available by deletions similar to those we have outlined. We are strongly of the opinion that this should not be filled with additional topics but rather used to give greater depth and to give the student a better and more intimate or first hand feeling for the nature and beauty of the calculus, and a real appreciation of its power for solving problems in today's world. We suggest three ways in which the traditional curriculum might be enhanced: (i) the use of a contextual approach, (ii) the use of calculus as a tool in the qualitative analysis of functions, especially in mathematical modelling, and (iii) the use of an open, interactive, and exploratory mode of classroom teaching. In all cases, these approaches can be reinforced by computer-enhanced classroom calculus teaching, in which the numerical and graphic capabilities of the computer can be used to explore new problems and painlessly collect empirical data. We now briefly explore each of these proposals.

(i) A contextual approach

Let us show them some of the power of mathematics and the power of the human mind. We can examine the cultural context in which this arose and how its solution in turn had a cultural aftermath (Edwards [3], Kline [7], and Toeplitz [14]). As we have already remarked, there is at present a great emphasis on the technical aspects at the expense of the intellectual. What are the sources of the concepts and results of the Calculus, and why would anyone have bothered to think about such things? These are fascinating questions, and an effective way of motivating students, by placing the Calculus in its context. The cultural aftermath and applications of the Calculus from the 17th century through to today illustrate most aptly the resulting unreasonable effectiveness of mathematics.

(ii) The qualitative analysis of functions in mathematical modelling

The computer is good at working with specific functions, or even with general functions of a specific algebraic form. But often in the mathematical models which are becoming an increasingly important part of the biological, social, and medical sciences, all we know, or wish to postulate, about our input functions, is their general geometric form. All we may wish to assume is that a given relationship is S-shaped or of diminishing returns in a certain interval, and we want to draw conclusions about the existence of certain optima, or the existence and stability of certain equilibria. Computers cannot do that. They can work with a quadratic, logistic, or exponential candidate and give us an idea of what to expect, but the results of real use (and power or beauty or simplicity) are only obtained with methods of analysis, often with considerable ingenuity. The calculus is an indispensible tool in such analysis, and the curriculum of the future must be rich in such examples of its use. Some useful references here are Hilton [4], Chapter IV in [6], Taylor [13] and Chapter V in [15].

(iii) An interactive mode of classroom teaching

It is often argued that the most effective teaching format is interactive, in which the students participate with the teacher in an open-ended problem solving exercise. Certainly it is a more interesting and enjoyable format than the traditional lecture, for both student and teacher, though it certainly places greater demands on the teacher. The method is widely used in the humanities and social sciences, but less common in the natural and physical sciences. We have no doubt that the reason for this is that curricula in the latter disciplines are based on the covering of a large number of highly specific units of material, and the lecture method affords the teacher greater control and allows him to cover ground more quickly. But we are of the opinion that often enjoyment, appreciation and depth of understanding are sacrificed in the lecture format. In short, effectiveness is sacrificed to efficiency. A course which managed to eliminate 10 to 15% of its material might be a good place to start experimenting with the use of more interactive classroom time. It would be useful to have a text book which included exercises and material designed for this type of classroom use. Taylor [13] contains a number of examples of this type.

The greater use of a problem-solving format will better prepare the student for mathematical life after the end of his or her Calculus course, and again provide a more meaningful setting for learning. Its meaningfulness can be enhanced by beginning such a problem, with student participation, in the class, before the necessary tools have all been developed in the course. Students can then have time between classes to think and write about avenues of solution, meet again and through interaction and feedback at this stage learn to evaluate their contributions, and develop the necessary tools or the readiness to appreciate such tools when the teacher introduces them. Again this offers opportunities to return to historical and cultural points that reinforce the students' web of understanding and meaning. Examinations will be reduced in emphasis and replaced by more project work.

7) Conclusion

Why train our students to perform calculations that a microcomputer can do more accurately and quickly? We have argued that both the style and content of the introductory Calculus course (including its textbooks and evaluation procedures) must change, not only to remove redundant material and give a richer view of the Calculus including new aspects which belong preeminently in our modern computer-based world, but because otherwise a generation of young students will dismiss mathematics and never come to understand the crucial role it has and continues to have in solving problems. The computer can be seen then as a tool to strengthen basic concepts of mathematics and as a tool to which mathematics has much to offer, rather than surplanting mathematics; and mathematics can be taught as an effective, enjoyable and unique way of thinking.

Bernard R. Hodgson
Département de mathématiques, statistique et actuariat
Université Laval
Québec, G1K 7P4

Eric R. Muller
Department of Mathematics
Brock University
St. Catherines, L2S 3A1

John C. Poland
Department of Mathematics
and Statistics
Carleton University
Ottawa, Ontario, K1S 5B6

Peter D. Taylor
Department of Mathematics
and Statistics
Queen's University
Kingston, K7L 3N6

CANADA

Bibliography

[1] D. Anderson, "Arithmetic in the computer/calculator age", in The Future of College Mathematics, ed. A. Ralston and G.S. Young, Springer-Vrelag, New York, 1983; pp. 235-242.

[2] B.W. Char et al., "An Introduction to Maple", University of Waterloo Research Report C58404, Waterloo, Ontario, 1984.

[3] C.H. Edwards Jr., The Historical Development of the Calculus, Springer-Verlag, New York, 1983.

[4] P.J. Hilton, ed., The Role of Applications in the Undergraduate Mathematics Curriculum, Report of the Assembly of Mathematical and Physical Sciences, National Research Council, Washington, 1979.

[5] B.R. Hodgson and John Poland, "Revamping the mathematics curriculum: the influence of computers", Notes of the Canadian Mathematical Society 15 (November 1983), pp. 17-23.

[6] J.T. Fey, ed., Computing and Mathematics, National Council of Teachers of Mathematics, Maryland, 1984.

[7] Morris Kline, Mathematics in Western Culture, Oxford University Press, New York, 1953.

[8] National Science Board Commission of Precollege Education in Mathematics, Science and Technology, The Mathematical Sciences Curriculum K-12: What is still fundamental and what is not. National Science Foundation, Washington, 1982.

[9] R. Pavelle et al., "Computer algebra" Scientific American 245 (Dec. 1981), pp. 136-152.

[10] M.E. Raynor, "Is Calculus essential?", Proceedings of the Fourth International Congress on Mathematical Education, 1984; pp. 50-52.

[11] R.R. Risch, "The solution of the problem of integration in finite terms", Bull. Amer. Math. Soc. 76, 1970, pp. 605-608.

[12] L.A. Steen, "Computer calculus", Science News 119 (April 18, 1981).

[13] P.D. Taylor, Calculus and the Analysis of Functions. Department of Mathematics and Statistics, Queen's University, Canada, 1978.

[14] Otto Toeplitz, The Calculus: a genetic approach, University of Chicago Press, Chicago, 1963.

[15] A.C. Tucker, ed., Recommendations for a General Mathematical Sciences Program. Report of the Committee on the Undergraduate Program in Mathematics, the Mathematical Association of America, Washington, 1981.

[16] H.S. Wilf, "Symbolic manipulation and algorithms in the curriculum of the first two years", in The Future of College Mathematics, ed. A. Ralston and G.S. Young, Springer-Verlag, New York, 1983; pp. 235-242.

Computers in the beginner's course of the calculus

Osamu Takenouchi
Faculty of Engineering Science
Osaka University

In the presense of computers, some change must be made for a better mathematics education. A good change can be expected from the very early stage of mathematics education, but here we will discuss the change that may take place in the teaching of calculus in the beginner's course in the college.

The author assumes that the main line, the main parts of the teaching of mathematics are not going to be altered very much in a decade or so from now. But, what is required for the achievement will be changed. At one time in the past, the requirement for the skillfulness in the calculation was dominant. Now that the computers do complicated computations rather easily, and the scope of mathematics used in the applications is wide spread, the conceptual side of mathematics is becoming to be given much weight in the teaching.

In the calculus, we have been teaching the methods of differentiations and integrations, and, as applications, the methods of getting maxima and minima, areas or volumes, etc. But, actual calcualtions of these will be done now by computers, and we have only to teach some of the fundamentals and the mechanisms for these kinds. In compensation with this, it is required that the students should get a good and deep insight into these matters.

Hitherto, some items have not been well treated. In some cases, the materials will get too complicated if one wants to make a general development; while in other cases, interesting features can be got only after a laborious computations. In this respect, computers are very useful tools in finding atracting phenomena, and the students will be fascinated in the mathematics, encountering with wonderful events.

The author will take this point of view, and will give some instances which he thinks to exert a good effect on this purpose.

1. Creating functions

A function $y = f(x)$ is, by definition, a correspondence which to each value of x associates a definite value y. However, functions which appear in the course of the teaching are generally given in an abstract, or a logical way. The student cannot know how to find the value of a given function except for rational functions. Radicals, exponential, logarithmic and trigonometrical functions are fundamental in the calcules, yet the student doesn't know the way to obtain their values. For the most part of the students, this must give some ambiguous impression which may be the cause of various difficulties.

2. Sequences, series

To discuss sequences and series is a very fascinating matter. While it lies in the beginning of the calculus, deep knowledge of the calculus is sometimes necessary to discuss them clearer. And, we may even say that many problems to be dealt with in mathematics and in computations concern with sequences and series. This is an important subject which is very useful for the students to get look at the profundity of mathematics. But, as the knowledge of various stages is required, we must be careful to choose good chances to enter in the discussions of these problems. Good preparations will yield a deep impression in the student's mind.

The main features of sequences and series to be discussed are:

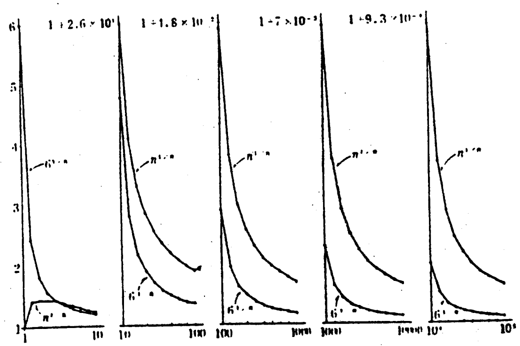
a. Order of growth, rapidity of convergence

b. Behaviour of sequences. Monotonity, zigzag type

Here, the author will discuss these problems by examples.

Ex. 1. $a_n = \sqrt[n]{a}$ $(a > 1)$, $b_n = \sqrt[n]{n}$

Both of them converges to 1, decrecasingly (b_n for $n \geqq 3$). We can compare the mode of convergence of these two sequences, by plotting the point on the graphic display (fig. 1).



Ex. 2. Fibonacci sequence a_n

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ··· ···

We know

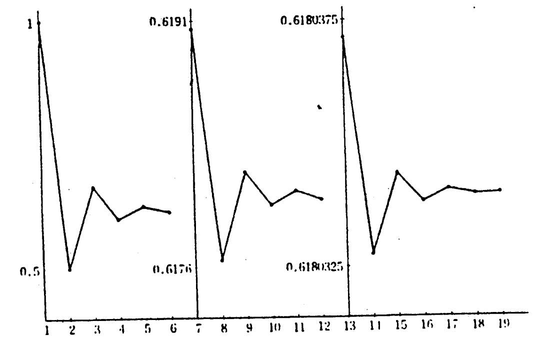
$$a_n = \frac{1}{\sqrt{5}}\left\{\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right\}$$

So the order of growth is about $\frac{1+\sqrt{5}}{2} = 1.6$ This may be checked experimentally by producing the numbers in the computers, and we make students to find some way for finding the order of growth (that it is between 1.5 and 2 is easily derived).

Putting

$$b_n = \frac{a_n}{a_{n+1}} ,$$

b_n converges to the golden ratio $\frac{\sqrt{5}-1}{2}$. The mode of convergence is typically zigzag. We can follow the manner of convergence on the graphic display (fig.2).



Ex. 3. $a_n = (1 + \frac{1}{n})^n$

This important sequence is frequently taken up in the use of computing mechanism. But, this sequence is rather squeamish, and careful treatment is necessary.

The table shown below is a result of computation using three different kinds of handheld calculaters. Observe that different apparatus yield different values, and the difference is sometimes remarkable.

1	2	3	4	5	6	7	8	9
2.000000000	2.250000000	2.370370370	2.441406250	2.488320000	2.521626372	2.546499697	2.565784514	2.581174792
	2.250000000	2.370370370	2.441406250	2.488320000	2.521626371	2.549499696	2.565784514	2.581174791
	2.250000000	2.370370370	2.441409250	2.488320000	2.521626372	2.546499697	2.565784514	2.581174792
10	20	30	40	50	60	70	80	90
2.593742460	2.653297705	2.674318776	2.685063839	2.691588029	2.695970140	2.699116372	2.701484941	2.703332461
2.593742460	2.653297705	2.674318773	2.685063838	2.691588029	2.695970129	2.699116355	2.701484941	2.703332458
2.593742460	2.653297705	2.674318776	2.685063838	2.691588029	2.695970138	2.699116370	2.701484941	2.703332461
100	200	300	400	500	600	700	800	900
2.704813830	2.711517124	2.713765155	2.714891744	2.715568522	2.716020056	2.716342744	2.716584845	2.716773206
2.704813829	2.711517123	2.713765131	2.714891744	2.715568521	2.716019940	2.716342683	2.716584847	2.716773181
2.704813829	2.711517123	2.713765155	2.714891744	2.715568521	2.716020038	2.716342721	2.716584847	2.716773206
1000	2000	3000	4000	5000	6000	7000	8000	9000
2.716923931	2.717602568	2.717828885	2.717942118	2.718010039	2.718055413	2.718087634	2.718112020	2.718130883
2.716932932	2.717602569	2.717828648	2.717942121	2.718010050	2.718054753	2.718086876	2.718111955	2.718130556
2.716923932	2.717602569	2.717828893	2.717942121	2.718010050	2.718055731	2.718087637	2.718111955	2.718130801
10000	20000	30000	40000	50000	60000	70000	80000	90000
2.718145918	2.718213329	2.718235890	2.718247035	2.718254646	2.718257908	2.718260082	2.718264432	2.718264432
2.718145927	2.718213875	2.718233807	2.718247851	2.718254646	2.718248304	2.718259694	2.718264839	2.718264009
2.718145927	2.718213875	2.718236253	2.718247851	2.718254646	2.718258089	2.718261597	2.718264839	2.718266455
100000	200000	300000	400000	500000	600000	700000	800000	900000
2.718265519	2.718270955	2.718270955	2.718270955	2.718268237	2.718270955	2.718260082	2.718281828	2.718254646
2.718268237	2.718275033	2.718250115	2.718278431	2.718279110	2.718170834	2.718143976	2.718280130	2.718253136
2.718268237	2.718275033	2.718274580	2.718278431	2.718279110	2.718268690	2.718277169	2.718280130	2.718277600
1000000	2000000	3000000	4000000	5000000	6000000	7000000	8000000	9000000
2.718281828	2.718281828	2.178173099	2.718281828	2.718281828	2.718173099	2.718336195	2.718281828	2.718010014
2.718280469	2.718281149	2.718009561	2.718281489	2.718281557	2.717194507	2.717194539	2.718281659	2.718009853
2.718280469	2.718281149	2.718254193	2.718281489	2.718281557	2.718172873	2.718145724	2.718281659	2.718254495

Looking at this table, we are led to discuss:

(1) Which value is correct?

(2) How can we get the correct answer?

(3) Why the different apparatus yield so big a difference?

From mathematical point of view, as a_n is an increasing sequence, we cannot decide very easily how close the calculated value is to the limiting value, contrary to the case of zigzag type.

We take first an auxiliary sequence which approaches decreasingly to the limiting value.

$$b_n = \left(1 + \frac{1}{n}\right)^{n+1} = \left(1 - \frac{1}{n+1}\right)^{-n-1}$$

As the limit e is inbetween a_n and b_n, the difference of a_n from e is roughly estimated as the half of $b_n - a_n$, or $\frac{1}{2n}$ e.

Correct estimate is

$$\left(1 + \frac{1}{n}\right)^n = e\left\{1 - \frac{1}{2}\frac{1}{n} + \frac{11}{24}\frac{1}{n^2} - \frac{7}{16}\frac{1}{n^3}\right\}$$

when $\frac{1}{n^4}$ can be disregarded

This formula can be derived by some elementary calculations starting from the binomial expansions. It can be derived also from the power series exapnsion of $\left(1 + \frac{1}{n}\right)^n = \exp n \log \left(1 + \frac{1}{n}\right)$

We can see from the form of this expansion that the acceleration method can be applied to get a better answer.

Put

$$c_n = 2\,a_{2n} - a_n$$
$$d_n = \frac{1}{3}\left(4\,c_{2n} - c_n\right)$$

Then, we are able to make following computation.

n	a_n	c_n	d_n
1000	2.716923932	2.718281206	2.718281828
2000	2.717602569	2.718281673	
4000	2.717942121		

Ex. 4. Leibniz series

$$1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \cdots \cdots$$

This famous series have $\frac{\pi}{4}$ as its sum. Let a_n be the partial sum of this series. Then a_n is a sequence of zigzag type, but the mode of concvergence is very slow. We here give a result of computation. The values are multiplied by 4 to get an easier comparison with π.

1	4.0000000000						
2	2.6666666667	20	3.0916238067	200	3.1365926848	2000	3.1410926536
3	3.4666666667	30	3.1082685667	300	3.1382593295	3000	3.1412593203
4	2.8952380952	40	3.1165965568	400	3.1390926575	4000	3.1413426536
5	3.3396825397	50	3.1215946526	500	3.1395926556	5000	3.1413926536
6	2.9760461760	60	3.1249271439	600	3.1399259881	6000	3.1414259869
7	3.2837384837	70	3.1273076680	700	3.1401640829	7000	3.1414497964
8	3.0170718171	80	3.1290931418	800	3.1403426541	8000	3.1414676536
9	3.2523659347	90	3.1304818854	900	3.1404815428	9000	3.1414815425
10	3.0418396189	100	3.1315929036	1000	3.1405926538	10000	3.1414926536

The remarkable phenomena seen from this table can be explained by

$$\left| a_n + \frac{1}{4n} - \frac{\pi}{4} \right| < \frac{1}{8n^3} \qquad \text{for } n\text{: even}$$

which is obtained by estimating

$$\int_0^1 \frac{1 - t^{2n}}{1 + t^2}\,dt + \frac{1}{2}\int_0^1 t^{2n-1}\,dt - \int_0^1 \frac{1}{1 + t^2}\,dt$$

Ex. 5. Euler constant

The series

$$1 + \frac{1}{2} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots\cdots$$

diverges to infinity. But its order of growth is very slow. Let a_n be the partial sum of this series. a_n is growing to infinity quite slowly yet steadily.

As we know, putting

$$b_n = 1 + \frac{1}{2} + \frac{1}{3} + \cdots\cdots + \frac{1}{n} - \log n,$$

b_n decreases and converges to a limiting value γ, Euler constant. So a_n is nearly $\log n + \gamma$, which is about 7.5 when $n = 1000$, and about 10 when $n = 10000$.

About the value of γ, we are not very sure how good approximation is obtained by only calculating b_n. To this end, an asymptotic expansion is known.

$$b_n = \gamma + \frac{1}{2n} - \frac{B_1}{2n^2} + \frac{B_2}{4n^4} - \frac{B_6}{6n^6} + \cdots$$

But the way to derive this formula, and the meaning of asymptotic expasions are not a matter for the beginner's course.

However, the first terms can be obtained elementarily by the repetition of integration by parts, which we will see below.

Put $b_n = a_n - a_{n+1}$. Then, we have

$$a_n - \gamma = b_n + b_{n+1} + \cdots\cdots$$

Now,

$$\begin{aligned}
b_n &= \log\frac{n+1}{n} - \frac{1}{n+1} \\
&= \int_0^1 \frac{1}{n+1-t}\,dt - \frac{1}{n+1} \\
&= \int_0^1 \frac{t}{(n+1)(n+1-t)}\,dt \\
&= \frac{1}{2}\frac{1}{n(n+1)} - \frac{1}{2}\int_0^1 \frac{t^2}{(n+1)(n+1-t)^2}\,dt
\end{aligned}$$

$$\begin{aligned}
\int_0^1 \frac{t^2}{(n+1)(n+1-t)^2}\,dt
&= \frac{1}{3}\frac{1}{(n+1)n^2} - \frac{2}{3}\int_0^1 \frac{t^3}{(n+1)(n+1-t)^3}\,dt \\
&= \frac{1}{3}\frac{1}{(n+1)n^2} - \frac{1}{6}\frac{1}{(n+1)n^3} + O\left(\frac{1}{n^5}\right) \\
&= \frac{1}{3}\frac{1}{n(n+1)(n+2)} + \frac{1}{3}\frac{2}{n^2(n+1)(n+2)} - \frac{1}{6}\frac{1}{n^3(n+1)} + O\left(\frac{1}{n^5}\right) \\
&= \frac{1}{3}\frac{1}{n(n+1)(n+2)} + \frac{1}{2}\frac{1}{n(n+1)(n+2)(n+3)} + O\left(\frac{1}{n^5}\right)
\end{aligned}$$

We then have

$$\begin{aligned}
b_n + b_{n+1} &+ \cdots\cdots \\
&= \frac{1}{6}\frac{1}{n(n+1)} + \frac{1}{6}\frac{1}{n(n+1)(n+2)} + O\left(\frac{1}{n^4}\right) \\
&= \frac{1}{6}\frac{1}{n^2} - \frac{1}{6}\frac{1}{n^2(n+1)} + \frac{1}{6}\frac{1}{n(n+1)(n+2)} + O\left(\frac{1}{n^4}\right) \\
&= \frac{1}{6}\frac{1}{n^2} + O\left(\frac{1}{n^4}\right)
\end{aligned}$$

So, finally we see

$$a_n - \gamma = \frac{1}{2n} - \frac{1}{12}\frac{1}{n^2} + 0(\frac{1}{n^4})$$

which may be applicable when $\frac{1}{n^4}$ is negligeable.

One observes also from this expression that acceleration method works as well.

3. Differentiation

(1) The meaning of differential coefficients.

The graphic display is very convenient. We can draw on it the graph of a given function. Taking a part adjacent to a point on the curve, and by enlarging it, we can observe that a curve looks like a straight line in a very small vicinity of a point. In this way, we can give a good understanding for the meaning of the differential coefficients and tangents.

(2) Successive differentiations and Taylor expansions.

Successive derivatives of a function are given only for some easy cases. But, for example, the function tan x is never treated, because it becomes too complicated to continue the differentiation process. In principle, there is no difficulty, or we may even say it is very easy. For $y = \tan x$, $y' = \sec^2 x = 1 + \tan^2 x = 1 + y^2$, and we continue the differentiation process. The result will be given as a polynomial of tan x which are shown below.

階数	1	$\tan x$	$\tan^2 x$	$\tan^3 x$	$\tan^4 x$	$\tan^5 x$	$\tan^6 x$	$\tan^7 x$	$\tan^8 x$	$\tan^9 x$	$\tan^{10} x$
0	0	1	—								
1	1	0	1	—							
2	0	2	0	2	—						
3	2	0	8	0	6	—					
4	0	16	0	40	0	24	—				
5	16	0	136	0	240	0	120	—			
6	0	272	0	1232	0	1680	0	720	—		
7	272	0	3968	0	12096	0	13440	0	5040	—	
8	0	7936	0	56320	0	129024	0	120960	0	40320	—
9	7936	0	176896	0	814080	0	1491840	0	1209600	0	362880

Getting the successive derivatives of tan x , we can now have the Taylor expansion of tan x .

The author considers this example important because, if we do not try to do this as it was so far, there may be a danger that the students consider in the following way.

Successive derivatives of sin x, cox x are taught, but not of tan x . Then tan x cannot have these derivatives, much less the Taylor expansion.

We have to exclude this kind of prejudice.

4. Integration

Computers are most often used for numerical integration. But our aim is not to give the method of numerical calculus. We use computers for a better understanding of the concepts of integration.

(1) Every continuous function can be integrated.

There are functions whose primitive functions are not elementary functions. This kind of functions are never treated in the teaching of integration. But, basically, there are no differences among

$$\int_0^1 \sqrt{1-x^2}\,dx, \quad \int_0^1 \sqrt{1-x^3}\,dx, \quad \int_0^1 \sqrt{1-x^4}\,dx.$$

We should treat them in the same manner.

(2) Improper integrals

In the computation of the improper integrals, there are many interesting, still elementary techniques are used.

5. Conclusion

Hitherto, the author discusses various examples. Of course, these examples are all known, but, to put them in practice, it was impossible without the aid of computers. They are very bewitching, and, once having got to know them, one is more interested in mathematics. The author believes that this is a way of using computers in the mathematics education for a better understanding of mathematics.

COMPUTERS EXPERIMENTS ON MATHEMATICAL ANALYSIS TEACHING AT THE POLITECNICO OF TORINO (ITALY)

Maria MASCARELLO RODINO
Anna Rosa SCARAFIOTTI ABETE

Dipartimento di Matematica - Politecnico di Torino

We would like to report on some experiments concerning the use of informatic tools in teaching basic mathematical courses at the Politecnico of Torino, Faculty of Engineering Sciences.

These experiments refer in particular to the courses of Mathematical Analysis 1° and Mathematical Analysis 2° adressed to the students of Mechanical Engineering in the years from 1980 to 1983, using pocket computers. This activity has been carried on in the years 1984 and 1985, in the same courses, using micro computers, such as Sharp MZ80B and IBM PC.

At this second stage, the experiment has concerned a restricted number of students, selected on the basis of a test.

As a general remark, we begin by observing that the traditional teaching of Mathematics in our Engineering Schools seems to be too abstract and far from the applications, for the most students. It is therefore necessary to introduce in the "curricula" some examples of actual construction of mathematical models which can be of some interest for students in Engineering.

It becames natural to use informatic tools whenever a "visualisation" is possible (as for the use of the microcomputers); moreover, it is possible to point out to the students in a clear way how the mathematics in the "discrete" may lead to the computational approach of the problems.

We would like now to detail some contents of this experience.

- Experience with pocket computers.

They have been used in the exercise-sessions of the course on Mathematical Analysis 1°. The main topics have been sequences and their applications. We have also considered the tabulation of some elementary functions; this leads on the computer to the formal proceeding:

$$x \longrightarrow f \longrightarrow f(x) .$$

That gives the opportunity for a "concrete" study of the domain and of the image of a given real valued function of a real variable.

To carry out the experience in a correct way, it has been necessary, of course, to rely on basic informatic arguments. To this aim, in the main course of the lectures, after giving some notions of the theory of formal languages, the teacher introduced machine-numbers and algorithms for floating-point arithmetic computations. Always in this direction and already in the first part of the course, some proofs of classical analysis results were presented in a computational form.

- Experience with micro computers.

The use of the micro computers in basic mathematical courses is particularly convenient whenever a "visualisation" is possible, as help for the learning proceeding.

In the exercise-sessions of the course of Mathematical Analysis 1°, for example, one studied on the micro computers the graph of a real valued function of one real variable, underlying local and global properties.

The most important experiment, however, has been for us the study of dynamical systems using micro computers.

On this subject we are preparing a report giving full details of our experience. In fact we have begun in the course of Mathematical Analysis 1° with the study of discrete dynamical systems, which was introduced after the study of sequences defined by recurrent formulae. As a natural continuation, in the course of Mathematical Analysis 2° we considered continuous dynamical systems, giving a formal expression of the qualitative results. Finally we returned to the use of micro computers to find numerical results; this was done in order to check the known results of the theory, and also to conjecture new results, concerning open problems.

Trying to draw conclusions, we first would like to say that, in our opinion, the results of this experience have been good for all students.

As in the aim of this experience, some arguments which were traditionally considered as "hard" and "too abstract" (as the convergence of a sequence, the sum of a series,...) have been made more attractive, and easier to teach and to learn. Moreover, we gave the basis for a numerical analysis of some problems which are not solved in a satisfactory way, by the qualitative theory, at least from the point of view of a student of Engineering.

Finally, the students are now prepared to face a given problem, by finding suitable mathematical methods to solve it, and by adapting the methods to a given informatic tool.

As a conclusion, we mention that the Politecnico of Torino begins this year a new didactic experiment, with the aim of giving to all students of the first year some basic informatic notions. This "laboratory of informatics" will not interfere with the contents of the courses, but it will give the basis for specific exercise-sessions basing on informatic tools. This new experiment may have a very interesting future for the teaching of the mathematics.

List of teachers involved in this experimentation:

Giuseppe Geymonat
Maria Mascarello Rodino
Luigi Montrucchio
Anna Rosa Scarafiotti Abete

List of treated topics:

- Sequences.
- Series.
- Integration theory.
- Ordinary differential equations :
    - i) discrete dynamical systems
    - ii) continuous dynamical systems.

REFERENCES.

1 V.I. Arnold, Equazioni differenziali ordinarie, Ed. MIR, Mosca, 1979.

2 P.Boieri-...-A.R. Scarafiotti, Personal computers in teaching basic mathematical courses, SEFI Annual Conference: The Impact of Information Technology on Engineering Education, Erlangen, 1984.

3 F.Cugno-L.Montrucchio, Some new techniques for modelling non-linear economic fluctuations: a brief survey, in Non-linear models of fluctuating growth, R.M.Goodwin-M.Krüger-A.Vercelli Edts, Lectures Notes in Economics and Mathematical Systems, 228, Springer-Verlag, N.Y., 1984.

4 B.P.Demidovič-I.A.Maron, Fondamenti di calcolo numerico, Ed. MIR, Mosca, 1981.

5 G.Geymonat, Lezioni di Matematica I, Per allievi ingegneri, Levrotto & Bella, Torino, 1981.

6 G.Geymonat, Congetture e dimostazioni: usi didattici dei calcolatori tascabili nell'insegnamento della matematica, La Fisica nella scuola, XV-4, Zanichelli, Bologna, 1982.

7 M.Hénon, A two-dimensional mapping with a strange attractor, Comm. Math. Physics, 50, 1976, 69-77.

8 P.J.Hilton-G.S.Young, New Directions in Applied Mathematics, Springer-Verlag, N.Y., 1982.

9 E.Isaacson-H.B.Keller, Analysis of Numerical Methods, J.Wiley & Sons, 1966.

10 M.Mascarello-A.R.Scarafiotti, Didattica e calcolatori: esperienze nei corsi di Analisi Matematica al Politecnico di Torino, Dipartimento di Matematica, Universita' di Torino, Seminar, February 1985.

11 M.Mascarello-A.R.Scarafiotti, Utilizzo di strutture informatiche di base nei corsi di Analisi Matematica del biennio al Politecnico di Torino, Dipartimento di Matematica, Politecnico di Torino, Mimeo, 1984.

12 F.Massobrio, Simulazioni di dinamiche caotiche mediante l'uso del computer, Mimeo, 1984.

13 A.R.Scarafiotti Abete, Teoria e pratica sui calcolatori pocket, Levrotto & Bella, Torino, 1980.

14 V.Smirnov, Cours de Mathématiques supérieurs, Ed. MIR Mosca, 1981.

15 A.Tikhonov-V.Arsénine, Méthodes de résolutions de problèmes mal posés, Ed. MIR, Mosca, 1976.

Bernard Winkelmann, IDM Bielefeld

THE IMPACT OF THE COMPUTER ON THE TEACHING OF ANALYSIS*

1. Starting Point

In this introduction, I shall treat the following topics: the fundamental role of computers in the mathematics teaching of Sekundarstufe II (ages 16 to 18), the present teaching approaches to integrating the computer into analysis instruction, and G. Richenhagen's ideas on the complementary role of discrete and continuous mathematics.

I consider the *importance of the computer in mathematics instruction* to be analogous to that of the hand-held calculator in the more arithmetically oriented mathematics instruction of junior classes. The latter has led, or will lead, to a profound change in educational goals and methods - reevaluation of concrete calculating skills, more emphasis on the ability to get a general picture and to apply known methods /1/. In the same way, the computer has a profound effect on the mathematics education of the senior classes with its more complex mathematical structures and methods. The following effects are apparent:

In the field of learning goals, a shift of emphasis towards higher abilities must be expected, as abilities which can be algorithmized as such are losing importance and value for the student. This is not only true for numeric-calculatory skills, but increasingly for symbolic-algebraic ones as well. In the field of teaching methods, the computer, if it has been loaded with the appropriate programs, will function as a de-technicizing aid, almost as a super hand-held calculator which permits the pupil to overcome the computational obstacles in the treatment of more complex problems and more realistic applications, e.g. in dealing with larger matrices, in the numerical solution of differential equations, or in the symbolic treatment of more complicated formulas; this will serve to widen the scope of mathematics education in terms of content. On the other hand, a computer equipped with the appropriate languages and environments can become an instrument for solving problems in the hands of the student (interactive programming); in this case, the student tends to understand techniques more on the cognitive level, and no longer mainly on the level of skill. Beyond that, the computer, with its possibilities of illustration and symbolization, will provide opportunities for providing more comprehensive and rapid mathematical experience.

This presents problems and tasks for educators mainly on two levels. On a more technical level, there is the necessity of providing more suitable software. On a more fundamental level, the problem is, besides determining trends, to achieve a balance in the quantitative and qualitative relation of new and old goals and methods. The following will give some specific examples and approaches for the field of analysis.

I have presented a survey of the most important present approaches to using computers in analysis instruction elsewhere /2/ and will not enlarge on this topic here. The following conclusion from this survey may be kept in mind:

* translated by Günther Seib, IDM Bielefeld. This is the English version of Occasional Paper No. 29.
/1/ For fundamental considerations of this matter see /Winkelmann 1978/

On the one hand, the computer creates new opportunities for analysis instruction, e.g.
- numerical and graphical illustrations,
- more complex and more realistic applications,
- a language in which to describe the traditional calculi,
- CAL (computer-aided learning) in its various forms.

On the other hand, some traditional motivations for treating conceptually exacting analysis in school can no longer be maintained without further discussion, for instance:
- calculations such as finding extreme values or areas can be easily programmed without analysis,
- practical applications, as in physics or technology, use discrete methods in computer programs.

This results in a crisis: the legitimacy of traditional analysis in school is challenged, educators will have to make clear to the general public, and the teacher will have to explain to his pupils asking critical questions, where treatment of continuous analysis does make sense nowadays.

The following sections will be devoted to this question.

G. Richenhagen has made an important contribution to dealing with the question by developing the complementarity between numerical and continuous aspects in analysis, and its deformation in school instruction /3/. His analyses, however, have not yet resulted in a constructive proposal on how we should determine in detail the desirable relation between these two components of teaching. The constructive proposal for a reorientation of analysis submitted in this paper is intended as a contribution to the present debate.

2. Characterization of Analysis in Application

I should like to confine this section to one aspect of the problem of "continuous versus discrete analysis". Hence, I shall deliberately leave aside considerations of history of science, theory of science, and philosophy (such as, for instance, about the "role and function of the infinite"), adopting instead the point of view of the user, that is of somebody who is interested in mathematics merely because he uses mathematical models, in particular, models containing analysis, to solve his (extra-mathematical) problems /4/. Although the role of applications, specifically those of analysis, has changed, both by the growing number of disciplines using corresponding models and by new methods, particularly the use of computers, an understanding of the fundamental approaches in which mathematizations take place remains indispensable. Examples of such concepts are:

/3/ cf. /Richenhagen 1983/
/4/ I have described this view of analysis in school elsewhere under the term of "Gebrauchsaspekt der Analysis", /cf. Winkelmann 1981/, pp. 59f, 61 and 75f.

- variable quantity, change
- functional connection
- local rate of change
- average value
- cumulation. /5/

I shall refrain from discussing here how far traditional mathematics education in school was able to attain the goal of teaching these.

Now it is evident that these central approaches to mathematical applications can be implemented both by discrete and by continuous conceptualizations. The corresponding continuous concepts are: function - differential equation - derivation - weighted integral - integral. As opposed to that, the corresponding conceptualizations in discrete analysis are

- sequence, time series
- difference equation
- difference
- arithmetical mean value
- sum.

These discrete concepts are obviously technically and intellectually much simpler than their continuous counterparts. The fact that they also in principle fulfill the requirements of, say, physics, particularly of mechanics, is shown by several textbooks. Thus, D. Greenspan writes, in the Preface to his book Arithmetic Applied Mathematics (1980):

> In this book we will develop a computer, rather than a continuum, approach to the deterministic theories of particle mechanics. Thus, we will formulate and study new models of classical physical phenomena from both Newtonian and special relativistic mechanics by use only of arithmetic. At those points where Newton, Leibniz, and Einstein found it necessary to apply the computational power of the calculus, we shall, instead, apply the computational power of modern digital computers. Most interestingly, our definitions of energy and momentum will be identical to those of continuum mechanics, and we will establish the very same laws of conservation and symmetry... In addition, the simplicity of our approach will yield simple models of complex physical phenomena and solvable dynamical equations for both linear and nonlinear behavior. The price we pay for such mathematical simplicity is that we must do our arithmetic at high speeds.

The following pages will give some justifications which are in my opinion crucial in answering the inevitable question now raised: "Why use the concepts of continuous analysis in school at all?"

In doing this, the reasoning will be developed mainly in five steps:

1. Continuous analysis is insufficient to obtain concrete numerical results.
2. Most concrete models of analysis have a discrete basis.
3. Nevertheless, the transition from models to concrete numerical results, in general, cannot be accomplished without continuous analysis (rounding errors, change of step-width, assumptions of invariance).
4. Some remarks on the role of the symbolic computations of traditional analysis in the present context of application.
5. Some considerations about the relevance of an applications-oriented approach.

/5/ cf. the recently published volume /Tietze et al. 1982/, in particular pp.111f.

As to 1.: Insufficiency of Continuous Analysis for Obtaining Concrete Numerical Results. Richenhagen /6/ has collected convincing evidence and examples for this. This is why we shall merely recall some of the facts: most integrations cannot be executed analytically, but only numerically; this is all the more true for solving differential equations. But even tasks as simple as determining the extremes of a familiar school function like x * sin x will require numerical methods. School mathematics has hitherto confined itself in a rather unnatural way to problems involving classes of functions which were solvable by analytic methods. It has paid dearly for this with heavy losses in reality content and relevance. Improvements by the use of hand-held calculators and computers would seem to be quite naturally in prospect here.

As to 2: Most Concrete Models of Analysis Have a Discrete Basis. This is first evident for modellings in the social sciences or in population biology, in which the quantities to be modelled are numbers of items or individuals, or monetary units, which anyway cannot be subdivided at will. But in physics, too, for instance, most models start discretely: even disregarding the fact that the universe is finite in principle and structured in particles, and that there are quanta (i.e. smallest units), it is a fact in case of quantities which are usually conceived of as being continuous, and mathematized accordingly in that discipline, that concrete models based say, on results of measurements, will start discretely simply because continuous functions cannot be obtained as results of series of measurements which yield only discrete sequences or time series (this does not hold, of course, for modellings based on theoretical approaches).

As to 3: The Transition from Models to Concrete Numerical Results, in general, Cannot Be Accomplished without Continuous Analysis. This is true, for one thing, because of the rounding errors which inevitably occur in numerical computing, and have to be controlled by a superordinate model (this is the other side of the complementarity between numerical and analytical mathematics developed by Richenhagen). A second, deeper reason follows from a closer look at the discrete aspects mentioned in points 1 and 2: it is found that the step-widths used in 1. and 2. are basically independent of each other. The density of the values measured in the measuring process is generally determined according to practical aspects. It results from consideration of information content and "cost". One of the most fundamental hypotheses for determining the step-widths is that a diminuition of step-widths may yield more exact results, but basically none which differ in principle. The phenomena which are to be observed and/or described are considered to be invariant with respect to the step-width used in the obserations, provided it is sufficiently small. This fits in with the assumption that the corresponding limits exist. It is only on the basis of this assumption that the measuring process can be carried out in a discrete way chosen by practical considerations. In this case, however, the phenomena concerned are basically invariant with respect to the step-width, and are thus best described in mathematical models which do not explicitly contain step-width. The fact that the step-width, with which the measuring data were obtained, is only of marginal importance even for the model, explains why the step-widths used, say, to numerically solve the corresponding differential equations, will generally be completely independent of the step-width used in measurement. The latter are determined by practical criteria such as cost and the precision required.

This fundamental consideration, which is decisive in what follows, has been formulated here only for the special case where the results of discrete measurement are used as a

/6/ cf. /Richenhagen 1981/

starting point. It is true, in an analogous way, for all the other cases in which mathematizing and modelling is done by analysis. In particular, this consideration helps us to explain why some disciplines in which the natural structure is discrete, such as number of individuals in population biology, nevertheless use continuous models, despite the fact that this would seem inappropriate at first glance: the impact of such small changes on the phenomena concerned in the respective models is only marginal.

As to 4: The Role of Symbolic Computations. If continuous analysis were necessary only to grasp concepts independent of step-widths, it could be understood as mere theoretical superstructure which guarantees the admissability of the conceptualizations and methods used, but which need not be further considered by the user. Yet beyond this role of theoretical superstructure, calculus may be a powerful tool, simpler and more elegant than discrete analysis. Apart from simple cases in which the functions used for modelling can still be handled entirely on the symbolic level, calculus furnishes an instrument which is of great value in identifying, beforehand, quite a number of qualitative properties of the solutions sought such as periodicity, boundedness, smoothness, potential singularities etc., and hence for guiding the numerical methods which are essential for quantitative evaluation. Further, for this kind of calculus, something analogous to numerical computing is true: the algorithmic work it requires can be taken over to a large extent by computers, while man increasingly has to take care just of global programming. The programs for symbolic computing, however, are much more complex than those used for numerical computing, and this is why they are not available on hand-held calculators as yet.

As to 5: The Relevance of The Applications-Oriented Approach. Analysis was able to take its largely undisputed position in the classroom mainly because the various disciplines and social groups were able to emphasize the great number of applications for whose understanding and technical command differential and integral calculus are indispensable. It is quite another question whether analysis as taught at school ever made sense in preparing for these applications, or in guiding pupils towards the latter; at least, I consider modifying analysis with this objective in view an important motive for working with analysis in school. This statement is not intended to dispute the fact that teaching analysis allows us to attain other significant goals of mathematics education at the Gymnasium which are independent of applications /7/. With regard to this second aspect, analysis can be replaced by other fields, but in terms of its concrete applications as dicussed above, calculus cannot be replaced. If this is accepted, the question whether continuous analysis is indeed necessary for applications today becomes crucial for legitimating analysis in school.

3. Conclusions for Analysis in School

Our considerations have shown that even today (so far as applications are concerned) continuous analysis cannot be dispensed with when describing problems for which analysis has been classically used. This, however, need not lead to the conclusion that analysis education at school should go on teaching continuous analysis as before. For the discussion has shown the function of continuous analysis in applications, and teaching must be done in such a way that this function is fulfilled. This requires that the transition from the discrete to the continuous model be experienced by the students and that the respective particular possiblities and limitations of the model type in question

/7/ For a pronounced view in this aspect cf. /Führer 1982/

be perceived. To me, it would seem dishonest to try to explain to the student the importance of analysis for applications by means of unrealistic and oversimplified minimum–maximum exercises. Rather, it would seem crucial to have the student at least begin to assess the usefulness of the various components of the system of analysis, i.e. concepts, approaches, calculi, translation schemes in practical applications. This goal should be attained by appropriate problem solving in the classroom; and explication should play a subordinate part. It remains to be seen how a balance between the individual components can be achieved. The following aspects, however, should be included in any case:

a) Analysis teaching should in any case include treatment and study of discrete models. This leads to numerical computations. It does not necessarily imply explicit teaching of numerical mathematics, but requires including important numerical basic facts such as propagation of errors.

b) Establishing models is an important activity which must not be neglected in favor of interpreting models. In particular, this means that the techniques of finding suitable functions are as important as discussing functions.

c) The role and function of (continuous) calculus must be developed in an appropriate way. It cannot be used to obtain numerical results, save in exceptional cases; it can, however, guide and direct the use of numerical methods.

d) The recent development of computer science has established techniques of description, in particular programming languages, which permit the precise description even of complicated processes such as, for instance, the algorithms necessary for symbolic differentiation. School mathematics should increasingly make use of this.

4. Some Examples to Illustrate the Ideas Presented

The following examples are intended to further clarify and illustrate the constructive ideas presented above. Further, their simplicity is meant to prove that they can basically be realized within the frame of school analysis. And lastly, they are intended to encourage the search for further, similar and better examples. The existing literature on computer use in analysis teaching /8/ abounds with examples which can be used in our sense. These, however, often seem to be alien to an analysis education oriented towards the classical goals.

4.1 Two Examples for the Transition from Discrete to Continuous Analysis

These examples are intended as introductory ones, i.e. to be used even before introducing the corresponding concepts of derivation, limit, and integral. They are, however, intended to lead towards these concepts.

/8/ cf. the fundamental papers indicated and the literature in /Winkelmann 1982/. Beyond that, there is an increasing number of articles in almost all periodicals on mathematics education which present similar ...

First Example: Free Fall

This example requires at least an intuitive familiarity with Newtonian mechanics, in particular with the essential phenomenon of inertia, i.e. with the fact that forces acting on a body which otherwise moves freely in space, will not directly induce this body to be displaced, but rather to change its velocity first. These intuitive ideas should be familiar indeed in the age of space travel. They are somewhat encouraged by computer games like moon landing etc. /9/, or by intentional learning environments such as the Dynaturtle in computer systems with turtle graphics /10/. Of course, this intuitive familiarity cannot be taught in the mathematics lesson at this point (introduction to analysis); the teacher, however, should remind pupils of the appropriate phenomena and make sure that these intuitions are indeed present in their minds.

A further favorable, but by no means indispensable prerequisite would be familiarity with the methods of considering motion as discrete (as may be done, for example, in the Mittelstufe (grades 9-11) by considering pursuit problems or biological search problems)/11/.
The task now is to model the free fall of a body under the influence of Earth's gravitational acceleration. X designates the distance covered (in meters), V the velocity attained, and T the time (in seconds) elapsed since the onset of free fall. A modelling approach is the following, where H means a small, but arbitrarily chosen step width:

```
Onset:  X: =  0
        V: =  0
        T: =  0

Step:   X: = X+V*H
        V: = V+G*H          (G = 9.81)
        T: = T+H
```

The observant reader will have noted that the order of sequence of the first two computing steps under "Step" could be reversed. The order chosen here means that V will be altered, i.e. increased, only after the displacement. This means, that the velocity is always chosen a bit too low, i.e. that the X values calculated will always be a bit too small. If, however, the two lines for X and V under "Step" are interchanged, we will always obtain a velocity a bit too high. This yields an opportunity to asses the margin of error, in quite a natural way, by approximating the "true value" from both sides. We shall not pursue this idea further here.

Instead, we shall start from the observation that the values obtained must always be too small, but that the error should get smaller as step width decreases, a fact which can

/9/ This refers to simulation games where the player can influence acceleration (thrust of retrorockets). As opposed to that, the commercial games in which a potentiometer serves to regulate speed or even position directly, are utterly unrealistic and at best ineffective for developing the concept of inertia.

/10/ Such learning environments (computer-aided microworlds) have been described in /Papert 1980/, chapter 5. Actual realizations for personal computers (Aplle II and TI 99/4a exist in the LOGO systems, cf. /Abelson 1982/ in particular p. 121 ff. Similar programs could be easily realized with UCSD Pascal for all personal computers able to visualize on the screen.

/11/ Biological search problems (animal seeks food, predator seeks victim) are found in /Abelson & di Sessa 1981/ p.70ff. For pursuit problems, cf. /Stein 1977/.

be illustrated numerically by several computer runs. Now we shall attempt to describe the results in an exact way by formulas, which is still possible in this simple case. What is sought is an expression for the value of X at a fixed time T which is subdidivided into n steps (T=n*H). Simulating the computer run with general numbers for small values of n then leads to the following formulas:

$$V_{T=n*H} = n * H * G$$

$$\begin{aligned} X_{T=n*H} &= 0 + H*H*G + 2*H*H*G + \dots + (n-1)*H*H*G \\ &= (1 + 2 + \dots + (n-1)) * (H*H*G) \\ &= \frac{(n-1)*(n-2)}{2} * H*H*G \\ &= \frac{n^2 - 3n + 2}{2} * H^2 * G \qquad \longleftarrow \quad T = n * H \\ &= \frac{G}{2} T^2 + \frac{T^2(-3n+2)}{2n^2} G \\ &= \frac{G}{2} T^2 + \frac{1}{n} (\ldots\ldots) \end{aligned}$$

Even without considering the limit, the last line can be interpreted to say that there is a fixed value, namely 1/2 * G * T^2, from which the value calculated by the computer deviates less and less the bigger n is, i.e. the smaller the step width is. Physics now suggests that we take this fixed value as the true value, and consider the deviations from it as conditioned by the the choice of discretizations, i.e as merely technical, and of no physical significance./12/
While the effects of decreasing step widths can be very closely studied in this example, it would not seem equally suggestive to develop the relationships between V and X, and between acceleration and V as quotient of differences and later as differential quotient. By contrast, interpreting the integral, i.e. V as cumulative effect of the acceleration G, or X as cumulation of V, is rather informative.

The example quoted can be continued, say, by considering the oblique throw, adding air resistance (this is where the analytical solution at this level fails), and finally treatment of a planet's movement round a central body. /12a/ These more complex examples may serve to further demonstrate the fundamental approach, i.e. constructing displacements by means of the changes in velocity. In this instance, the effects of decreasing the step widths can only be studied numerically.

This approach is obviously not concerned with teaching numerical techniques, and it is also not concerned with illustrating the limitations of numerical computing, for instance, by means of cumulating rounding errors: other topics such as the transition from the

/12/ The approach to discretization used here has been kept as simple as possible lest the fundamental efect of reducing step width be swamped by more technical ballast. The nesting method scetched above could be well used to illustrate the idea of a half-step procedure (cf. /Eisberg 1978/) which, however, will quite untypically yield precise values in the case of the quadratic goal function used here. To keep this false impression from sinking in, additional and more complex examples like those sketched in the following should be treated at all cost.

/12a/ A further continuation of this chain of models is given in /Bork, A. and Peckham, H., 1981/ where the main interest lies in the teaching of physics quotient of differences to the differential quotient are better suited for that purpose because of the "subtraction disaster" occurring there. Basically, the concern here is the field of application of differential equations, and the transition from local laws to global applications. These can be dealt with at the mathematical level assumed here because of the discrete method of proceeding and the direct, uncomplicated programming. Thus, the scope and the fundamental approach of analysis-related mathematizations can be shown beforehand./13/

The example quoted as an introduction is of course rather poor as seen from the application situation. The important thing with further examples is not only to develop models, but to use these purposefully for solving quantitative problems, asking, for instance, in the case of the oblique throw: "Which is the most favorable angle of throw?" or, in case of the trajectory of planets: "What is the relationship between distance and period of revolution?", or "Which kinds of reflexion angles are obtainable in reflecting space probes on far-away planets?" etc.

In this vein, quite a number of other examples can be found in application-oriented volumes on differential equations. Of course, it is not the technical treatment, requiring continuous analysis, which can be taken over from these books, but only the basic approach of the differential equation. Such topics could be population dynamics, radioactive decay and its applications, or the dosing of drugs./14/

Second Example: Calculating Areas

Integrals occurring in application situations can often be interpreted and understood in two different ways: as generalized product, or as cumulation of local effects. In the case of the generalized product (example: work = force multiplied by distance), the mean value of one quantity is multiplied with the other quantity; this is how calculation of an area by means of an integral can be conceived of as the generalization of calculating the area of a rectangle. The classical forms of writing the integral, especially in the form of the Stieltjes integral, are an expression of this conception. In case of the cumulation of local effect, as opposed to that, the integral is rather more conceived of as the solution of an uncomplicated differential equation. The insight that these two interpretations so different at first glance are equivalent represents the Fundamental Theorem of differential and integral calculus.

That the area can also be conceived of in the second, dynamical sense is illustrated by the following recursive formulation of an area calculation in the programming language ELAN:

/13/ This implies, of course, that teaching will really proceed to differential equations later on. Proceeding to these does not necessarily imply treatment of analytical methods of solution, but it does imply presenting and discussing the concept and basically properties and varieties such as parametricity of solutions, initial value problem etc.

/14/ For discrete and continuous models in population dynamics cf. /Rosenberg-Winkelmann 1979/. Many examples which can enhance pupils' motivation and which can be treated with this discretization approach are also found in the introduction to /Braun 1978/.

```
REAL PROC function (REAL CONST x)_;

REAL PROC area approximation (REAL CONST a, b, h):
  (*a and b are the interval boundaries with a < b, h is step width*)
  IF   b - a < h THEN (b - a) * function (a)
       ELSE h * function (b - h) + area approximation (a, b-h, h)
  END IF
END PROC area approximation;
```

In this case as well, error assessment is easily possible:

If /f (x) - f (y)/ <= L* (/x - y/),

the biggest possible error is equal to the number of recursion steps multiplied by maximum error in the small area, hence:

(b - a) / h * (L * h) * h = h * (b - a) * L

It is typical of this recursion that it allows us to focus attention entirely on the step, i.e. on the approximated calculation of a narrow area (which is done here by the approach of width multiplied with height <as measured at an arbitrary point>). This allows, on the one hand, an easier subsequent transition to other numerical methods of integration by simply calculating this narrow small area differently and more exactly (trapezoid method, etc.). On the other hand, it suggests a concept of area which is seen as a function of the upper limit of integration, a concept which already supplies important flexibilities of attitudes towards intuitions and concepts /15/.
Recursive formulation of this example of course requires a higher programming language. While the first example (free fall) could be formulated in any language, in BASIC this example can be calculated, but no longer formulated in a manner which facilitates understanding. Conversely, similar formulations are possible in PASCAL or LOGO. Among other things, this example was intended to show that the possibilities of expression of modern programming languages offer formulations and lines of thought which are not only elegant, but also essentially practical and helpful.

Developing the procedure for calculating areas will of course not be child's play. Students getting to know recursions for the very first time in this example will certainly be out of their depth. Students, however, who have worked with LOGO in the primary stage (ages 6 to 10) or in the secondary stage I (ages 11 to 14) will find no new difficulties.

Such examples and other similar ones aim at establishing the following understanding of continuous analysis, which is to be elaborated and deepened by further teaching:

Provided the step width (of the discrete methods) is small enough, the concrete results (trajectories, area approximations, growth curves) will change only marginally (in case of the phenomena observed here). This invariance is very important for applications; it is that which justifies the method of modelling extra- and intramathematical processes by means of arbitrarily chosen step width. (Continuous) analysis is the direct study of these invariances. It is indispensible for understanding such processes, in which local and

15/ This is not to question the fact that efficient programming requires recursions be partly dissolved, for instance in order to avoid multiple computions of the same function values. The intention here is to achieve an understanding, to communicate basic conceptions, and not yet to qualify pupils for maximum efficiency with the computer.

global behaviour are linked by regularities. Beyond that, analysis develops a symbolic calculus, which, in simple cases, permits us to calculate invariances directly, and is able, in general, to guide the use of numerical methods.

This latest statement will be explained in more detail in the following examples.

4.2 On the Role of Calculus

The fragments of examples which follow are not intended as introductions. In part, they already presuppose developed calculus. They are, however, meant to illustrate the role and function calculus may still have today. The first of these examples will show the useful co-operation of calculus and numerical techniques, the second will treat the programming of a calculus which is intended to achieve a change of emphasis from executing the calculus to understanding and developing the fundamental algorithms. The third example will show how calculus can be technically assisted by computer aids.

First example: Partial and Numerical Integration /16/

The following integral shall be calculated:

$$A = \int_0^1 \frac{1}{\sqrt{x}} \; \frac{1}{x+1} \, dx .$$

A direct numerical attempt at integration is bound to fail here, as the integral is improper. This is where theory or symbolic calculus must come to aid. Indeed, a partial integration approach according to

$$\int_a^1 f' g \, dx = f(1)g(1) - f(a)g(a) - \int_a^1 f g' \, dx$$

with

$$f(x) = 2\sqrt{x} , \quad g(x) = 1/(x+1)$$

will provide the following expression:

$$\int_a^1 \frac{1}{\sqrt{x}} \; \frac{1}{1+x} \, dx = \frac{2}{2} - \frac{2\sqrt{a}}{a+1} + \int_a^1 2\sqrt{x} \; \frac{1}{(x+1)^2} \, dx .$$

From this, we finally obtain, by passage to the limit $a \rightarrow 0$, as the equivalent expression of the integral to be calculated originally:

$$A = 1 + 2 \int_0^1 \frac{\sqrt{x}}{(x+1)^2} \, dx .$$

/16/ cf. /Lax et al. 1976/, p.222

If this integral cannot be evaluated without difficulties by the methods of calculus, it can now be treated numerically without problems as a proper integral.

Second Example: Symbolic Differentiation by Program

In traditional analysis education, symbolic computations rightfully play an important role (in English, this subject is even called "Calculus"), but they could hitherto be taught only by the method of "example and emulation" as there was no language to describe calculatory processes. Here, the systematic languages developed by informatics provide new opportunities of formulating, on a higher level of understanding, the classical calculi conceived of as symbolic algorithms, thus enabling students to better understand them. It is obvious, however, that the classical concept of calculi is not completely congruent with the algorithm concept of informatics - just consider the fact that not every function can be integrated. On the other hand, if we limit the scope of the manipulation, this may define a partial 'calculus' which corresponds to an algorithm in the sense of informatics.

In the following, the example of differentiating elementary functions will serve to show tentatively how some manipulations of analysis can be treated by means of interactive programming. The algorithms or programs resulting from this, however, are of far greater complexity than normal numerical programs. This complexity can be reduced by several measures: first, by providing certain complex data structures, including the necessary elementary operations - such as, for instance, an abstract type of data "elementary function" in the sense of modular programming, something which would be easily made possible by ELAN's package structure or MODULA 2's modulus concept; second, by not completely formulating the entire program and limiting it to a section which will run - for instance, only rational functions instead of general elementary functions, or only addition and product rules in the derivation of elementary functions, and appropriate classes of functions. To me, it would seem important that selecting such a partial section and the program-specific and mathematical prerequisites of the total program are openly discussed in the classroom; this might lead to a far deeper understanding of the discussion of a function than the mere traditional manipulation.

As an example for such programs, a section of the formulation of a sectional program in LOGO is given /17/.

In this example, elementary functions are represented as a tree; vertices are the simple functions, i.e. constants, id, sin, cos, exp, etc. Each node consists of an operation +, -, *, ^ (power), O (chain). This is implemented in LOGO as a (recursive) list, the function

$x \rightarrow x * \sin(x + a)$,

for instance, being represented by the nested list of three elements

[ID TIMES [SIN O [ID PLUS 2]]];

the first element being ID, the second the operation TIMES, the third element a list which is itself again an elementary function.

/17/ cf. /Winkelmann 1982/.

The procedure of derivation now is recursive like the data structure on which it is based; according to the rules of derivation, the derivation of a composite function is led step by step to an elementary function, which is composed of the partial functions and their derivations. We reproduce one section /18/:

```
TO DERIVATIVE :FUNCTION
IF LENGTH :FUNCTION = 1 _
IF LENGTH :FUNCTION = 3 THEN
   IF FIRST BUTFIRST :FUNCTION = THEN CHAIN.RULE :FUNCTION
   _
END

TO CHAIN.RULE :FUNCTION
OUTPUT (LIST OUTER.DERIVATIVE "TIMES INNER.DERIVATIVE)
END

TO OUTER.DERIVATIVE
OUTPUT (LIST (DERIVATIVE FIRST :FUNCTION) "O (LAST :FUNCTION) )
END

TO INNER.DERIVATIVE
OUTPUT DERIVATIVE LAST :FUNCTION
END
```

It would seem obvious that the mechanism of derivative calculus, including the necessity of the various rules, can be better understood after such a program has been discussed in the classroom.

Third Example: Interactive Assistance of Calculus

As there are already some program packages (available for microcomputers as well) which control parts of symbolic calculus, it is suggestive to use these to assist work with calculus - just as the hand-held calculator can be used to take over required numerical calculations. Thus, for instance, Simmonds (1982) presents an integration package in BASIC which, operated by the user, will carry out certain manipulations - substitutions, partial integrations, reductions, etc. - with the expressions presented to it. The program package muMATH from MicroSoft, which is available for the operation system CP/M used on microcomputers and some other, goes even further. It is able to carry out nearly automatically rather complicated symbolic computations, which, in part, significantly exceed the requirement level of the graduation examination in secondary education (Abitur) /19/. This may lead - as has been indicated in the introduction - to a certain neglect of concrete calculus-related manipulation abilities and to increased emphasis on the principles underlying calculus, as well as on its possibilities and limitations.

To give a concrete example: in the program package muMATH already mentioned, there are prefabricated functions for (symbolic) differentiation and integration, which, however, do not contain any heuristics for approaches such as partial integration, but rather include the necessary algebraic transformations and simplifications as an integral part.

For the expression defined by

    F : LN (X^2 + A),

where ":" denotes the assignation operator,

    DIF (F, X),

for instance, will provide the derivation of F for X, i.e.

    2 * X / (A+X^2).

As opposed to that, an attempt at integration according to

    INT (F, X)

(indefinite integral over F with regard to X) will not be directly successful because of the concrete form of the expresion F. At this point, the function PARTINT (either provided by the teacher or developed collectively in the classroom) may assist. It can be formulated as follows in the language muSIMP on which the package muMATH is based:

```
FUNCTION PARTINT (F, U, X),

V : F / DIF (U, X),

U * V - INT (U * DIF (V, X), X),

ENDFUN.
```

The function PARTINT is called with three parameters: with the expression to be integrated F, the choice U for the partial integration according to F = U' * V, and with the integration variable X. The last but one line of the function definition tells how the expression returned back by the function is to be calculated. By means of this function, the choice

    PARTINT (F, X, X)

now leads, for A > O, to the solution desired

    -2*X + X*LN(A+X^2) + 2*A^(1/2)*ATAN(X/A^(1/2)).

This function PARTINT now can serve, on the one hand, to illustrate the co-operation of heuristics and calculus, and on the other hand it provides an opportunity to get rapidly acquainted with a large number of approaches for partial integrations and to test them. Appropriate printouts of intermediate results could of course be easily arranged.

Literature

Abelson, H. & diSessa, A.: Turtle Geometry: The Computer as a Medium for Exploring Mathematics. MIT Press, Cambridge, MA, 1981.

Abelson, H.: Logo for the Apple II. BYTE/McGraw-Hill, Peterborough, NH, 1982

/18/ This is based on the LOGO offered for Apple II by Terrapin, version 1.0; cf. /Abelson 1982/.
/19/ For a first general desciption cf. /Wilf 1982/.

Bork, A. and Peckham, H., 1981: Computer Applications in Mechanics. In: Bork, A., Learning with Computers. Digital Press, Bedford, MA 1981, 83-97.

Braun, M.: Differential Equations and their Applications: An Introduction to Applied Mathematics. Springer, New York 1975.

Eisberg, R.M.: Applied Mathematical Physics with Programmable Pocket Calculators. McGraw-Hill, New York 1976.

Führer, L.: Analysis als Pflicht? - Begründungsversuche, Trends und Neuansätze. In: H. Pfeiffer, H.G. Steiner (Hrsg.): Fragen der Differenzierung im Mathematikunterricht der gymnasialen Oberstufe. Tagungsband zu einer gleichnamigen Tagung vom 1.-5.6.1982 in Haus Ohrbeck bei Osnabrück. IDM, Bielefeld 1982.

Greenspan, D.: Arithmetic Applied Mathematics. Pergamon Press, Oxford 1980.

Lax, B., Burstein, S., Lax, A.: Calculus with Applications and Computing, Vol. I. Springer, New York 1976.

Papert, S.: Mindstorms: Children, Computers, and powerful Ideas. Harvester Press, Brighton 1980.

Richenhagen, G.: Zwei Fallstudien zur Numerik. IDM, Occasional Paper Nr. 2, Bielefeld 1981.

Richenhagen, G.: Numerisch vs. Analytisch - Überlegungen zum epistemologischen Ort der Schulanalysis. mathematica didactica 6 (1983), 45-56.

Rosenberg, D., Winkelmann, B.: Deterministische Wachstumsmodelle für eine Population. In: Winkelmann, B. (Hrsg.): Mathematische Modelle in der Biologie. Band I. IDM: Materialien und Studien Bd. 14, Bielefeld 1979, p. 9-70.

Simmonds, D.G.: A computer-assisted integration package. In: Int.J.Math.Educ. Sci.Technol. 13 (1982) p. 427-440.

Stein, G.: Verfolgungsprobleme im Mathematikunterricht. In: Der Mathematikunterricht 23 (1977) H. 1.

Tietze, U.P., Klika, M., Wolpers, H.: Didaktik des Mathematikunterrichts in der Sekundarstufe II. Vieweg, Braunschweig-Wiesbaden 1982.

Wilf, H.S.: The Disk with the College Education. In: American Mathematical Monthly 89 (1982), p. 4-8.

Winkelmann, B.: Hand-held calculators and mathematics educators. Some strategic perspectives. In: H.G. Steiner (Ed.): Comparative Studies of Mathematics Curricula. Change and Stability 1960 - 1980. Bielefeld: IDM 1980. Materialien und Studien Band 19, p. 574-596.

Winkelmann, B.: Ein System von Aspekten einer schulischen Analysis. In: IDM, Schriftenreihe 7/1976/81, p. 55-96. Bielefeld 1981.

Winkelmann, B.: Ansätze für den Computereinsatz im Analysisunterricht. In: LOG IN 2 (1982), Heft 2, p. 22-26.

THE INFLUENCE OF COMPUTER GRAPHICS ON THE TEACHING OF GEOMETRY

Simo K. Kivelä

Helsinki University of Technology

Since last spring we have had in Helsinki University of Technology an IBM computer dedicated to CAD, i.e. Computer Aided Design. We have used the system in teaching geometry, too, and we think that this field will be increasing next years. The programs which we use are

CADAM (= Computer-Graphics Augmented Design and Manufacturing System) by CADAM Inc. (in Germany the name is CODEM)

and

CATIA (= Computer-Graphics Aided Three Dimensional Interactive Applications) by Dassault Systemes.

The main part of CADAM is two dimensional; that means, it is possible to produce only geometric plane figures. CATIA is fully three dimensional; it can handle models made of solids, surfaces etc.

I will represent here something about the influence of computer graphics and CAD systems on the teaching of geometry.

THE HISTORY OF GEOMETRY IN HELSINKI UNIVERSITY OF TECHNOLOGY

Traditionally, we have had two courses in geometry for first year students. The course of analytic geometry is compulsory for everyone. It consists of lectures (36 hours) and exercises (24 hours); the contents are usual: points, lines, planes, conics, vector algebra, something about matrices etc. The other is a course of classical descriptive geometry, lectures 26 hours and the same amount for drawing exercises. The purpose is to teach the geometric grammar of picture drawing: how to make pictures using parallel or central projections, how to construct the section lines of solids in the pictures etc. Of course, the aim is also to develop three dimensional thinking.

Twenty years ago, the course of descriptive geometry was compulsory for nearly every student, but its importance has since decreased. Today, only 5 percent of our students take this course. During some last years we have been moving the emphasis in the direction of the geometric principles of computer graphics. Now when we have the CAD system, we think that we should have a course of a new kind; not classical descriptive geometry but something about the geometry which one needs in order to understand computer graphics and CAD systems.

COMPUTER GRAPHICS AND CAD

In the simplest form, computer graphics means the drawing of line figures on the screen of a terminal or on a sheet of paper by means of a plotter. Curves are drawn as broken lines. In colour graphics the desired colour can be given to the line segments and it is also possible to fill a closed area by a colour.

The computer graphics of this kind is sufficient for many purposes: computer games, statistical diagrams, texts of different styles etc.

The geometry which is needed, is very elementary. It is only two dimensional and there is no need for many concepts: point, line, broken line, circle, curve, probably ellipse and something else; that is all. But, it has to be emphasized that it is absolutely necessary to master these few concepts in order to be able to discuss the simple geometry on the screen.

However, simple computer graphics is not sufficient for all applications. For example, one can think about the industrial design processes. The goal is to make a three dimensional thing, a car, an aeroplane, a piece of a machine or something else. Traditionally, the design meant drawing two dimensional pictures of the thing. But because the thing itself is three dimensional, it is of course better, if one can design the three dimensional object directly and not only pictures of it.

Therefore, CAD systems have been developed which can handle - more or less - three dimensional information. In a fully three dimensional system one defines a three dimensional model of the object in the memory of the computer. The model consists of geometrical entities (points, lines, curves, planes, faces, surfaces, solids) and the relations between them. The definition is done by using keyboards (one or more), light pens and tablets at the terminal. A picture of the model is needed on the screen; the light pen (or the tablet) is used in referring to the entities which are already defined and which are seen on the screen. For instance, if one likes to rotate a part of the model in a new position, one has to define the rotation as a geometrical mapping (the angle of rotation is given by the keyboard, the axis by the light pen) and then to apply it to the desired part (given by the light pen).

The difference between the three dimensional model and its picture must be emphasized. The model is in the memory of the computer in an abstract form; during the design process it is possible to form different pictures of the model on the screen without changing the model itself. The model can be seen in the pictures from different directions, the pictures can be scaled etc.

It is quite clear that in CAD much more geometry is needed than in simple computer graphics. The main reason is the three dimensional thinking. Two aspects must be considered:

a) The algebraic aspect or how to compute the geometry which is needed. This is important for a person who will design computer graphics (especially CAD systems) or who will program the routines. Everyone who uses (or buys or sells) computer graphics or CAD systems should know something about this field.

b) The purely geometric aspect is important to both the designer and the user of CAD. Some kind of geometric imagination is needed, but this is not always the traditional way of thinking. For example, the thinking by geometrical mappings is useful. Traditional two dimensional constructions must be generalized to three dimensions: everyone knows how to find the two points on a line which have the given distance to a third point, but in CAD we need also the three dimensional counterparts.

WHAT KIND OF GEOMETRY DO WE NEED TODAY ?

During some past decades it has been thought that the importance of geometry is decreasing. In the schools and on the university level too, the courses of geometry have been reduced. But I think that geometry is more important than the situation of today shows. We have to understand and shape the three dimensional world around us and geometry is a good tool for this purpose. In addition, we use more and more computer graphics and also therefore we need more and more geometry. Of course, computer graphics is not the only point of view on geometry; for example the historical and axiomatic aspects are also worth studying, but I do not consider them here.

It is not an easy task to outline the curricula of geometry for today. In the following I will give some examples which - I hope - show the direction where we should go. I have mainly thought about the university level and especially technical universities.

1) Projection mappings. In order to form a two dimensional picture of a three dimensional object we need some projection mapping. The most usual ones are different parallel and central projections. In figures 1 and 2 there are two parallel projections of the same house; figures 3 and 4 are central projections. Of course, there are other projections, too. Figure 5 is the stereographic projecton of a part of wall, figure 6 is a cylindrical projection of the same object.

The geometry of projection mappings which is needed, is of two kinds: the geometric properties of regularity (e.g. in the parallel projection the image of a line is a line; in the central projection the cross ratio is an invariant) and the computing of the mappings. In the latter, vector algebra, matrices and homogeneous coordinates are good tools.

Sometimes it is necessary to analyze the relations between two projections of the same kind. An example is the stereoscopic pair (figure 7).

2) Three dimensional models. When one defines a body of the three dimensional space, one has to form a model of it. But there are models of different types. The simplest one is the wire frame model. As an example we can think a dodecahedron (figure 8). In the wire frame model we have only the line segments which form the edges of the dodecahedron.

In order to be able - for example - to compute where a line intersects the dodecahedron, we have to know the parts of planes which limit the body. By adding these parts to the model we get the face model.

The body is not completely defined before we know where is the matter, inside or outside of the surface of the body. When we add this information to the model, we have the volume model.

As an example of a wire frame model where the position of the faces and the matter is not trivial, is figure 9.

The handling of the models leads often to the problems of the classical topology: the relations between the numbers of faces, edges and vertices, the connectivity of bodies.

3) Curves and surfaces are in computer graphics usually defined through parameter representation; the approximating functions are often polynomials. To draw the curve means to compute several points on the curve uniformly and to connect them using line segments. Thus, the curve is represented by a broken line, which cannot be distinguished from the curve, if the number of the points is sufficiently large. In the case of a space curve the points must of course be projected to a plane in order to get a picture. The surfaces (or patches of surface) are drawn using the isoparametric curves, i.e. the curves on the surface along which one parameter is constant.

To compute the points uniformly on the curve is not always easy. One can think for example the curve by name Folium of Descartes whose parameter representation is

$$x = \frac{2t}{1 + t^3}, \qquad y = \frac{2t^2}{1 + t^3},$$

where $-\infty < t < \infty$ (figure 10).

When one goes along the curve from top left to the origin, the parameter t varies from -1 to 0; on the loop from 0 to $+\infty$, t = 1 at the midpoint of the loop; from the origin to down right from $-\infty$ to -1. Hence, the growth of the parameter is not at all uniform and it can be difficult to find the points uniformly on the curve. The ideal parameter is of course the arc length and the solution of the problem is to change the parameter. This can be done by using classical differential and integral calculus either analytically or numerically.

Differential and integral calculus has other geometric applications, too, in handling curves and surfaces. Examples: tangent and normal vectors, curvature, developing of surfaces without or with stretching and compressing.

4) Geometric transformations. In order to handle objects in the three dimensional space (or in the plane) and to make geometric constructions one needs geometric transformations: translation, rotation, reflection, scaling, probably affinity and projectivity.

As in the case of projection mappings, the geometry of two kinds is needed: the purely geometric aspects, i.e. thinking through transformations, and the algebraic aspect, how to compute the transformations.

An algebraic example: Given an orthogonal 3x3 matrix A for which det(A) = 1. A matrix of this kind represents a rotation whose axis goes through the origin. The axis and the rotation angle are to be found. This is an eigenvalue problem!

An example of the thinking through transformations: Suppose that two of the faces of a dodecahedron are given (figure 11). In order to create the lower part of the dodecahedron it is sufficient to define a rotation of 72 degrees around the z axis and to apply it four times to the face F and its copies (figure 12). The upper part of the dodecahedron can then be created by reflecting the lower part in the xy-plane (figure 13), rotating the result 36 degrees around the z axis (figure 14) and translating it up to the correct position (figure 8).

5) Geometric constructions in the three dimensional space often require generalizations of the ideas which are used in the two dimensional case.

In figure 15 is shown the construction of three adjacent faces of a dodecahedron. At first the faces S1, S2 and S3 are lying in the same plane. Then we rotate the face S2 around the axis AB and the face S3 around the axis AC until the points D and E meet. The point D is moving in the plane T2 which is normal to AB, the point E in the plane T3, normal to AC. The points meet on the intersection line L of the planes; this is easy to construct. The correct point on the line is the intersection of L and a sphere whose center is A and radius AB. (Euclid used circles, but we can also use spheres as easily!)

6) Beautiful pictures. In order to get beautiful pictures in computer graphics one has to remove hidden lines, construct shadows, shade the surfaces, form the reflections of light etc.

Some of these constructions - removing the hidden lines for example - are easy for a human being, but they all are hard for the computer. On the other hand, the constructions are important because they considerably increase the clarity of the picture (figures 16 and 17). A good mastery of the methods of analytic geometry is needed here.

COMPUTER GRAPHICS AS A MEANS OF VISUALIZATION

Good graphics are useful in teaching many fields of mathematics: geometry, differential and integral calculus, complex analysis etc. But usually it is hard to produce good figures. In many books one can see ellipses which look like breakfast rolls; if one has tried to draw a picture of a three dimensional geometric configuration, one has found that it is very easy to get three non-collinear points on the same line in the picture.

A solution to these problems could be computer graphics. Beautiful smooth curves can be drawn comparatively easily. If the graphics system is fully three dimensional, the line of sight can be changed if the first picture of the object does not look clear and good. It is easy to produce different views of the same object (figures 18 and 19).

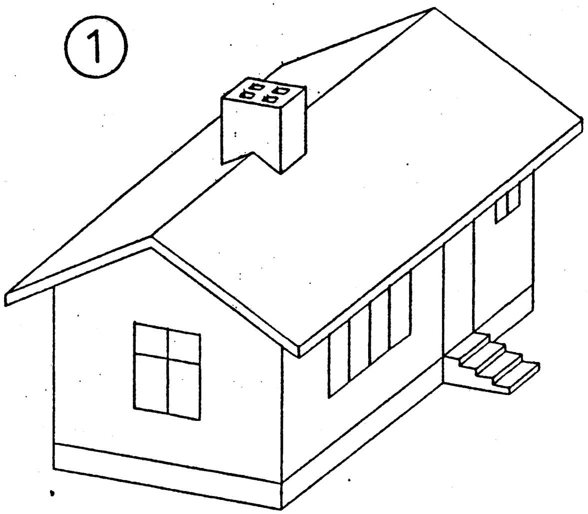
The only problem is that we do not have computer graphics systems which would be fully suitable for the purpose. The CAD systems are planned for the industrial design processes. They contain many special features which have no use in the visualization of mathematics and there is no reason to pay for these ones. On the other hand, they lack procedures which could be extremely useful, e.g. it is not usually possible to define a surface using the equation $z = f(x,y)$.

Another possibility could be to use graphic packages such as GPGS or DISSPLA. But then we have to do a lot of work: first to write the program which calls the routines of the package, then compile, load and execute it; and if we do not like the result, repeat the whole procedure.

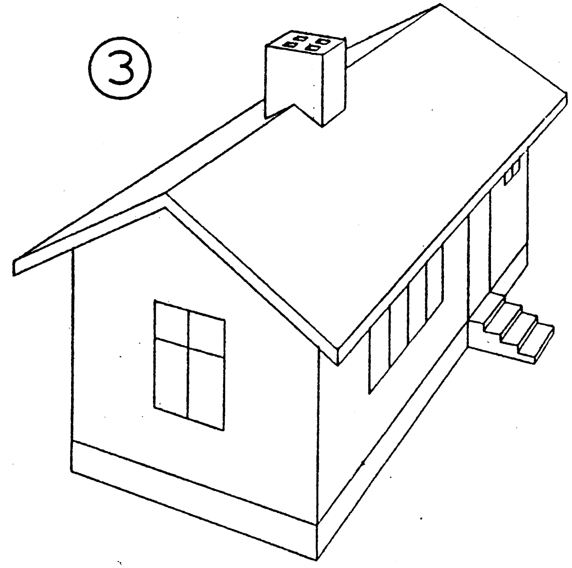
What we should need, could be some kind of three dimensional graphic editor, an easy way to define and reform objects of two or three dimensional space. It is surely possible to build such an editor on DISSPLA for example. Today, I do not think that any good system for this purpose exists yet.

28.

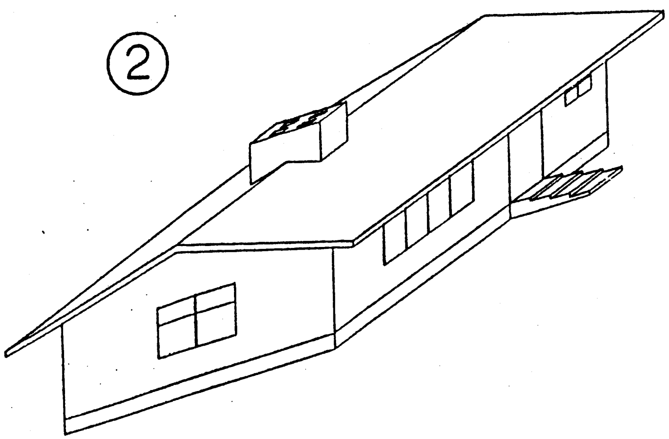
1



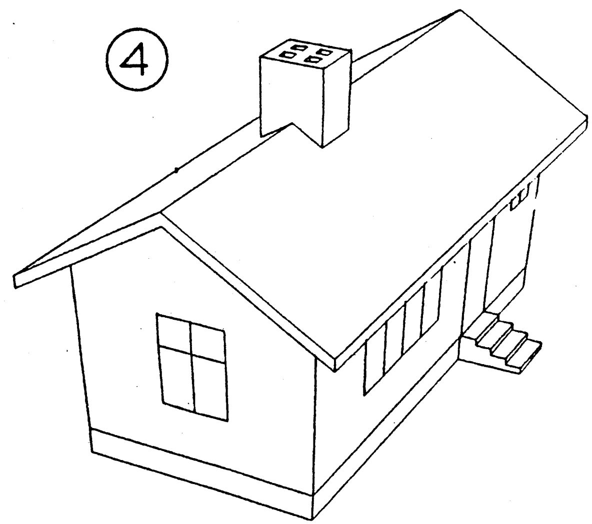
3



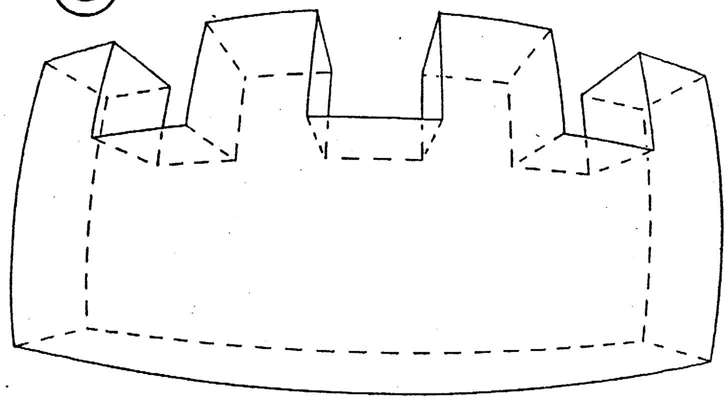
2



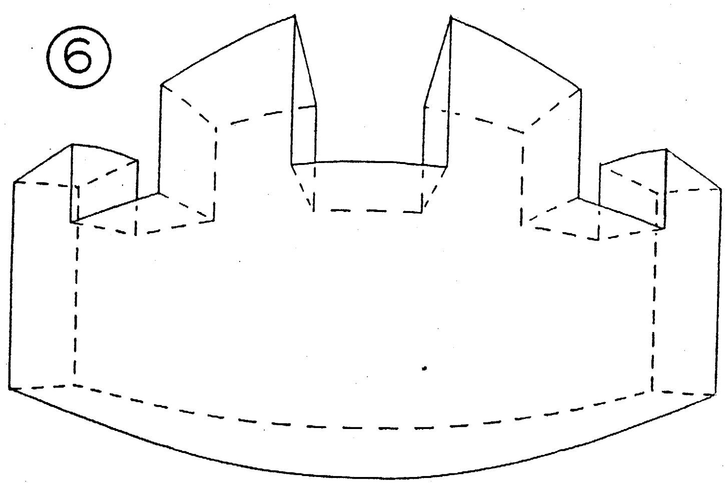
4



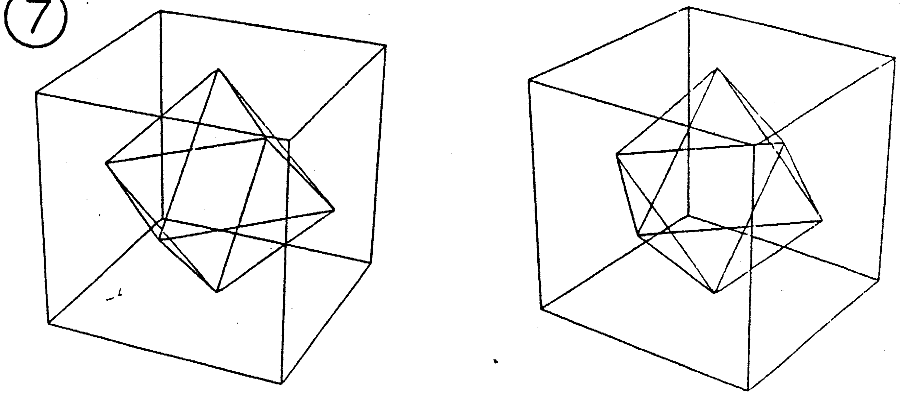
5



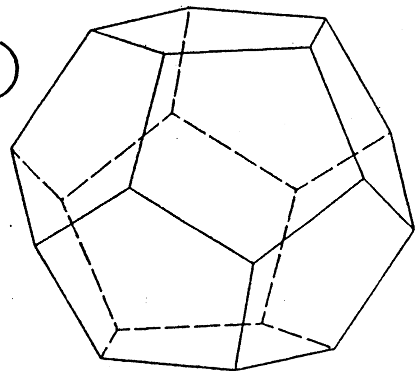
6



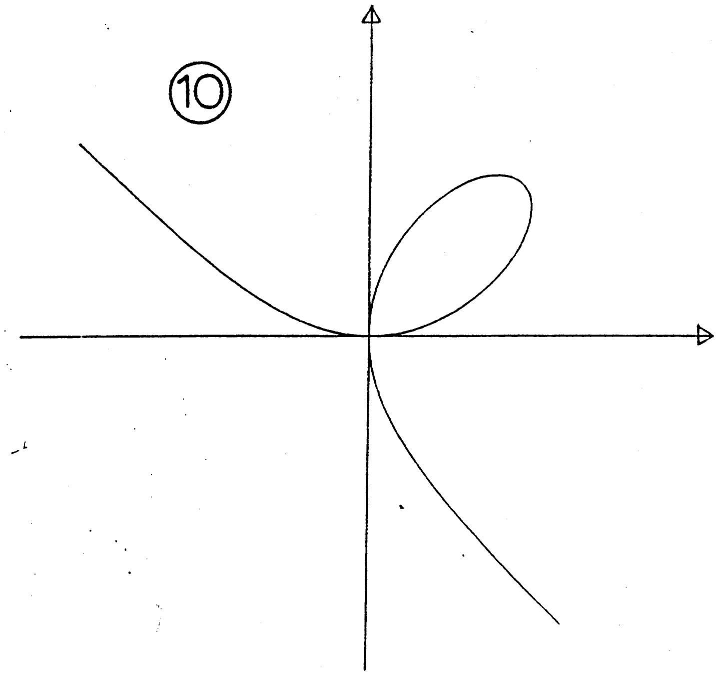
7



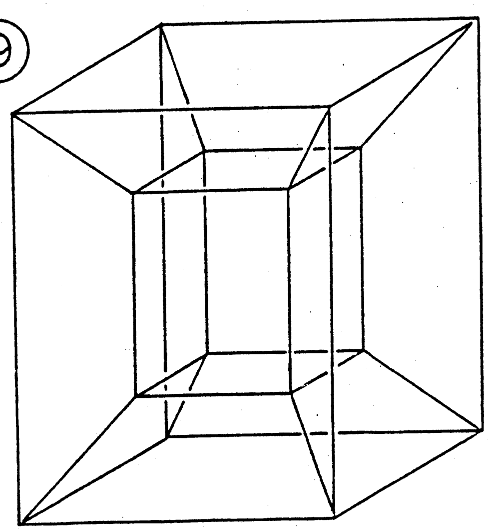
8



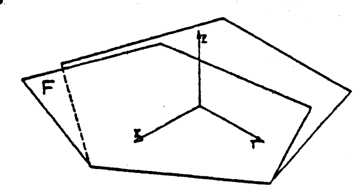
10



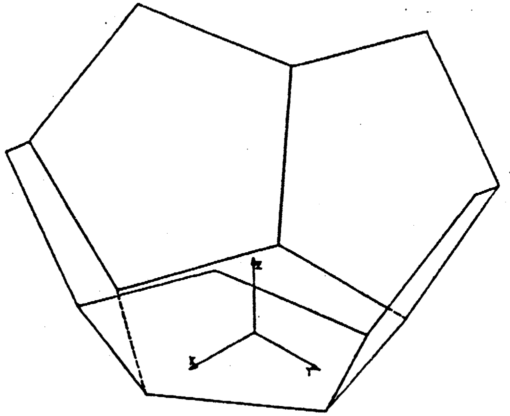
9



11

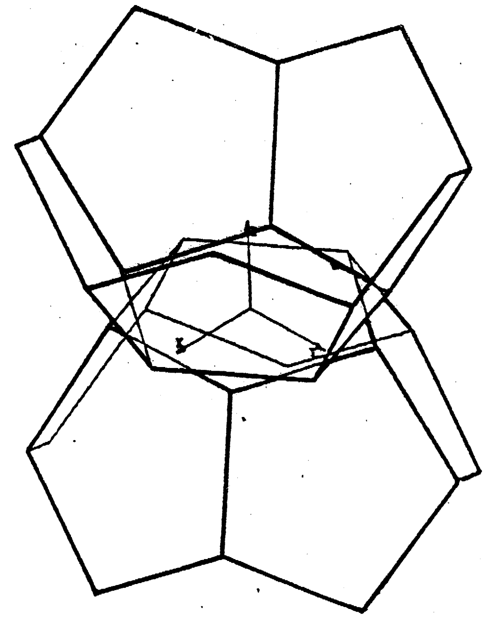


(12)



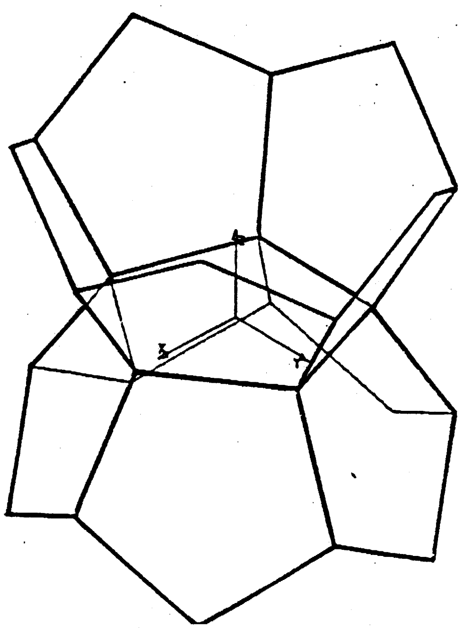
12

(14)

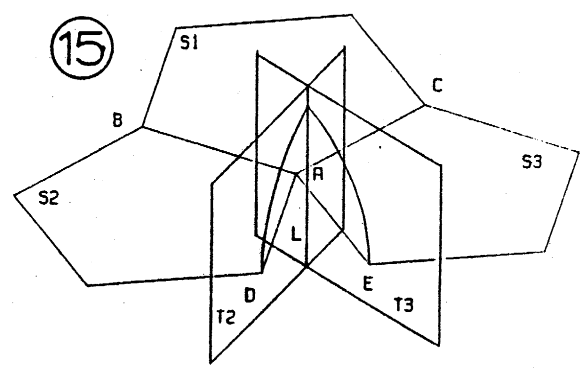


14

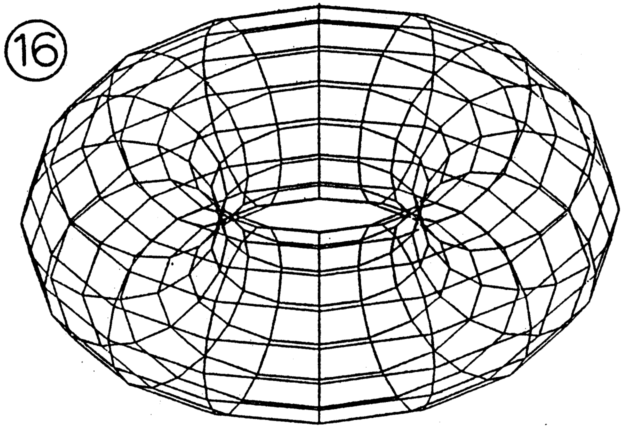
(13)



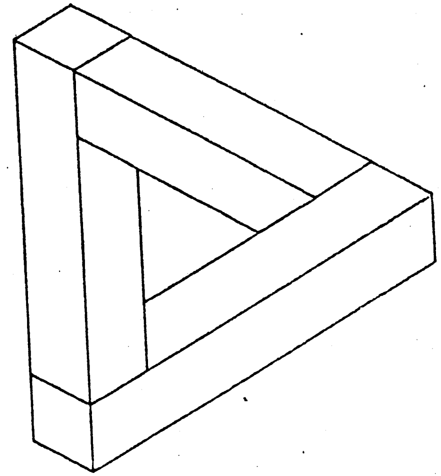
(15)



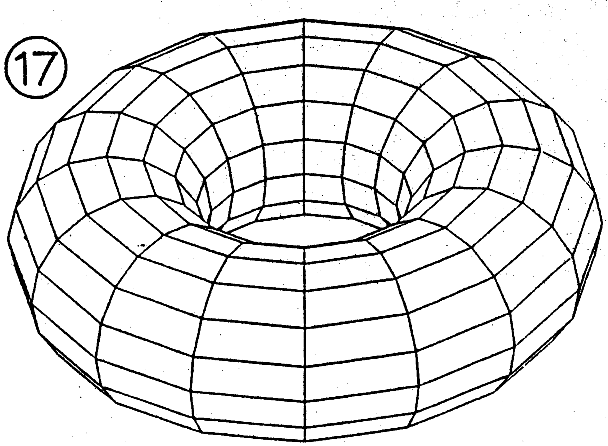
16



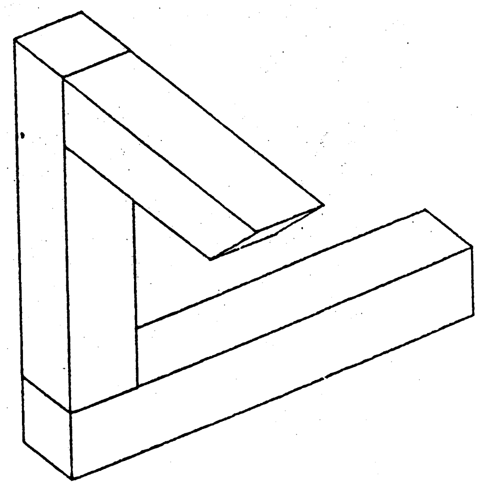
18



17



19



Les surfaces peuvent-elles être représentées par l'ensemble des zéros d'un polynôme à trois variables ?

François APERY

La géométrie des surfaces de l'espace $\mathbb{R}^3$ est un domaine privilégié où des moyens informatiques graphiques et un langage de calcul symbolique peuvent non seulement aider à visualiser les objets étudiés mais également à les construire et même à les concevoir.

Dans l'exemple que nous allons développer ici l'objet géométrique sera une surface immergée dans $\mathbb{R}^3$, c'est à dire l'image dans $\mathbb{R}^3$ d'une surface compacte connexe sans bord de classe C^1 par une application de classe C^1 et de rang deux. Une telle application peut admettre des points multiples correspondants à des points d'auto-intersection de la surface immergée, et on supposera que l'intersection des plans tangents en un tel point est de dimension minimum. Une application ainsi définie sera appelée immersion. Si il n'y a pas de points d'auto-intersection l'application est un plongement et la surface est dite plongée.

Pour chaque surface peut-on trouver une immersion dont l'image soit l'ensemble des zéros réels d'un polynôme à trois variables ?

Pour les surfaces orientables la réponse est affirmative même en remplaçant immersion par plongement: par exemple l'ensemble des zéros réels de $f^2(x,y) + z^2 - a^2 = 0$ est une surface de genre topologique p, où a est un nombre strictement positif suffisamment petit et

$$f(x,y)= p^{2p}.y^2+(x^2-p^2)\left(\sum_{k=0}^{[\frac{p-1}{2}]}(-1)^k\binom{p}{2k+1}x^{p-2k-1}(p^2-x^2)^k\right)^2$$

La courbe algébrique réelle $f(x,y) = 0$ paramétrée par $x=p\cos t$, $y=\sin pt$ est une courbe fermée à p boucles ce qui explique que le genre de la surface obtenue soit p. (fig 1)

Les surfaces non orientables sont immergeables mais non plongeables dans $\mathbb{R}^3$, elles ont nécessairement une courbe d'auto-intersection.

L'existence de polynômes dont une composante connexe de l'ensemble des zéros est une surface immergée non orientable de caractéristique paire inférieure ou égale à -2 a été prouvée par N.H. KUIPER .

Nous allons développer le cas de la surface de Boy caractérisée comme l'image d'une immersion du plan projectif réel (surface de caractéristique 1) dont la courbe d'auto-intersection est une hélice tripale à pas positif et dont chaque pale borde un disque. (fig 2,3) Signalons que le problème de l'existence d'un polynôme dont la surface de Boy soit l'ensemble des zéros s'est posé dés sa découverte par W.BOY en 1901. Nous pouvons décomposer la construction de ce polynôme en huit étapes pour mettre en évidence les moyens graphiques et calculatoires mis en oeuvre. Le détail des démonstrations se trouve dans: F. APERY La surface de Boy, Publication de l'I.R.M.A. Strasbourg 1984 .

1ère étape

Partant du contour apparent de la surface de Boy vue du point à l'infini de son axe de symétrie ternaire que l'on interprète ,suivant une idée de B. MORIN , comme une perturbation de l'ensemble des valeurs critiques de la singularité du mouchoir plié en quatre modulo une symétrie d'ordre 3 (fig 4,5,6), on établit une paramétrisation polynomiale de la projection de la surface de Boy sur un plan perpendiculaire à son axe de symétrie ternaire. Les calculs sont élémentaires et les polynômes trouvés sont choisis pour leur simplicité algébrique et la conformité de l'aspect du contour apparent. Une permutation circulaire des variables doit entrainer une rotation d'un tiers de tour pour respecter la symétrie d'ordre trois de la surface de Boy. La paramétrisation retenue est

$$X= 2((2u^2-v^2-w^2)(u^2+v^2+w^2) + 2vw(v^2-w^2) + wu(u^2-w^2) + uv(v^2-u^2))$$

$$Y= 2\sqrt{3}((v^2-w^2)(u^2+v^2+w^2) + wu(w^2-u^2) + uv(v^2-u^2))$$

La conformité de l'aspect du contour apparent laisse beaucoup de degrés de liberté et son contrôle se fait à l'aide d'une table traçante.

$2^{ème}$ étape

On cherche à relever la projection précédente par une fonction hauteur Z(u,v,w) admettant un maximum trois cols et trois minima sur le plan projectif réel. Si on représente le plan projectif réel comme quotient de la sphère $u^2+v^2+w^2 = 1$ par l'action antipodale et si ses pôles sont sur la droite $u=v=w$, alors la fonction hauteur définie sur cette sphère doit admettre deux maxima aux pôles, six minima répartis en un hexagone régulier sur l'équateur, et six cols aux sommets d'un octaêdre dont deux faces opposées sont approximativement des plans tropicaux, les sommets étant en quadrature de phase avec l'hexagone des minima (fig 7). Là encore le polynôme choisi

$$Z = (u+v+w)((u+v+w)^3 + 4(v-u)(w-v)(u-w))$$

l'est en fonction de sa simplicité (degré minimum) et de la conformité de la surface obtenue.La détermination de ce polynôme a nécessité l'emploi de moyens graphiques pour orienter la recherche (fig 8,9).

$3^{ème}$ étape

La surface rationnelle que nous venons de définir est paramétrée par trois polynômes homogènes du quatrième degré. Il faut calculer le rang de cette paramétrisation pour s'assurer qu'il est bien égal à deux. Ce calcul revient à montrer que trois polynômes homogènes à trois variables ne s'annulent pas simultanément sur le plan projectif réel. Il s'agit d'un problème d'élimi--nation sur le corps des réels et non des complexes, car un résultat d'al-gèbre montre que ces polynômes ont nécessairement un zéro projectif complexe en commun. La technique du résultant ne s'applique donc pas et il faut éli--miner les variables une à une. La taille des polynômes interdit l'aboutis--sement des calculs à la main et justifie l'intervention d'un ordinateur sur lequel est implanté un langage de programmation symbolique (Macsyma dans notre cas). L'élimination réelle se ramène à l'existence de racines sur un segment de $\mathbb{R}$ pour un polynôme du douzième degré à coefficients entiers in--férieurs en valeur absolue à 3.10^8. Vue la grandeur des nombres , la con--clusion par la méthode de Sturm reste à la machine.

La surface ainsi construite est du seizième degré et l'étude de sa courbe d'auto-intersection se révèle inabordable même avec de puissants moyens de calcul. Seule une allure approximative peut en être donnée par son observation graphique à l'aide d'ordinateurs traitant des images à trois dimensions. De plus la surface de Boy obtenue ne constitue pas toute la partie réelle de la surface algébrique complexe du seizième degré et nous n'avons aucun moyen de controler qu'elle en constitue même une composante connexe. Néanmoins l'étude de ce modèle va permettre d'en concevoir un autre plus satisfaisant quoique paramétré par des fractions rationnelles plutot que par des polynômes.

$4^{ème}$ étape

L'usage d'une table traçante et d'un programme ad hoc permet d'étudier les sections planes du modèle construit précédemment perpendicu--lairement à l'axe de symétrie ternaire et de reconnaitre parmi elles la section du plan des cols qui se décompose en une courbe du type ovale et et une courbe fermée à trois points doubles (fig 10). Si on veut imposer à notre nouveau modèle d'être de degré minimum, c'est à dire six, il est clair que l'ovale doit être un cercle et la courbe à trois points doubles une hypocycloide à trois rebroussements allongée(fig 11).

$5^{ème}$ étape

Diversesconsidérations géométriques ont conduit à penser que l'on pouvait imposer au modèle d'être engendré par des ellipses passant par un point fixe appelé pôle et dont les plans envelopperaient un cône de sommet au pôle coupant le plan des cols suivant une hypocycloide à trois rebroussements. Toutes ces hypothèses sont vérifiées par la surface romaine de Steiner (fig 12), et il est naturel de chercher à déformer les ellipses de cette dernière de façon qu'au lieu de couper le plan correspondant au plan des cols de la surface de Boy suivant un cercle double, elles le coupent suivant un cercle et une hypocycloide à trois rebroussements allongée.

La paramétrisation de la surface romaine considérée est

$$\left|\begin{array}{l} X \\ Y \\ Z \end{array}\right. = (1+t^2)^{-1}.\left|\begin{array}{l} \frac{\sqrt{2}}{3}(\cos 2\theta + t\sqrt{2}\cos\theta) \\ \frac{\sqrt{2}}{3}(\sin 2\theta - t\sqrt{2}\sin\theta) \\ 1 \end{array}\right. \qquad t\in\mathbb{R} \qquad \theta\in[0,\pi]$$

celle de la surface de Boy correspondante est bien une immersion et vaut

$$\left|\begin{array}{l} X \\ Y \\ Z \end{array}\right. = (1-\sqrt{2}t\sin 3\theta + t^2)^{-1}.\left|\begin{array}{l} \frac{\sqrt{2}}{3}(\cos 2\theta + t\sqrt{2}\cos\theta) \\ \frac{\sqrt{2}}{3}(\sin 2\theta - t\sqrt{2}\sin\theta) \\ 1 \end{array}\right.$$

Tous les calculs sont élémentaires et le degré de liberté subsistant dans le paramètre de déformation des ellipses est fixé pour des raisons essen--tiellement esthétiques (fig 13).

$6^{ème}$ étape

L'élimination de θ et t dans la paramétrisation précédente, bien que fastidieuse, peut être entièrement menée sans le secours d'une machine et conduit à l'équation explicite du sixième degré suivante:

$$64(X_0-X_3)^3X_3^3-48(X_0-X_3)^2X_3^2(3X_1^2+3X_2^2+2X_3^2)+12(X_0-X_3)X_3(27(X_1^2+X_2^2)^2-24X_3^2(X_1^2+X_2^2)+36\sqrt{2}X_2X_3(X_2^2-3X_1^2)+4X_3^4)+(9X_1^2+9X_2^2-2X_3^2)(-81(X_1^2+X_2^2)^2-72X_3^2(X_1^2+X_2^2)+108\sqrt{2}X_1X_3(X_1^2-3X_2^2)+4X_3^4)=0$$

On prouve alors que l'ensemble des zéros de ce polynôme est exactement le modèle donné par la paramétrisation.

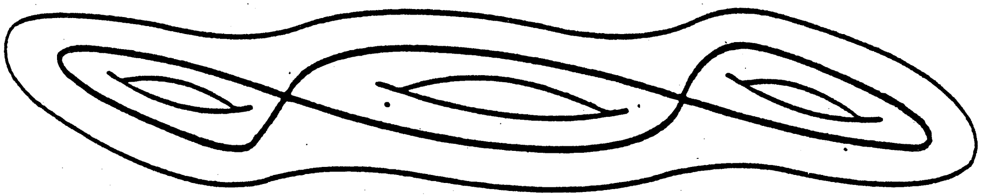
$7^{ème}$ étape

Il est maintenant possible de déterminer la courbe d'auto-in--tersection de la surface de Boy qui se révèle être une sextique gauche du type hélice tripale à pas positif. L'emploi du langage Macsyma permet de vérifier que les nappes de la surface se coupent bien transversalement le long de cette hélice tripale, les calculs n'étant pas praticables sans ce secours. Il reste que l'on peut construire l'image réciproque de la courbe d'auto-intersection par la paramétrisation et constater que chaque pale de l'hélice borde bien un disque. Nous avons la surface de Boy comme ensemble des zéros d'un polynôme du sixième degré.

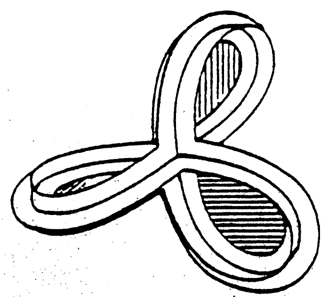
$8^{ème}$ étape

La construction précédente est obtenue par déformation de la surface romaine de Steiner contenant six singularités du type bonnet croisé. L'aide du langage Macsyma est précieuse pour montrer que la déformation décrite réalise le passage de la surface de Steiner admettant six singularités stables, à la surface de Boy sans singularités , par l'intermédiaire de trois singularités génériques (au sens des familles à un paramètre de surfaces), en l'occurrence de confluences hyperboliques de bonnets croisés par paires. L'emploi de moyens graphiques traitant des objets géométriques à trois di--mensions autorise la visualisation de cette confluence.

Cette étude illustre les services que peut rendre l'informatique dans le domaine de la géométrie des surfaces de l'espace, aussi bien pour la visualisation que pour atteindre des résultats mathématiques exigeant des calculs surhumains.

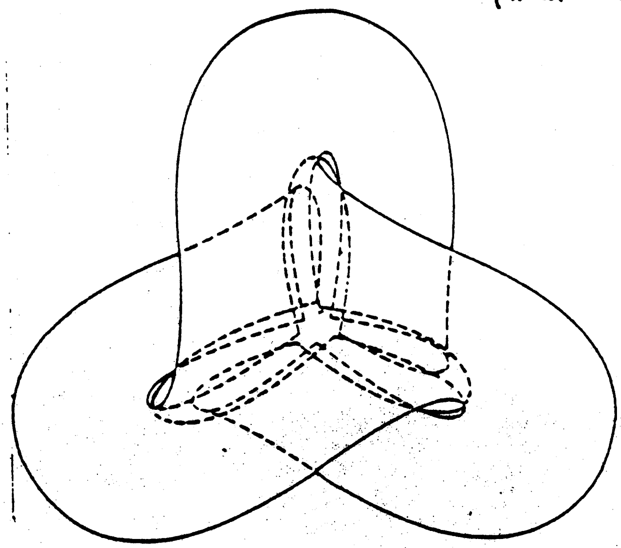


① Courbe algébrique $x = 3\cos t$, $y = \sin 3t$ (trait gras)
Surface algébrique $729\,y^2 + (x^2-9)(4x^2-9)^2 + z^2 - a^2 = 0$ (trait maigre)



Voisinage de l'hélice tripale à pas positif dans la surface de Boy.

②

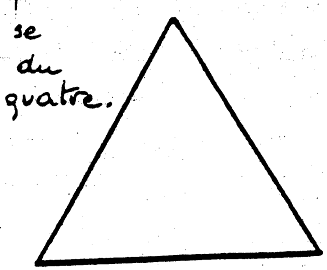


Surface de Boy obtenue en bouchant les trois pales de l'hélice (fig. 2) par trois disques et en recollant un disque au voisinage de l'hélice le long de leur bord commun restant.

③

Triangle équilatéral revêtu quatre fois aux sommets duquel se trouvent trois singularités du type mouchoir plié en quatre.

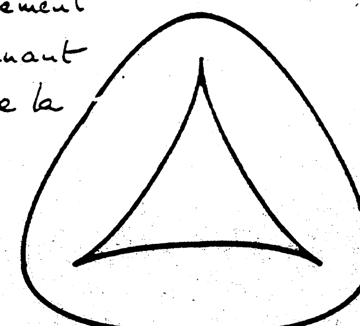
④



↑ suite

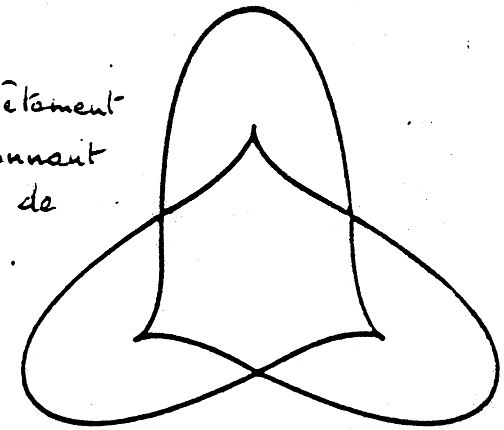
Perturbation du revêtement de la figure 4 donnant le contour apparent de la surface romaine de Steiner.

⑤

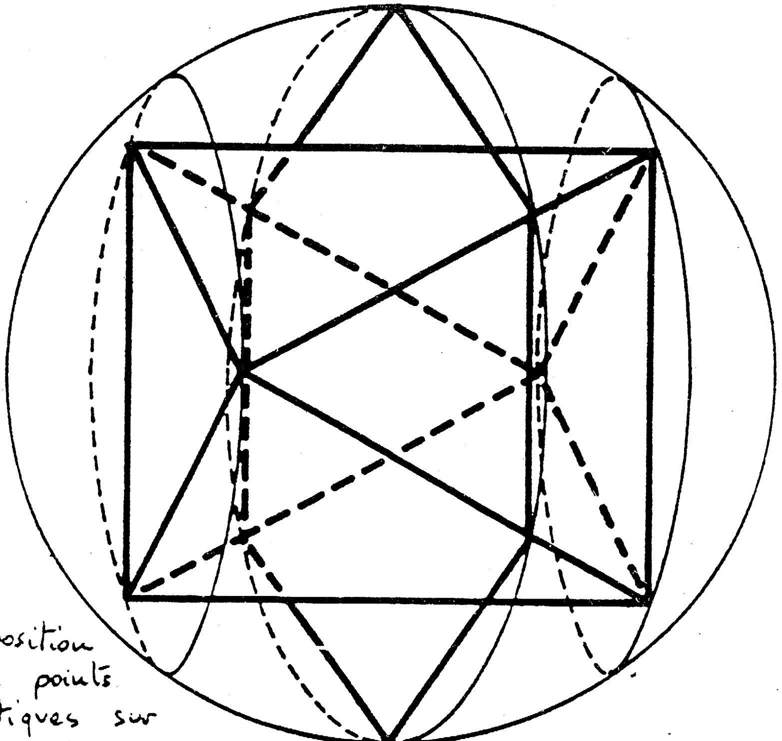


Perturbation du revêtement de la figure 4 donnant le contour apparent de la surface de Boy.

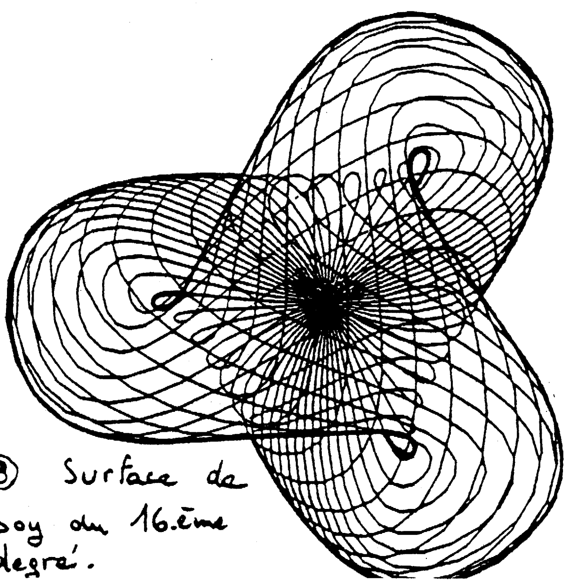
⑥



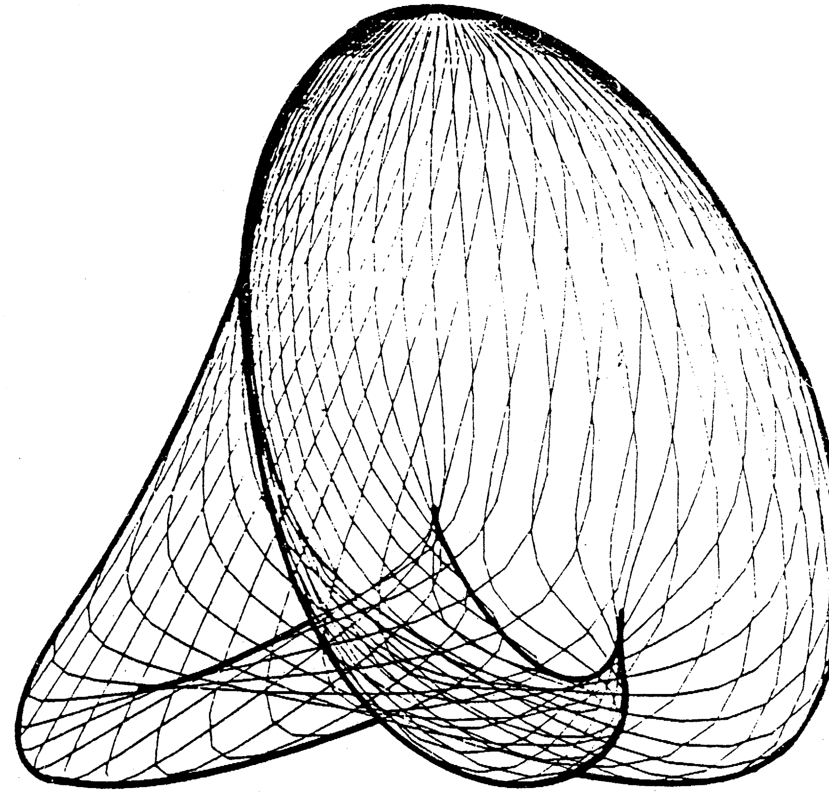
7



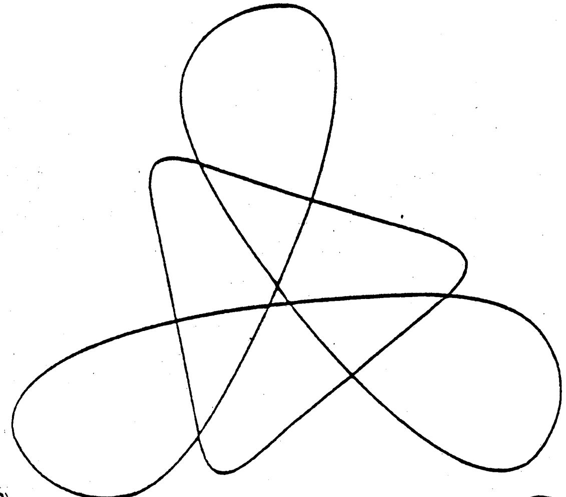
Disposition des points critiques sur la sphère.



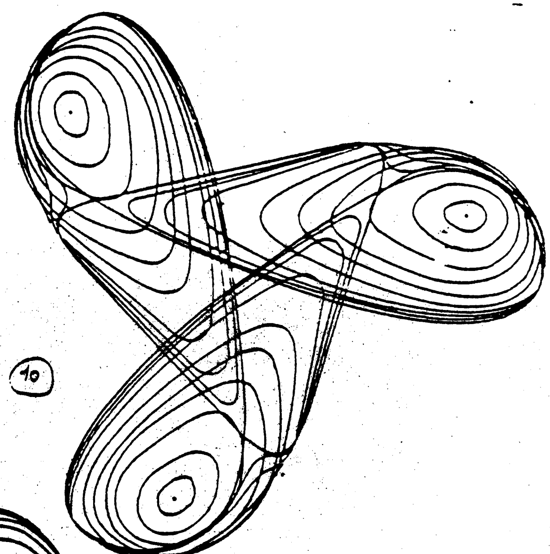
8 Surface de Boy du 16.ème degré.



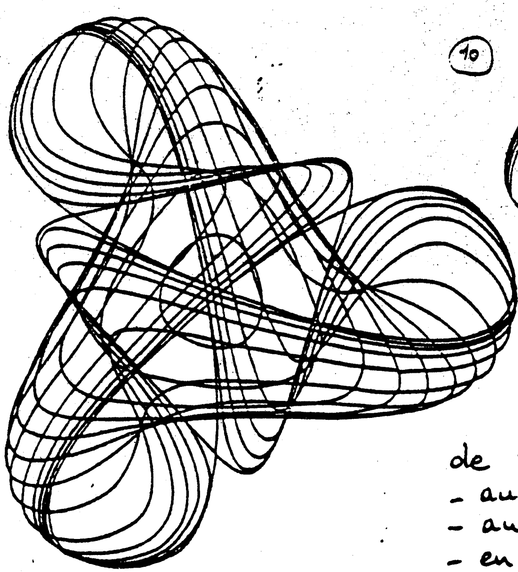
9 Surface de Boy du 16-ème degré.



(10)



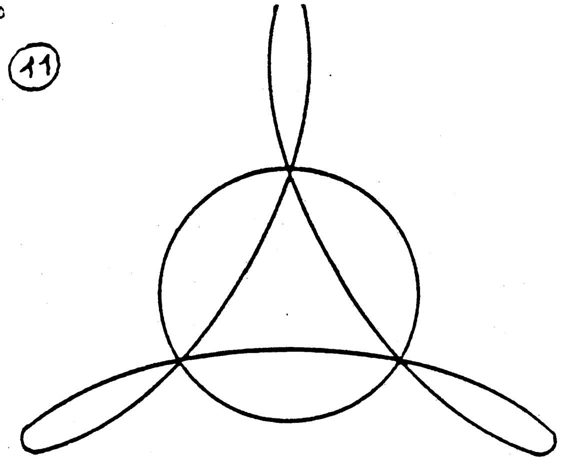
(10)



Sections planes de la surface des figures 8 et 9 par des plans perpendiculaires à l'axe de symétrie ternaire :
- au dessus
- au niveau
- en dessous

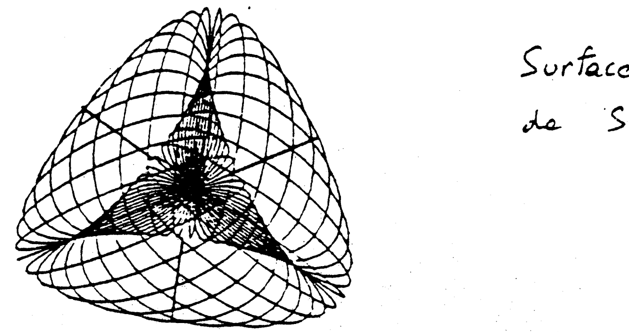
} du plan des cols.

(11)



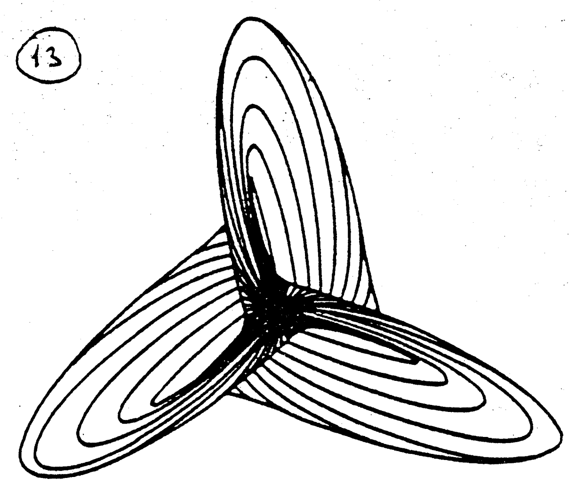
Cercle et hypocycloïde à trois rebroussements allongée.

(12)



Surface romaine de Steiner

(13)



Surface de Boy du 6-ème degré

VISUALIZING CALCULUS CONCEPTS USING A COMPUTER

David Tall

Mathematics Education Research Centre
University of Warwick
COVENTRY CV4 7AL

Computers are rapidly providing facilities for complex calculations and symbolic manipulation, but these often give only the *results* of the processes without displaying the processes themselves. It will fall upon mathematics educators to provide ways of developing an understanding of underlying mathematical processes so that the results may be used with greater insight.

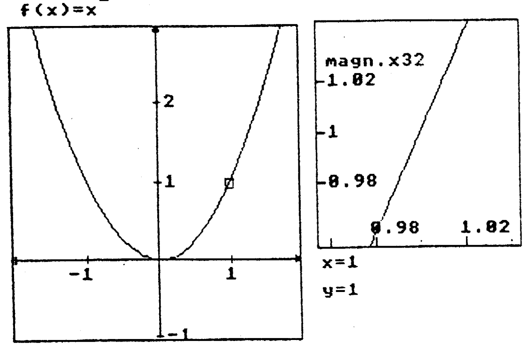
This paper reports an approach to the learning of the calculus [1] using programs to promote understanding of the fundamental concepts. The first four sections contain descriptions of the programs and the rationale underlying their design. Section 5 describes some of the reactions encountered in schools in which they have been tested and the final section discusses implications for the ICMI study. Experience shows that it is not simply a matter of re-sequencing current mathematical concepts to fit with the computer; the facilities of the computer demand a fundamental re-working of the mathematical theory itself.

1. Differentiation

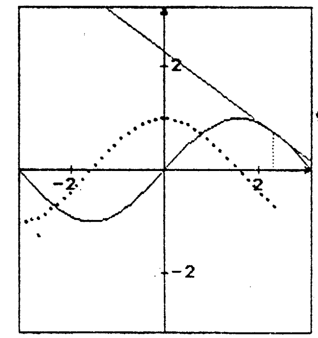
Traditionally the foundation of the calculus is the notion of a limit, either of a chord approaching a tangent or algebraically as a ratio

$$\frac{f(x+h)-f(x)}{h} \qquad (1)$$

as h tends to zero. The computer brings a new possibility to the fore. Instead of viewing the idea of differentiation as calculating the gradient of the tangent to a graph, we may begin by considering *the gradient of the graph itself*. Although a graph may be curved, under high magnification a small part of the graph may magnify to look like a segment of a straight line. In such a case we may speak of the gradient of the graph as being the gradient of this magnified (approximately straight) portion. Figure 1 exhibits the graph of $y=x^2$ near $x=1$ approximating to a straight line segment of gradient 2.



To represent the changing gradient of a graph, it is a simple matter to calculate the expression (1) for fixed h as x varies. The program GRADIENT includes a routine that moves in steps along the graph drawing the chord through the points x, x+h on the graph and plotting the gradient of the chord as it proceeds. Figure 2 shows the gradient of the graph of sinx being built up. It clearly approximates to cosx; by superimposing the graph of cosx for comparison the gradient function may be investigated experimentally before any of the traditional formalities are introduced.



One product of this type of investigation is that it doesn't require a very small value of h to get a good computer picture of the gradient. It leads to the question: why bother to take limits at all? The answer is given by investigating a graph such as $f(x)=1/x$ which has a disconnected domain of definition. This graph clearly has negative gradient everywhere, but any fixed value of h gives values of x and x+h on either side of the origin whose chord has *positive* gradient. Thus the need to take limits arises from a purely practical consideration of handling functions whose domains are disconnected, to make sure that the gradient is only calculated between points on the same connected component. This leads naturally into the formal consideration of limits and the development of the formulae for calculus.

In developing the formulae, the symbols dx and dy can be given a meaning, dx being an increment in x and dy the corresponding increment in y, not to the graph, but to the *tangent* to the graph. Better still, one may view (dx,dy) as a *vector* representing the direction of the tangent, a valuable idea when we come to look at the meaning of differential equations.

There are other bonuses. The program naturally copes with positive or negative values of h in the formula (1) and pictures are often drawn with negative gradients instead of the traditional graph of an increasing function drawn in the majority of text-books. The moving graphics give a dynamic interpretation of the changing gradient, which in turn helps to expand the students' mental image of the concept.

Intuitive approaches to the calculus usually explain what a derivative is, without saying what it isn't. With the computer graphic approach it is easy to show what a non-differentiable function looks like. A function is not differentiable at a point if its graph near the point doesn't magnify to look straight. For instance a function may have different left and right derivatives, which simply means that the magnified graph looks like two straight half-lines meeting at an angle. The left and right derivatives at the point are the gradients of the corresponding lines. Simple examples include $|x^2-1|$ at $x=\pm1$ and $|\sin x|$ at multiples of π.

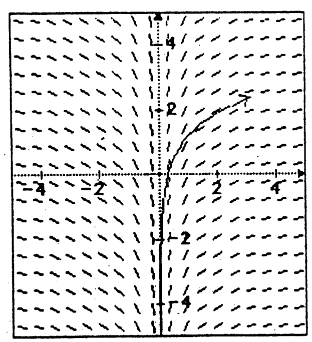
It is possible to draw a function everywhere continuous and nowhere differentiable (a concept previously beyond the reach of elementary calculus): its graph is so wrinkled that it never magnifies to look straight. The program BLANCMANGE draws such a graph and allows investigation of its properties.

2. Integration

Integration involves two entirely separate concepts:
anti-differentiation and summation processes such as finding the area under a graph.

Anti-differentiation is usually viewed as the reversal of the process of handling the formulae for differentiation and is largely seen by students as a problem-solving exercise in manipulating formulae. Graphically it may be characterized as knowing the gradient dy/dx=f(x) of a graph and requiring to find a graph y=I(x) fitting this information. The program ANTIDERIVATIVE draws short line segments through an array of points (x,y) with the gradient f(x). A solution y=I(x) is simply traced out by following the direction of the lines. (Figure 3.) As the gradient direction is a function of x alone, the solution curves clearly differ by a constant. The program draws solutions numerically using a step along the graph rather than a fixed x-step. In doing so it remains on a connected component of a solution. When solving the equation dy/dx=1/x, a solution curve starting to the right of the origin always remains on the right. Thus the fact that two antiderivatives differ by a constant is seen only to be true over a *connected component* of the domain, a considerable advance on the limited view in most elementary courses where the antiderivative is given in the form I(x)+c for an "arbitrary constant" c, without mention of any restriction on the nature of the domain.

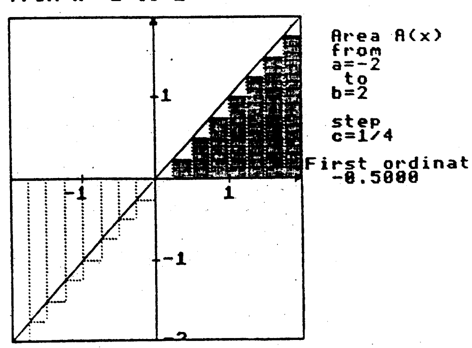
f(x)=1/x



The program AREA calculates the area between the graph and the x-axis by a variety of methods. The numerical values are displayed and each part of the area is drawn using different colours for positive and negative results. Students can see that a positive step gives a positive result when the graph is above the axis and negative when below. (Figure 4.) They can see equally well that a negative step reverses the signs, a concept traditionally regarded as difficult yet clearly represented by moving graphics.

f(x)=x

from x=-2 to 2



The fundamental theorem, that the area function is an antiderivative of the original function, can be demonstrated graphically in a neat way. If A(x) is the area under the graph y=f(x) from a fixed point c to the variable point x, the area from x to x+h is approximately A(x+h)-A(x).

The fundamental theorem depends on the fact that

$$A(x+h)-A(x) \doteq f(x)h \quad \text{for small } h,$$

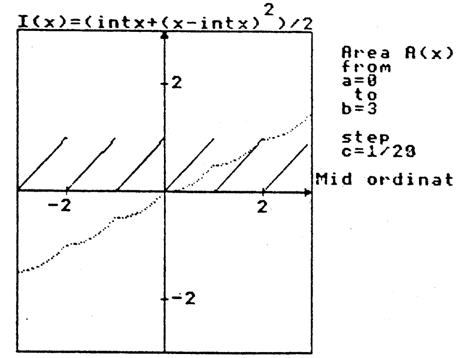
with the approximation getting better as h tends to zero.

Graphically this may be represented by stretching the x-range and leaving the y-range at a normal scale. Where f(x) is continuous this pulls out a small part of the graph approximately flat, giving a rectangle of approximate height f(x) and width h.

This is the natural place for questions of continuity to arise. Continuity is largely irrelevant in differentiation (where it is an automatic property of a differentiable function), but it arises as a separate consideration in integration. The continuity of f(x) is essential for the area function A(x) to be differentiable and satisfy A'(x)=f(x). But certain discontinuous functions have continuous area functions which are not differentiable at the points where the original function is discontinuous. This concept is fairly subtle when approached theoretically. But a pictorial representation gives a striking insight. Figure 5 draws the cumulative area function for f(x)=x-INTx as a sequence of dots. The area function is visibly continuous, but is not differentiable at the integer points. Here the area graph has "corners" and does not magnify to look straight.

f(x)=x-intx

$I(x)=(intx+(x-intx)^2)/2$



3. First Order Differential Equations

In most preliminary courses the study of differential equations is no more than a rag-bag of isolated techniques for solving specific equations which happen to be amenable o a particular approach: separable, exact, homogeneous, linear with constant coefficients, and so on.

A computer-drawing approach offers a much more comprehensive view of the process of solution. A linear first order differential equation:

$$dy/dx=f(x,y)$$

is simply an extension of the antidifferentiation program mentioned earlier. At each point (x,y) in the plane we know the gradient of the required solution curve, namely $f(x,y)$. The problem is to draw a curve which everywhere has this gradient. The naive solution is to draw a direction field with short line segments having the direction $f(x,y)$ calculated at the mid-point (x,y). The solution is to trace a curve following through the direction field. (Figure 6.) This is done numerically and may be investigated parallel to a consideration of the numerical methods involved. The picture itself powerfully suggests ideas about the nature of the solution. For instance the differential equation

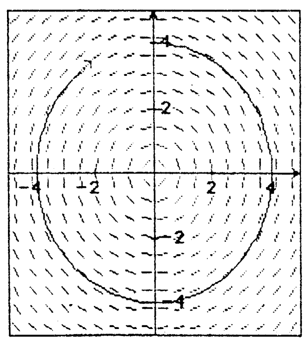
$$y\frac{dy}{dx} = -x$$

does not have a global solution as a function $y=f(x)$, it has *implicit* solutions

$$x^2+y^2=\text{constant}$$

which are circles centre the origin. At points where the circles meet the x-axis the tangents are vertical. Thus the normal interpretation of dy/dx as a derivative function is inappropriate, but the interpretation as a vector direction (dx,dy) allows $dx=0$ with dy non-zero. With the graphical interpretation it is much easier to see a first order differential equation as one giving information about the direction of the tangent (dx,dy).

dy/dx=-x/y



It transpires that several of the statements made about differential equations in elementary text-books are erroneous. It may happen that following the direction field (as in the case $dy/dx=1/x$ mentioned above) stays in a certain region of the plane. Thus the solution, and the perennial "arbitrary constant" is only relevant in this region. A global solution (for $x\neq 0$) could be $\log|x|+c$ for $x<0$ and $\log|x|+k$ for $x>0$ where k and c are different. The oversimplified statement that an "nth order differential equation has n arbitrary constants" may be seen in a more appropriate light.

To trace out a unique solution, numerically or theoretically, requires that the differential equation specifies a value of dy/dx at every point along the solution curve. The equation

$$x\frac{dy}{dx} = 3y$$

may be solved by separation of the variables as:

$$y=kx^3.$$

Every solution curve passes through the origin where the direction dy/dx is not specified. A perfectly legal solution is to have a different value of k on either side of the origin.

A combination of graphical and numerical solutions of first order differential equations gives powerful insights into the theory, complementing the isolated analytical approaches and exposing the weaknesses in the mathematics in the current curriculum.

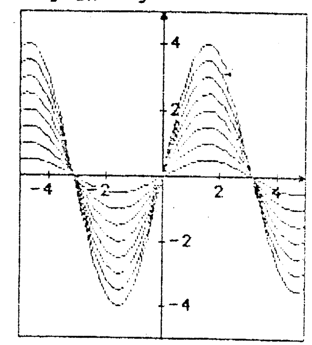
4. Higher Order Differential Equations

It might be thought that the direction field in first order equations is a special case. For a second order differential equation the theory seems different. Through every point in the plane there are an infinite number of solutions. Figure 7 draws some of the solutions of the equation

$$d^2y/dx^2 = -x$$

through the origin.

d²y/dx²=-y



For each starting direction from a given point there is a unique solution.

Solutions of such equations are often attacked by introducing a new variable,

$$v=dy/dx$$

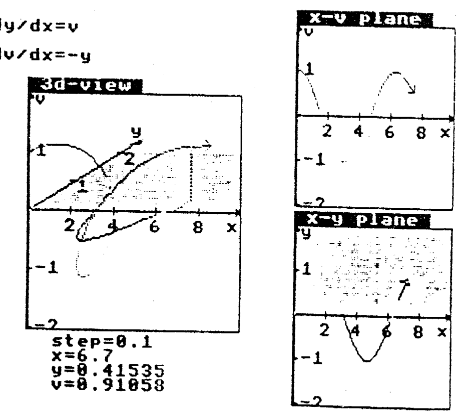
giving two linear equations:

$$dy/dx=v$$
$$dv/dx=-x.$$

Thinking of this system as having one independent variable x and two dependent variables y,v then in (x,y,v) space the two linear equations again give a tangent (dx,dy,dv) in the direction $(1,v,-x)$. Thus there is a direction field, but it is in three dimensions not two. It is possible to draw a representation of the three-dimensional solution and update appropriate coordinate planes at the same time. (Figure 8.) Alternatively the three-dimensional picture may be replaced by the third coordinate plane (the "phase plane").

dy/dx=v

dv/dx=-y



In this way the theory of ordinary differential equations may be given a unified meaning that enriches and complements the collection of isolated analytic techniques.

5. Changes in learning style and modes of operation

The programs described in this paper are at present being trialled in a group of five schools involving some four hundred pupils in experimental and control groups; they have also been used in experiments with mature students on polytechnic courses and science students learning the calculus at university.

They are powerful general-purpose utilities rather than self-contained programmed learning and are structured for a wide range of uses, from teacher demonstration (with facilities to slow down or stop the action where necessary) to student investigation. Enlightened teacher demonstration can easily involve dialogue with the students rather than a straight lecture-presentation. For example, when the derivative of x^n is introduced, a program may be used to draw the gradient function in various special cases. Students can *investigate* the pattern for $n=0,1,2,3$, *conjecture* the general formula for x^n, then *test* the formula for various values of n, such as $n=4,5$ or $n=-1,-2$, $1/2$, π and so on, before going on to *prove* the formula algebraically for certain values of n. Often the calculus is introduced to students at a stage when the proof of the general formula is beyond them, but this does not limit their imagination in suggesting values for testing which would be far beyond their ability for algebraic manipulation, such as $n=33.5$ or $-7/2$. In drawing the graphs in such cases they begin to appreciate the range of values for which the formulae are valid, a factor often sadly lacking in blind algebraic manipulation.

Students are quite capable of producing valuable results in extended investigations. In less than an hour a small group of 16 year old mathematics students fresh to the calculus discovered the derivative of x^n, tested it for negative and fractional powers, conjectured the correct derivatives of sinx and cosx and, by trial and error, found that the graph of k^x and its derivative were the same for a value of k between 2.7 and 2.8. When challenged further they discovered the derivatives of ln(abs(x)) and tanx through quite unexpected channels of deduction and eagerly investigated given functions with problematic left and right derivatives at certain points. When left to their own devices they went on to study functions of their own choice, eventually admitting defeat with the derivative of $(1-x^2)^{1/2}$. At this stage they grew bored with random investigations and were ready for a more structured study of the formulae for differentiation.

In this way student investigations lend themselves to being embedded in a structured curriculum. The legacy of such investigations is an enriched intuition in the students which may be built upon at a later stage. By careful thought it is often possible to recast analytic proofs in such a way that they become both intuitive *and* rigorous.

6. Implications for the ICMI study

The experiences in developing the programs lead to the following responses to the main questions in the ICMI discussion document [2]:

1. The programs described here are the product of a single individual and have been piloted in a number of schools and made available commercially. They are not, as yet, part of an experimental curriculum and are currently being used to support the existing syllabuses. In Britain there are major changes taking place in school curricula across a wide range of subjects. The Cockroft Report [3] has suggested enlightened changes in content and approach, but an accident of history caused it to be written at a time when microcomputers were only just appearing in schools. The core curriculum leading to 16+ does not include the use of computers, though these may beintroduced in experimental syllabuses. Mathematical projects, such as SMILE based in the Inner London Education Authority, already use the computer for mathematical investigations but there is a lack of good software, especially for older age groups.

2. There is increasing pressure on science students at university to become more "computerate", leading to the introduction of "hands-on" experience in "Mathematics for Science" courses. The calculus programs in this article have been used in mathematics courses for biology students; their practical approach proved a valuable complement to the traditional "methods" course.

3. The philosophy behind the development of the calculus programs is highly practical, with pure mathematics revitalized by geometric intuition and numerical methods.

4. The nature of the symbolic mathematics in the calculus will clearly need to be re-considered. It is one thing to learn the formulae for the calculus, it is quite another to fully understand the mechanics of the way a computer implements these in programs such as MUMATH. The understanding of the latter is unnecessary for the majority, provided they know the principles on which the formulae are deduced. A long-term practical approach would be to introduce simple numerical algorithms for the processes in the calculus using short programs written and modified by the students themselves, coupled with prepared software to give graphical representations of the results. A combination of theory and experimental investigation as described in earlier sections could then lead on to essential ideas in the theory.

5. The relevant question to ask in this paper concerns the balance between discrete mathematics and the continous mathematics of the calculus. A computer simulation of the latter essentially uses finite differences, blurring the distinction between continuous and discrete. Basic ideas of rates of change (differentiation) and growth (integration) will still remain central in mathematics and in many applications. A valuable case could be made for clearly contrasting the theoretical world of the calculus with the practical world it represents. Students are likely to gain considerable insight through the distinction between a practical limit and a theoretical one, or a practical tangent (drawn on a computer between two close points on the graph) and a theoretical one (touching the graph).

6. The computer stimulates changes in the order of presentation. In the present work two areas arose which would have been useful precursors to the calculus. First, it would have been of great assistance to know of vectors before embarking on discussion of the tangent direction. (Vector directions would avoid the anomaly of the vertical tangent.) Second, and of far greater import, a greater computer awareness would have allowed more short algorithms to have been programmed by the students themselves instead of relying mainly on software. Thus the *mathematical* aspects of computing should be taken into the early part of the mathematics syllabus and not left purely as part of computer science.

7. It is clear that the computer allows understanding of higher order concepts in a more immediate and appealing way, but this involves a very different kind of mental imagery from that afforded by

traditional approaches. The computer will allow many mathematical topics to be introduced earlier in university courses, but educators need to research precisely what is being learnt in such courses. In the case of the calculus one of the most striking changes is the practical ability to carry out numerical methods on a computer. Procedural computer languages will allow these to be performed in a more meaningful way.

8. The calculus programs in this paper already show several ways in which the topic may be presented in a first course. Later in mathematical analysis an algorithmic approach can be made to many theorems, such as the Intermediate Value Theorem. Now that powerful algorithms for finding solutions can actually be implemented, they will naturally take the place of theoretical existence statements where appropriate. But there is a difference between theoretical existence by a contradiction proof and practical existence through computation which will need more circumspect treatment.

9. The $64000 question is what to leave out. The developments in this paper have been *added* to a traditional course rather than replacing parts of it. In practice some areas move faster as a result of insights into the fundamental ideas, but overall the course would take a little longer to achieve more. It has been suggested that the arrival of programs such as MUMATH to do the symbolic manipulations will lessen the need for fluency in formal techniques of differentiation and integration. However, the programs which carry out these manipulations tend to require half a megabyte of memory or more and will not be generally available to individual users for several years. In the meantime, the consequences of deleting such material from the curriculum should be studied carefully.

10. Teachers of mathematics do not need to study the whole of computing. They need to be trained to cope with those parts of computing relevant to mathematics teaching, including mathematical aspects of programming (particularly in mathematical algorithms) and the use of appropriate software in the development of mathematical concepts.

THe ICMI study could assist greatly by moving towards a consensus as to what constitutes the core of "computing relevant to mathematics".

References

1. David Tall *Graphic Calculus for the BBC computer* Shiva Publications 1985

2. *The influence of Computers and Informatics on Mathematics and its Teaching*. An ICMI discussion document 1984

3. W. H. Cockroft *Mathematics Counts* H.M.S.O. 1982

COMPUTER GRAPHICS AS AN ESSENTIAL RESEARCH TOOL IN THE ITERATION OF RATIONAL FUNCTIONS

Beverly West
Department of Mathematics
Cornell University, Ithaca, NY USA

A discussion on the influence of computers on the teaching of mathematics must consider today's role of the computer in mathematical research and the impact this makes on students. I should like to explain a bit about the current research of John Hubbard at Cornell, with his colleagues Adrien Douady of Paris and Bodil Branner of Copenhagen. Their work in iterating rational functions demonstrates an essential use of computer graphics. Furthermore, the results are visually very exciting, and provide a rare instance where even secondary school and beginning university students can be captivated and given a glimpse of the action in current mathematical research.

The history is fascinating. The basic theorems were all proved roughly 1905-1920 by two French mathematicians, Pierre Fatou and Gaston Julia. Then essentially nothing happened until the very late 1970's when computer graphics allowed us to get a look at these things. The subject is now exploding in all directions.

Iteration means performing an operation over and over and over again, always using the result of the previous computation as input for the next. Any quadratic polynomial can be written, up to a change of variable, as $z^2 + c$. To iterate $z^2 + c$, start at some point z_0, called the "seed", and form the sequence

$$z_0^2 + c = z_1, \quad z_1^2 + c = z_2, \quad z_2^2 + c = z_3, \ldots,$$

where both z and c may be complex numbers. We are interested in whether this sequence tends to infinity, which it will certainly do if $|z_0|$ is sufficiently large. In the complex z-plane a filled-in Julia set K_c is defined to be the set of points that do not iterate to infinity:

$$K_c \equiv \{z \mid (z^2 + c)^{\circ n} \not\to \infty\}.$$

If $c = 0$, then the Julia set K_0 is the unit disk.
If $c = -2$, the Julia set K_{-2} is the interval $[-2,2]$.

But for all other c the Julia sets are much more complicated, as shown in Figure 1, and Fatou and Julia never saw them.

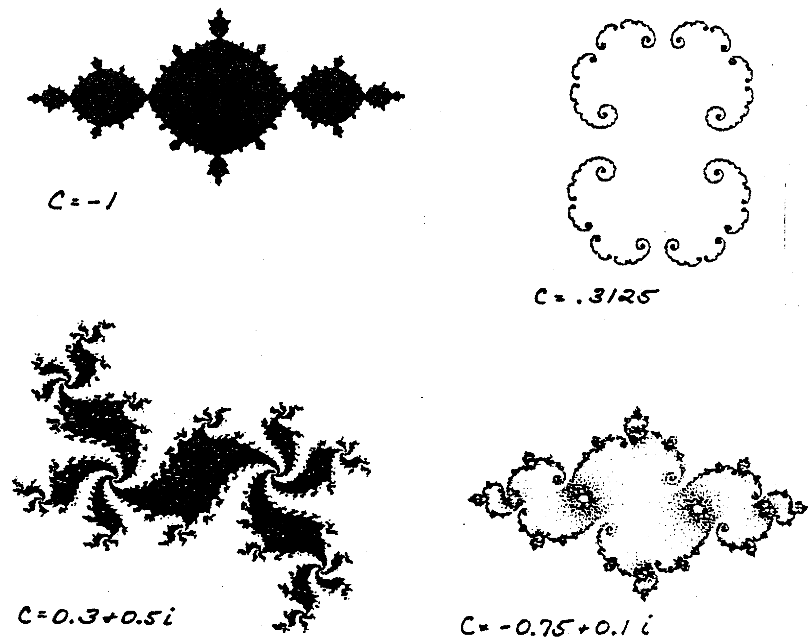


Figure 1

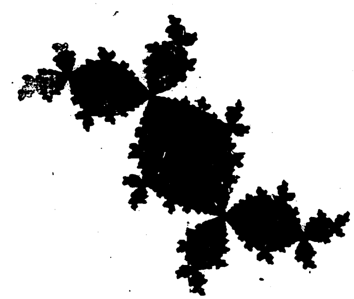
These pictures were made by a scanning program (on an array processor) as follows: The computer screen gives a grid of 400x400 points representing the square $[-2,2]\times[-2,2]$. For each picture c is fixed as labelled. Then each point of the grid is used as a seed z_0 to construct the iteration sequence for $z^2 + c$, and the point is colored according to whether the iteration is ***bounded*** ($\not\to \infty$ and thus belonging to K_c) or ***unbounded*** ($\to \infty$ and thus outside K_c). A fortuitous theorem assures us that if $|z_n| > 2$, then $\lim_{n\to\infty} z_n \to \infty$, so we have a practical test for "approaching infinity". Thus, if for some n, $|z_n| > 2$, the point z_0 is given the background color; if after several hundred iterations this has not happened, the point z_0 is given the foreground color.

As is already apparent in Figure 1, this most simple iteration leads to amazing complexity, and you will notice that some of the K_c are connected and some are disconnected. Long before we had these pictures, Fatou and Julia proved the following theorems:

K_c <u>is connected if and only if</u> $0 \in K_c$.

<u>If</u> $0 \notin K_c$, <u>then</u> K_c <u>is a Cantor set</u>.

Today we can illustrate these theorems with computer graphics showing inverse images of a closed curve surrounding K_c, forming a nested sequence of sets converging on K_c. Figure 2 shows the result for a connected K_c, and Figure 3 for a disconnected K_c. (In the latter case, the split occurs when a "disk" misses c, because its inverse image must miss 0, the critical point of $z^2 + c$. From covering space theory, a disk that does not include the singularity has two distinct preimages; each of those will have two distinct preimages, and so on, so the result is a totally disconnected Cantor set.)



$C = -0.2+0.75i$

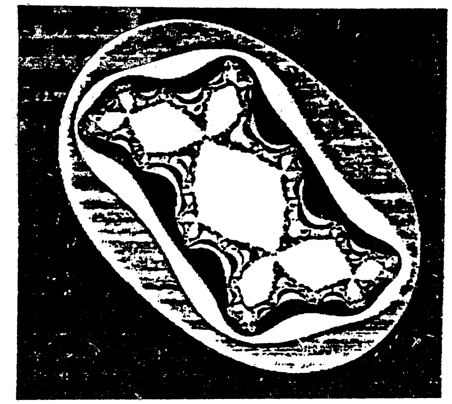
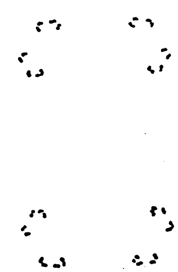


Figure 2



$C = 0.675$

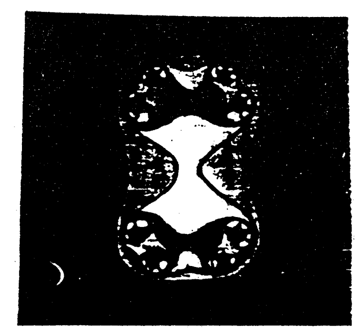


Figure 3

The question of which values of c give rise to which kinds of Julia sets was not easy to answer; the pattern seemed to be very complicated. In the late 1970's, Benoit Mandelbrot of IBM's Thomas Watson Research Center looked in the parameter space of the complex constant c and plotted a picture of those values of c that give rise to a finite cycle in the dynamics. The resulting set M is called the Mandelbrot set, and it has the property that

$$M \equiv \{c \mid K_c \text{ is connected}\} .$$

The Mandelbrot set is shown in Figure 4; it is computed by a scanning program exactly as for the Julia sets, except that in this case every iteration begins with 0 and has a different value of c. Now we see that there *is* a pattern for the question of which values of c lead to connected K_c's, although it is a very complicated pattern.

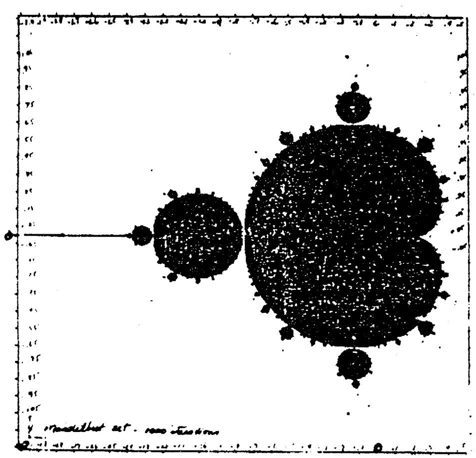


Figure 4

The boundary of the Mandelbrot set is wildly fractal, as shown by zooming in on one of the "necks" in Figure 5. The closer we get to the boundary, the more detail emerges.

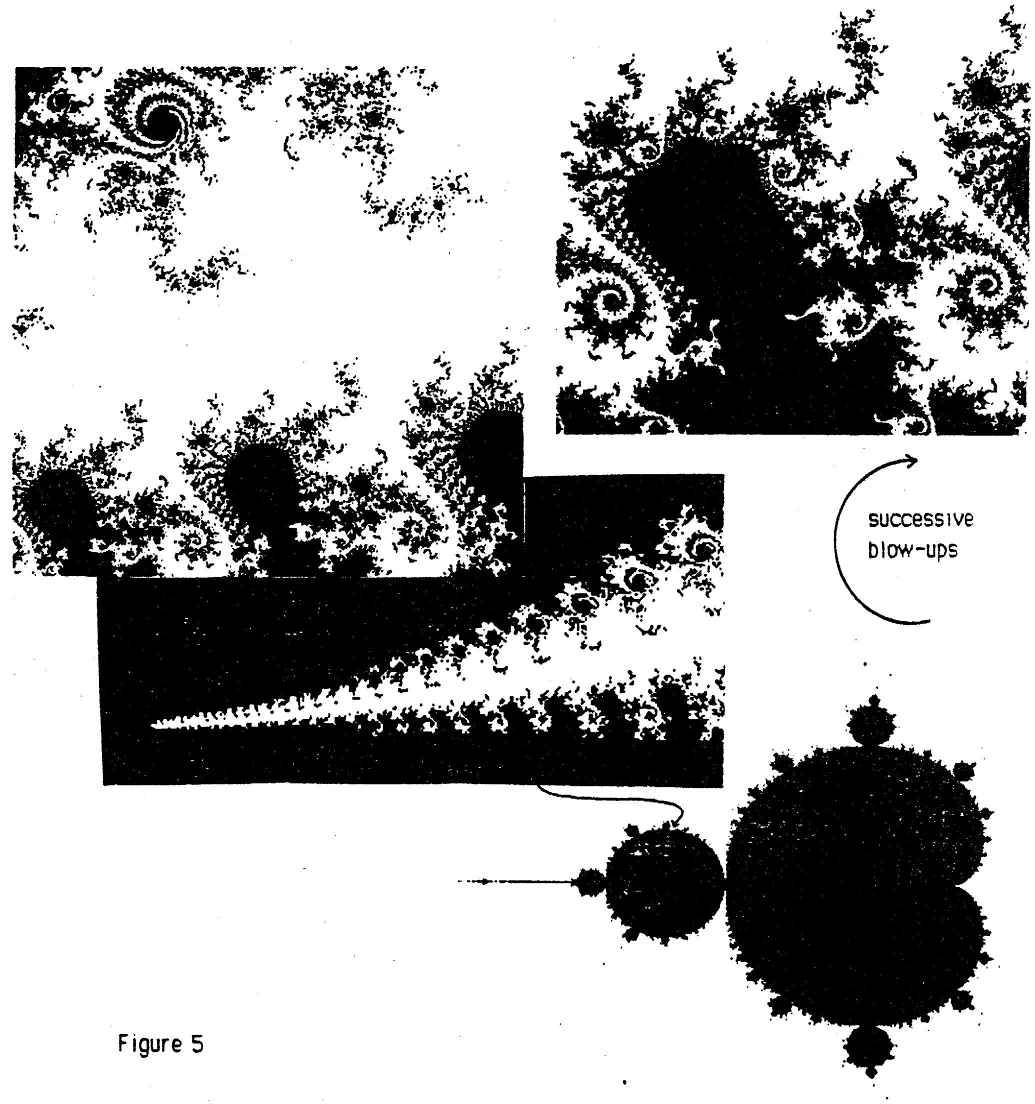
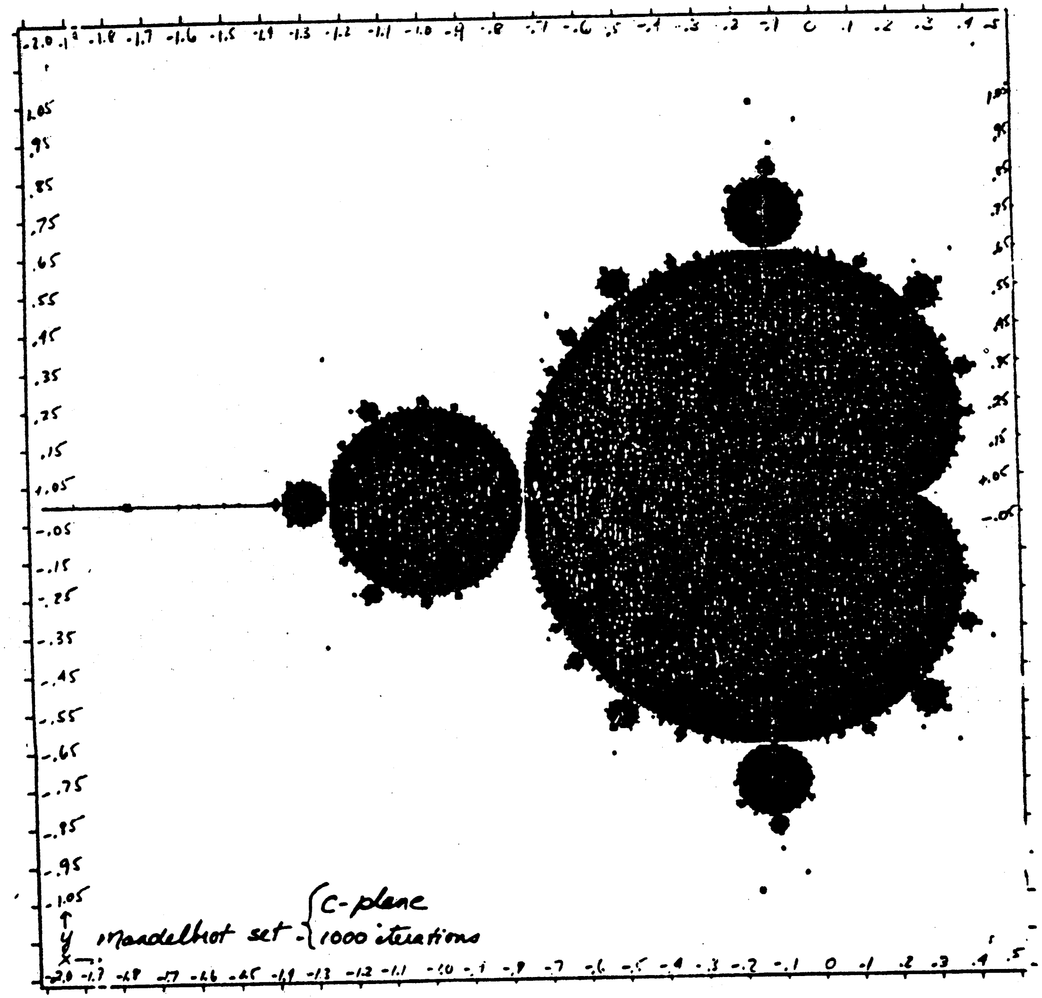


Figure 5

Iterating Rational Functions



Choose a c in the black areas & get a connected Julia set (bumps & dots give most interesting ones)

There seems to be a sort of self-similarity, and indeed, in Figure 6 you can see that what first appeared as little dust spots above the original Mandelbrot set (Figure 4) enlarge to look like tiny copies. Such little copies also appear all along the "antenna" of M at the left.

Figure 6

But, in fact, on closer examination it turns out that the Mandelbrot set is totally <u>not</u> self-similar, that each "copy" has its own distinctive signature in the fine detail. One of the best ways to see this is to plot the Mandelbrot set with contours showing how soon the iterations reach 2 in absolute value; the pattern of contours around each copy of M is entirely different from the pattern of contours around the whole Mandelbrot set, as shown in Figure 7.

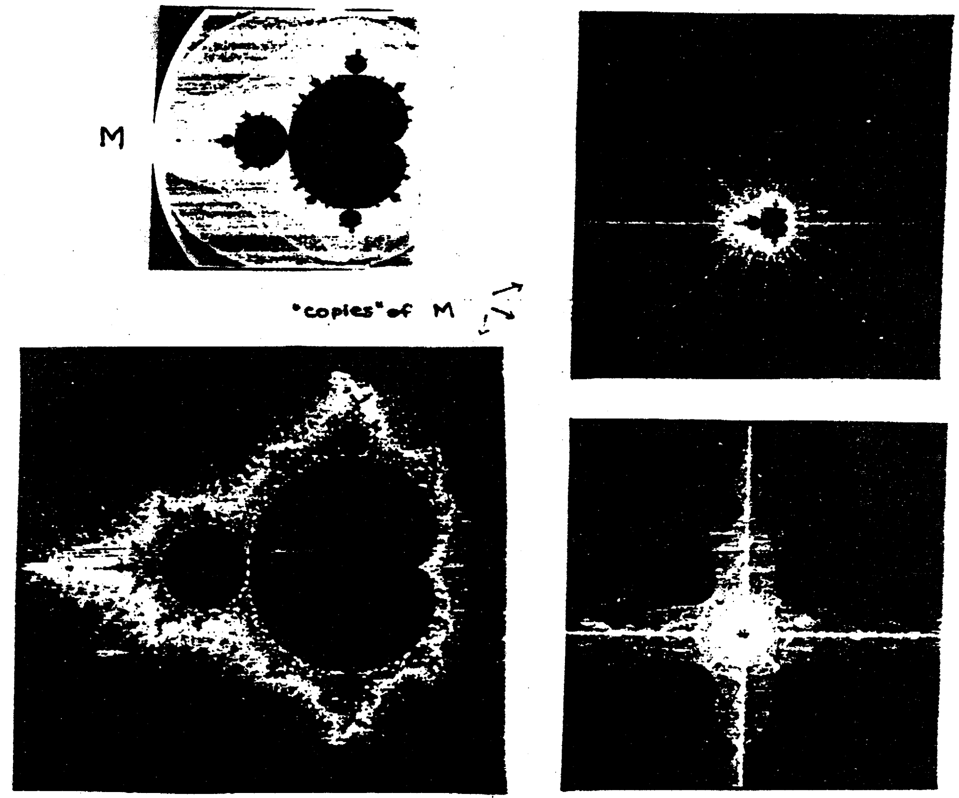


Figure 7

More of the complications illustrated by these contours are shown in Figure 8, which are successive blow-ups of one of the tiny necks in the Mandelbrot set, resulting in the last photo as yet another emerging copy of the Mandelbrot set. (These pictures should be seen in color to be best appreciated; the contours range from red for points which first reach two in absolute value, through yellows to blues -- the Mandelbrot set itself remains black, but in the xerox, the red also appears as black.)

successive blow-ups of region indicated by arrow
in first picture at upper left

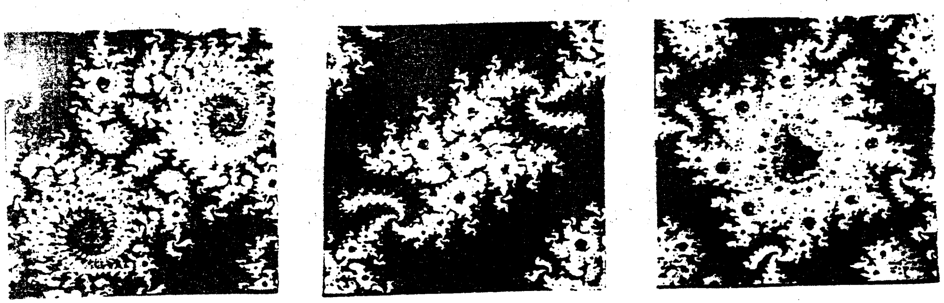


Figure 8

Another amazing property of the Mandelbrot set is that it is simply connected, a theorem first proved by Hubbard and Douady; the little "dust spots" are actually connected by tiny threads to the main Mandelbrot set. These threads are wispy filaments contained within the deepest contours.

But who cares about all this? Isn't $z^2 + c$ a rather restricted function? Hubbard and Douady have also proved the universality of the Mandelbrot set, and with their theory of polynomial-like mappings, they can show that the Mandelbrot set will appear in the parameter space of virtually any complex-analytic iterative scheme. Just two examples of this fact are provided in the next two figures.

Figure 9 shows some of the pictures obtained in Branner's and Hubbard's iteration of cubic polynomials.

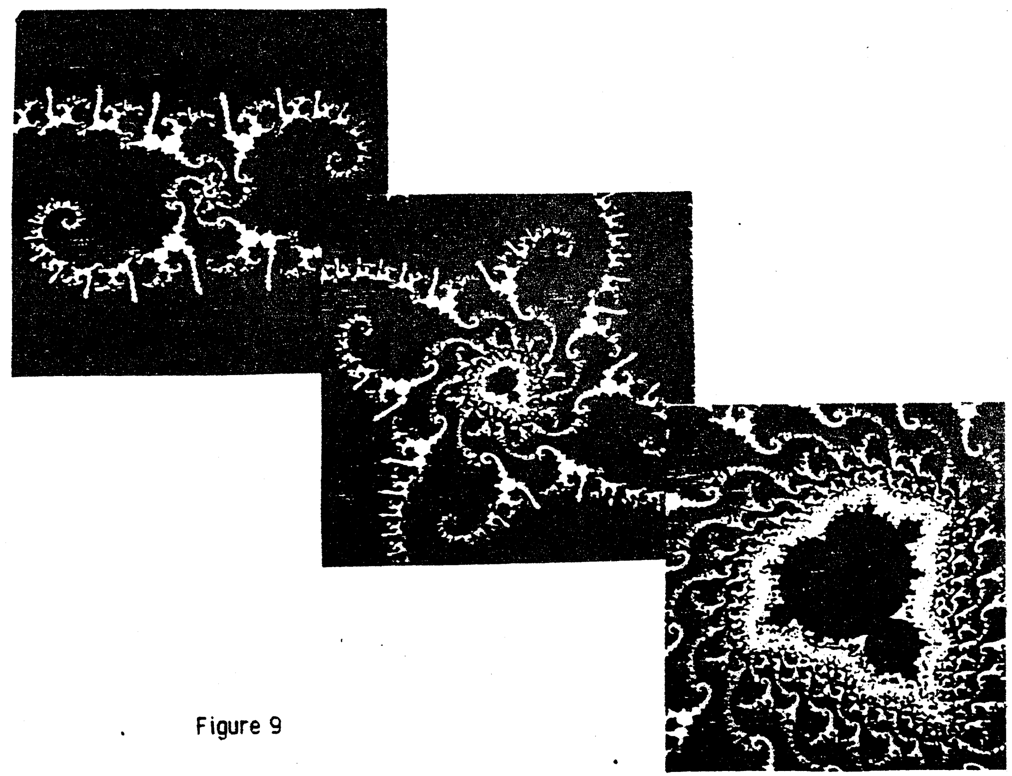
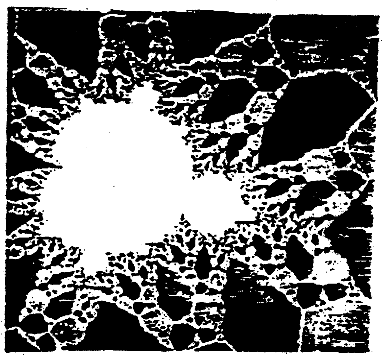


Figure 9

And Figure 10 is perhaps more amazing -- these pictures occurred in a study of Newton's method for cubic polynomials. They are pictures in the parameter space, each point being colored according to which of the three roots 0 will converge under Newton's method for finding the zeros; the values of the parameter for which there is no convergence at all are colored in a fourth color, yellow (which appears as white in the xerox). Once again, different spots of yellow enlarge to show different patterns in the surrounding colors.



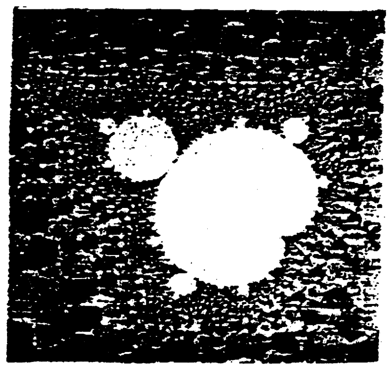


Figure 10

We see over and over again that different copies of the Mandelbrot set are surrounded by different patterns. Being able to see these patterns and their differences is what gives rise to the ideas for theorems, only a few of which have been mentioned above. The computer has not been used to actually prove the theorems, but it has been an absolutely essential tool for guiding the direction of the theoretical work of Hubbard, Douady, and Branner, and their many colleagues and students in dynamical systems.

REFERENCES

Paul Blanchard, "Complex Analytic Dynamics on the Riemann Sphere", Bulletin (new series) of the American Mathematical Society, Volume 11, Number 1, July 1984 (a detailed survey, with a complete bibliography)

Adrien Douady, "Systèmes Dynamiques Holomorphes", Seminar BOURBAKI 35^{e} année, 1982/3 nº 599.

Adrien Douady and John H. Hubbard, "Étude Dynamique des Polynômes Complexes", Publications Mathematiques d'Orsay, 84-02.

Benoit B. Mandelbrot, THE FRACTAL GEOMETRY OF NATURE (W.H. Freeman, 1981).

L'étude graphique réhabilitée par l'ordinateur

Par Michèle ARTIGUE
Véronique GAUTHERON
Emmanuel ISAMBERT

Depuis quelques années, nous travaillons à l'I.R.E.M de Paris 7 sur une présentation des équations différentielles dans un esprit très voisin de D. Tall, Hubbard J. and West B.; nous aimerions y ajouter quelques réflexions issues de notre expérience propre.

Le portrait de phase :

Nous nous sommes limités à l'étude des systèmes différentiels autonomes de dimension 2, c'est-à-dire de la forme

$$S \begin{cases} \frac{dx}{dt} = f(x,y) \\ \frac{dy}{dt} = g(x,y) \end{cases}$$

Ce sont ceux qui se prêtent le mieux à une étude graphique, assistée ou non par ordinateur.

Notre état d'esprit est tout différent de la démarche classique (dans l'enseignement) consistant à chercher les différentes expressions algébriques représentant les solutions : c'est en effet l'information fournie par l'aspect graphique que nous nous proposons de mettre en avant. En d'autres termes, "étudier le système S" signifie essentiellement répondre à la question (quelque peu imprécise) : "Quelle est l'allure globale et locale de l'ensemble des solutions ?" (c'est-à-dire du portrait de phase).

Au cours de cette activité, on est amené à utiliser un type de raisonnement qui semblait depuis quelque temps banni par les mathématiciens "sérieux", ou du moins souvent suspecté de manque de rigueur : le raisonnement purement graphique. (Voir exemple en Annexe).

Peut-être est-il dans ce domaine plus difficile qu'ailleurs de contrôler la rigueur, de déterminer ce qu'il est légitime d'admettre (est-il évident qu'une courbe simple fermée partage le plan en deux régions connexes ?). C'est sans doute une des raisons qui ont fait, ces dernières décennies, négliger l'aspect qualitatif et graphique, y compris dans l'enseignement de la géométrie dans les lycées français.

Une autre raison est que, faute d'ordinateur, on aurait été amené à des calculs souvent répétitifs ou difficiles : trouver les isoclines, étudier la nature des points critiques, déterminer les régions dans lesquelles les trajectoires ont la même allure ; c'est long, et ne conduit pas toujours à un résultat complet et satisfaisant.

L'apparition des micro-ordinateurs munis de bons écrans graphiques ou de tables traçantes a bouleversé les données du problème en multipliant les possibilités d'observations et en permettant un gain de temps inespéré : il est en effet spectaculairement facile d'obtenir le tracé approché d'un grand nombre de trajectoires.

Diverses options sont possibles, du logiciel très sophistiqué (voir article précédent) permettant d'obtenir rapidement de nombreuses images, et qui ne suppose aucun travail informatique de la part des étudiants, jusqu'au petit logiciel de démarrage qui sera progressivement adopté par l'étudiant lui-même en fonction des problèmes rencontrés. Ces différentes options correspondent souvent à des moyens différents, mais il faut réaliser qu'elles vont nécessairement sous-tendre des stratégies d'enseignement différentes, même si dans tous les cas le logiciel est un outil et non l'objectif de l'enseignement.

Nous pensons qu'on peut déjà fournir des objets d'étude très riches avec un programme très simple (basé par exemple sur la méthode d'Euler). La compréhension, sinon du programme dans ses raffinements (tests d'arrêt, calcul automatique de l'échelle, balayage ...) au moins de son principe, nous paraît indispensable pour que l'étudiant comprenne la nature du résultat fourni par la machine, et ses limites (un étudiant passé par l'enseignement classique pourrait s'imaginer que l'ordinateur sait reconnaitre les différents "types" d'équation et applique les méthodes algébriques ad hoc).

On peut fournir des exemples très simples où l'aspect du dessin obtenu dépend de la méthode d'approximation (par exemple trajectoires fermées ou spirales), ou du balayage utilisé pour choisir les différentes conditions initiales.

Apprendre à voir

L'étudiant a besoin d'une étude préalable avant d'utiliser l'ordinateur, ne serait-ce que pour choisir la région du plan à explorer et l'échelle du tracé. On demande alors à l'ordinateur de fournir une image du portrait de phase. Se posent alors des questions parfois délicates :

- que faut-il regarder en particulier ?
- comment l'interpréter ?
- dans quelle mesure peut-on faire confiance au tracé fourni ?
- que reste-t-il à démontrer ?

Sur la figure 1, par exemple, qui représente les trajectoires du système

$$\begin{cases} \dfrac{dx}{dt} = x^2 y^2 - 1 \\ \dfrac{dy}{dt} = x^2 + y^2 - 4 \end{cases}$$

Il faut apprendre à repérer les huit points critiques ; le tracé montre que les points A,B,C,D sont des cols, et que E et F sont des noeuds - ce qu'on peut vérifier facilement par linéarisation du système au voisinage de ces points. Par contre, un doute subsiste quant à la nature des points G et H : les orbites autour de ces points sont-elles des spirales (cas d'un foyer) ou des courbes fermées (cas d'un centre) ? Seule une démonstration permettra de trancher. L'image, en tous cas, nous renseigne sur les positions relatives des divers types de trajectoires.

La figure 2, elle, nous montre les trajectoires du système défini en coordonnées polaires par

$$\begin{cases} \dfrac{d\rho}{dt} = \sin\rho \\ \dfrac{d\theta}{dt} = \cos\rho \end{cases}$$

Le tracé suggère l'existence de cycles limites, et on voit directement sur l'équation que ce sont les cercles de rayon $k\pi$ ($k \in \mathbb{N}^*$).

Insistons encore sur le fait que la possibilité offerte par la machine d'obtenir si facilement le "résultat" cherché, ne dispense pas des démonstrations, bien au contraire : celle-ci sera particulièrement ressentie comme indispensable par les étudiants lorsque subsiste une incertitude dans le tracé - ce qui est généralement le cas pour la nature des branches infinies et de certains ensembles limites.

On essaie plutôt de concevoir un enseignement basé sur une dialectique permanente entre l'observation des tracés et l'activité mathématique proprement dite (avec par exemple des zooms sur les régions intéressantes, pour des études locales). De cette confrontation permanente entre la théorie et la "pratique" va naître une attitude expérimentale faisant partie intégrante du travail mathématique : dégrossissement du travail par exploration, élaboration de conjectures, contrôles de résultats.

Nouvelles méthodes, nouveaux objets

Cette composante expérimentale de l'activité mathématique amène à constater des phénomènes inattendus, et fournit ainsi de nouveaux sujets d'étude ; prenons un exemple frappant :

L'étude du mouvement d'un pendule, avec un frottement égal à ρ, conduit à la résolution du système :

$$\begin{cases} \dfrac{dx}{dt} = y \\ \dfrac{dy}{dt} = -\,\omega^2 \sin x - \rho y \end{cases}$$

(où x est la position du pendule et y sa vitesse).

Une étude classique des points singuliers et des trajectoires singulières conduirait à priori, pour $\rho > 2\omega$, à tracer à la main le schéma de la figure 3.

Or l'ordinateur fournit, pour $\rho = 4\omega$, le tracé représenté sur la figure 4.

Nous constatons qu'en fait les trajectoires semblent "tomber" sur la courbe $y = -\frac{\omega^2}{\rho} \sin x$, pour ensuite en rester très proches.

Ici les notions de l'analyse non standard permettent d'exprimer ce phénomène de manière simple, en "l'accentuant infiniment" et en décrivant le portrait de phase ainsi obtenu.

Si en effet l'on suppose ρ infiniment grand, et qu'on effectue le changement de variable $\tau = \frac{t}{\rho}$, on obtient un portrait de phase correspondant à la situation suivante : quelles que soient les conditions initiales, le pendule va, après un temps infiniment court, suivre un mouvement infiniment voisin (en position et vitesse) d'une des deux solutions singulières.

Même pour des valeurs finies raisonnables de ϱ, cette description rend beaucoup mieux compte de l'allure effective des trajectoires que le premier schéma.

Dans la perspective d'une étude plus fine de ces problèmes (au niveau de la recherche) notons qu'il manque actuellement un pont entre l'interprétation qualitative fournie par l'Analyse non Standard, et la critique des résultats approchés au moyen de l'Analyse Numérique ; il reste là un vaste domaine à explorer, pour répondre à des questions du genre "pour quelles valeurs numériques des paramètres les portraits de phase ressemblent-ils - en un sens à préciser - à ce qu'on obtient en les supposant infinis ?"

En conclusion, nous nous accordons avec les auteurs des articles précédents pour penser que l'utilisation de l'ordinateur permet, d'une part d'enrichir l'enseignement actuel, et d'autre part de l'insérer, à un coût raisonnable, dans une démarche expérimentale.

Nous voudrions insister pour terminer sur le fait que cette utilisation de l'informatique peut et doit prendre des formes variées, chacune ayant son mérite propre : travail des étudiants sur des programmes interactifs perfectionnés, ou sur leurs propres petits programmes, ou sur des documents préparés à l'avance ; utilisation d'images fixes ou animées pour illustrer le cours ....

Michèle ARTIGUE . . . I.R.E.M, PARIS 7
Véronique GAUTHERON I.R.E.M, PARIS 7
Emmanuel ISAMBERT CSP, PARIS 13

Annexe

Exemple de plan d'étude graphique : la chevelure (figures 5 et 6)

$$\begin{cases} \dfrac{dx}{dt} = x^2 + y^2 - 1 = f(x,y) \\[2ex] \dfrac{dy}{dt} = -x = g(x,y) \end{cases}$$

* Puisque $f(x,y) = f(-x,y)$ et que $g(x,y) = -g(-x,y)$, la famille des solutions est symétrique par rapport à l'axe des y (par contre, on ne peut rien conclure du fait que $f(x,y) = f(x,-y)$ et $g(x,y) = g(x,-y)$.

* L'isocline horizontale est l'axe des y ; l'isocline verticale est le cercle unité ; les points critiques sont A(0,1) et B(0,- 1) ; on connaît le signe de f et g dans chaque région du plan ainsi délimitée, ce qui nous renseigne sur la direction du champ.

* D'après le sens du champ, B ne peut être qu'un col ; on peut d'ailleurs le vérifier par linéarisation.

* La linéarisation au voisinage de A, elle, ne permet pas de conclure, mais la symétrie du champ permet de montrer qu'il s'agit d'un centre : les trajectoires proches de A sont des courbes fermées entourant A.

* La solution partant de B vers le haut recoupe l'axe des y au dessus de A ; par symétrie, cette solution revient vers B ; Cette courbe entoure une région dans laquelle les trajectoires sont des courbes fermées.

* Toutes les autres trajectoires possèdent deux branches infinies ; en les comparant aux solutions de $\frac{dy}{dx} = -\frac{1}{2x}$, on peut montrer que ce sont des branches paraboliques de direction Ox.

* Le tracé par ordinateur confirme tous ces résultats.

* Pour plus de détails, voir :
"Systèmes différentiels, étude graphique", par M. ARTIGUE et V. GAUTHERON Ed. Cedic.

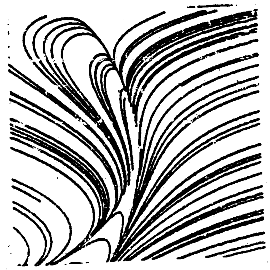


Fig 7 : les trajectoires du systèmes

$$\frac{dx}{dt} = [(x-2)^2 + y^2 - 1)\ (x^2 + y^2 - 9)] \qquad \frac{dy}{dt} = (x-1)^2 + y^2 - 4$$

au voisinage du point critique $x = 3$, $y = 0$

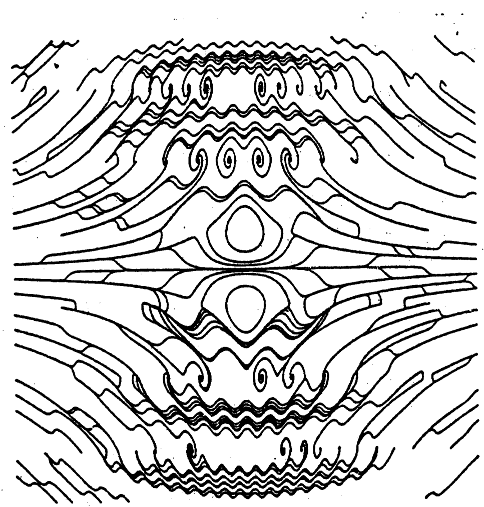


Fig 8 : Reflet du Nautilus traversant un banc de calmars
Vous avez reconnu le systême

$$\frac{dx}{dt} = \cos(y) \qquad \frac{dy}{dt} = \sin(xy)$$

$$\frac{dy}{dt} = -\omega^2 \sin x - \varrho y \qquad \text{pendule : } \frac{dx}{dt} = y$$

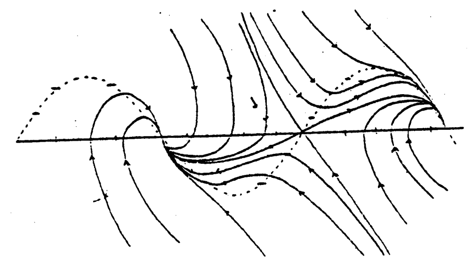


Fig 3 : tracé à la main

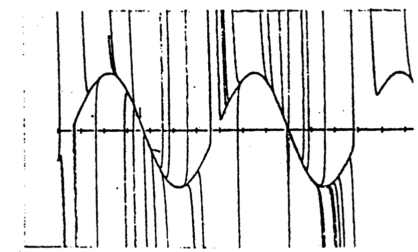


Fig 4 : tracé des trajectoires par ordinateu

$$\frac{dx}{dt} = x^2 + y^2 - 1 \qquad \frac{dy}{dt} = -x$$

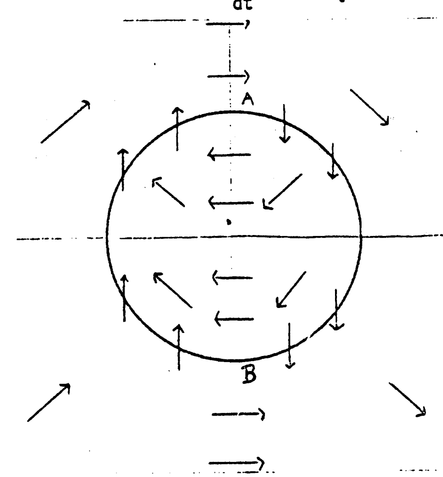


Fig 5 : isoclines et sens du champ

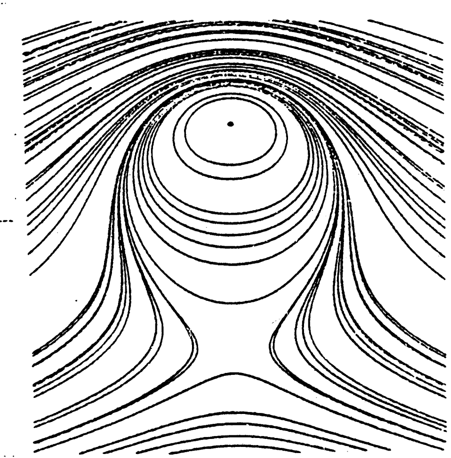


Fig 6 : trajectoires

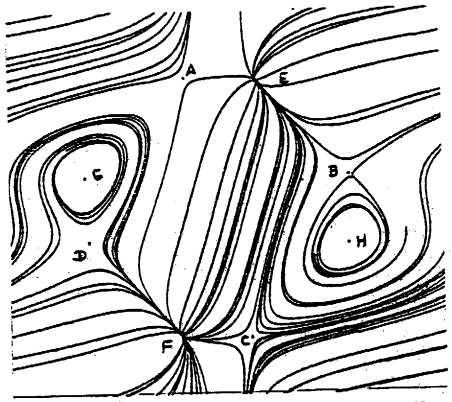


Fig 1 : $\frac{dx}{dt} = x^2 y^2 - 1$ $\frac{dy}{dt} = x^2 + y^2 - 4$

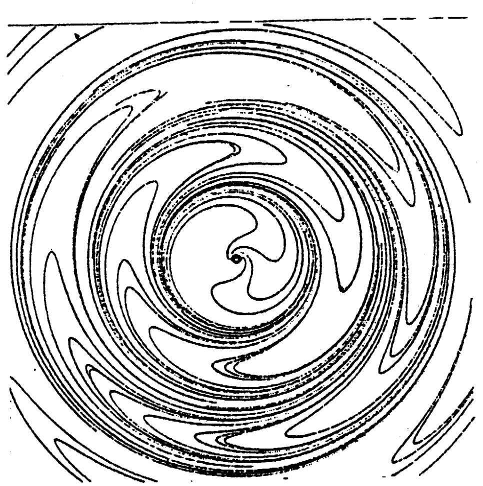


Fig 2 : $\frac{d\rho}{dt} = \sin\rho$ $\frac{d\theta}{dt} = \cos\rho$

On some multigrid finite difference schemes which describe everywhere non differentiable functions

Masaya Yamaguti, Masayoshi Hata

Department of Mathematics, Faculty of Sciences
Kyoto University
Kyoto 606
Japan

We propose a series of multigrid finite difference schemes which can describe everywhere non differentiable functions like Takagi's function and Lebesgue's singular function. Physical meanings of these functions are explained. This study is related to the numerical solution of some singular perturbation problem.

1. Introduction

Recently we observed that the Weierstrass function which is continuous but everywhere non differentiable can be obtained as a solution of very simple functional equation [2]. This functional equation contains a one-dimensional dynamical system and an initially given function g. And then, by changing these dynamical system and the function g, we can get many such families of irregular continuous functions that include the Takagi and Van der Waerden function which was found by T. Takagi in 1903 [1]. On the other hand, we noticed that G. De Rham had found some very simple functional equation which is satisfied by Lebesgue's singular function. This functional equation is very much related to our functional equation. We clarified that these functional equations can be converted to some boundary value problems for multigrid finite difference schemes which are an analog of singular perturbation. Using these result, we succeeded to get a very simple relation between Takagi's function and Lebesgue's singular function. And by product, using de Rham's functional equation, we could compute the Fourier-Stieltjes coefficient of Lebesgue's singular function and prove that it does not satisfy Riemann-Lebesgue theorem.

In the last section, we will show the physical meanings of these functions, explained by H. Takayasu who is a physicist in Nagoya.

2. Functional equation which describe everywhere non differentiable continuous functions

We begin with some trivial remarks. The first example is Weierstrass's function:

$$W_{a,b}(x) = \sum_{n=0}^{\infty} a^n \cos(\pi b^n x) \qquad (0 \le x \le 1),$$

a, b real positive, $0 < a < 1$.

This can be represented, when $b = 2$,

$$W_{a,2}(x) = \sum_{n=0}^{\infty} a^n \cos(\pi\phi^n(x)) \tag{1}$$

where $\phi(x) = 2x$ $(0 \le x \le 1/2)$, $\phi(x) = 2(1-x)$ $(1/2 \le x \le 1)$ and $\phi^n(x)$ means n-th iterate of $\phi(x)$. (Specially, $\phi^0(x) \equiv x$.)

Next example is Takagi's function [1]

$$T(x) = \sum_{n=1}^{\infty} \left(\frac{1}{2}\right)^n \phi^n(x). \tag{2}$$

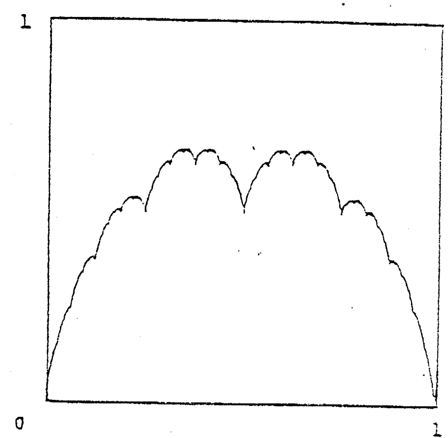


Figure 1. Graph of Takagi's function.

Remark. The above is not the original form of Takagi's function but we interprete the original definition using $\phi(x)$.

Both functions (1) and (2) satisfy the following functional equaton:

$$F(t,x) = tF(t,\psi(x)) + g(x), \qquad (0 \le t < 1) \tag{3}$$

where $\psi(x)$ is a given mapping from [0, 1] to [0, 1], and $g(x)$ is a given bounded function.

It is easy to see that $F(a,x) = W_{a,2}(x)$ for $g(x) = \cos\pi x$ and $\psi(x) = \phi(x)$, and that $F(1/2,x) = T(x)$ for $g(x) = \phi(x)/2$ and $\psi(x) = \phi(x)$.

One can put the equation (3) an initial value problem:

$$\begin{cases} \dfrac{\partial F}{\partial t}(t,x) = \dfrac{\partial}{\partial t}\{tF(t, \psi(x))\} & (0 < t \le 1), \\ F(0,x) = g(x). \end{cases} \tag{4}$$

Then we get the following theorem:

<u>Theorem 1.</u> Suppose that $g:[0,1] \longrightarrow R$ is a bounded function and that $\psi:[0,1] \longrightarrow [0,1]$ is a dynamical system. Then $F(t,x)$, which satisfies (3) and is bounded with respect to x for each t, is uniquely determined and expressed by the following

$$F(t,x) = \sum_{n=0}^{\infty} t^n g(\psi^n(x)). \tag{5}$$

We omit the proof because it is so easy.

<u>Remark.</u> This theorem is very general. For example, given functions $f(x)$ and $\psi(x)$ and a real value s $(o < s < 1)$, we can construct $g_s(x)$ such that the solution $F(t,x)$ of the initial value problem (4) with initial data $g_s(x)$ satisfies

$$F(s,x) = f(x). \tag{6}$$

Thus, we can obtain an expansion of usual Cantor function:

$$\sum_{n=0}^{\infty} \frac{1}{2^{n+1}} \chi_{[1/3,1]}(\phi^n(x)) \tag{7}$$

where $\chi_{[1/3,1]}(x)$ is the characteristic function on the interval $[1/3,1]$.

Now, let us recall de Rham's functional equation. His original work[4] was more general but we mention here a special case which relates to our equation. $M(x)$ is unknown function. His equation is as follows:

$$\begin{cases} M(x) = \alpha M(2x) & (0 \le x \le \frac{1}{2}) \\ M(x) = (1-\alpha)M(2x-1) + \alpha & (\frac{1}{2} \le x \le 1) \end{cases} \tag{8}$$

where α is a real number such that $0 < \alpha < 1$.

For comparison, we examine a spacial case of our equation for Takagi's function:

$$T(x) = \frac{1}{2}T(\phi(x)) + \frac{\phi(x)}{2}$$

which is rewrited in detail as below,

$$\begin{cases} T(x) = \frac{1}{2}T(2x) + x & (0 \le x \le \frac{1}{2}) \\ T(x) = \frac{1}{2}T(2(1-x)) + 1 - x & (\frac{1}{2} \le x \le 1), \end{cases} \tag{9}$$

then (9) is very similar to (8).

The solution of (8) is Lebesgue's singular function, which is strictly increasing continuous and has zero-derivatives almost everywhere for $\alpha \ne 1/2$. We denote this function $M_\alpha(x)$. Later on, we will see that there is a neat relation between $T(x)$ and $M_\alpha(x)$.

3. <u>Schauder expansion</u>

As an analogy of Fourier expansion, we have Schauder expansion of all continuous function on the closed interval [0, 1]. The basis function $F_{i/2^k,(i+1)/2^k}(x)$ is obtained from the function $F_{\alpha,\beta}(x)$ which is defined as follows:

$$F_{\alpha,\beta} = \frac{1}{\beta - \alpha}\{|x-\alpha| + |x-\beta| - |2x-\alpha-\beta|\}. \tag{10}$$

Our bases are the following sequence of functions:

$$1,\ x,\ F_{0,1},\ F_{0,1/2},\ F_{1/2,1},\ \ldots,\ F_{i/2^k,(i+1)/2^n},\ \ldots \tag{11}$$

<u>Theorem 2.</u> Any continuous function $f(x)$ on [0, 1] can be expanded uniquely as follows:

$$f(x) = f(0) + [f(1) - f(0)]x + \sum_{k=0}^{\infty}\sum_{i=1}^{2^k-1} a_{i,k}(f) F_{i/2^k,(i+1)/2^n}(x) \tag{12}$$

where the coefficients $a_{i,k}$ are

$$a_{i,k}(f) \equiv f\left(\frac{2i+1}{2^{k+1}}\right) - \frac{1}{2}\left\{f\left(\frac{i}{2^n}\right) + f\left(\frac{i+1}{2^k}\right)\right\}. \tag{13}$$

The proof is very elementary.

Now we can observe that

$$F_{i/2^k,(i+1)/2^k} = \chi_{[-i/2^k,(i+1)/2^k]}(x)\phi^{k+1}(x).$$

We can replace (12) by

$$f(x) = f(0) + [f(1) - f(0)]x + \sum_{k=0}^{\infty} b_k(x)\phi^{k+1}(x) \tag{14}$$

where $b_k(x) = \sum_{i=0}^{2^k-1} a_{i,k}(f)\chi_{[i/2^k,(i+1)/2^k]}(x)$.

Therefor, we can say that Takagi's function $T(x)$ has a special expansion (14) because that $f(0) = f(1) = 0$ for $T(x)$, and that

$$\forall k, \quad b_k(x) = \frac{1}{2^{k+1}} \qquad (0 \le x \le 1), \tag{15}$$

which means that $T(x)$ satisfies the following boundary value problem for a multigrid finite difference scheme because of (13).

$$\begin{cases} T\left(\frac{2i+1}{2^{k+1}}\right) - \frac{1}{2}\left\{T\left(\frac{i}{2^k}\right) + T\left(\frac{i+1}{2^k}\right)\right\} = \frac{1}{2^{k+1}}, & (0 \le i \le 2^k - 1, \ \forall k) \\ T(0) = T(1) = 0. \end{cases} \tag{16}$$

<u>Remark.</u> If we replace $1/_{2}k+1$ at the right hand side in (16) by $(1/_{2^{k+1}})^2$, then we get usual smooth solution $x(1-x)$ of a Poisson equation $\partial^2 u/_{\partial x}2 = -2$ with boundary condition $u(o) = u(1) = 0$.

The following theorem suggests that Takagi's function can be generalized to some nice class.

non

Theorem 3. A function $f(x)$ is continuous on $[0, 1]$ and $f(0) = f(1) = 0$ if and only if it has expansion $\sum_{n=1}^{\infty} c_n \phi^n(x)$ whose coefficient c_n satisfies

$$(17) \qquad \sum_{n=1}^{\infty} |c_n| < +\infty .$$

The proof of sufficiency is easy. The necesity is a little hard to prove. We only point out that the property of some orbit of the dynamical system $x_{n+1} = \phi(x_n)$ which pass near the mid point 1/2 plays an important role (See [I]). We think this theorem correspond to Sidon's theorem for lacunary Fourier series. Of course, the regularity depends on $\{c_n\}$. If $\{2^n c_n\} \in l_1$, then $f(x)$ is bounded variation. If $\overline{\lim_{n\to\infty}}|2^n c_n| \neq 0$, then $f(x)$ has no derivative everwhere. We call $f(x)$ the generalized Takagi function in this last case.

4. Multigrid finite difference schemes

As we have seen from (16) that $T(x)$ satisfies some multigrid finite scheme, the generalized Takagi function $T_G(x)$ satisfies the following boundary value problem:

$$(18) \qquad \begin{cases} T_G(\frac{2i+1}{2^{k+1}}) = \frac{1}{2}\{T_G(\frac{i}{2^k}) + T_G(\frac{i+1}{2^k})\} + c_k \\ T_G(0) = T_G(1) = 0 , \quad 0 \le i \le 2^k - 1, \quad \forall k \end{cases}$$

where $\{c_n\} \in l_1$, and $\overline{\lim_{n\to\infty}}|2^n c_n| > 0$.

Now we are going to look for some multigrid boundary value problem for the function $M_\alpha(x)$.

We proved that the following boundary problem is the right one:

$$(19) \qquad \begin{cases} M_\alpha(\frac{2i+1}{2^{k+1}}) = (1 - \alpha)M_\alpha(\frac{i}{2^k}) + \alpha M_\alpha(\frac{i+1}{2^k}) \\ M_\alpha(0) = 0, \quad M_\alpha(1) = 1, \quad 0 \le i \le 2^k - 1, \quad \forall k, \end{cases}$$

the proof is easy using the equation (8).

This boundary value problem is closely related to some singular perturbation problem:

$$(*) \qquad \begin{cases} \varepsilon\frac{d^2u}{dx^2} - \frac{du}{dx} = 0 \qquad \varepsilon \text{ small} > 0 \\ u(0) = 0, \quad u(1) = 1, \end{cases}$$

because our problem (18) can be rewritten as follows:

$$(**) \qquad \begin{cases} \frac{1}{4\cdot 2^{k+1}} \dfrac{M_\alpha((i+1)/2^k) - 2M_\alpha((2i+1)/2^{k+1}) + M_\alpha(i/2^k)}{(1/2^{k+1})^2} \\ + (\alpha - \frac{1}{2}) \dfrac{M_\alpha((i+1)/2^k) - M_\alpha(i/2^k)}{\frac{1}{2^k}} = 0, \\ M_\alpha(0) = 0, \quad M_\alpha(1) = 1, \quad 0 \le i \le 2^k - 1, \quad \forall k \text{ and } 0 < \alpha < \frac{1}{2}. \end{cases}$$

That is, (**) is some kind of discretization of (*) but the mesh size depends on ε. Thus we can regard $M_\alpha(x)$ as a kind of boundary layer solution.

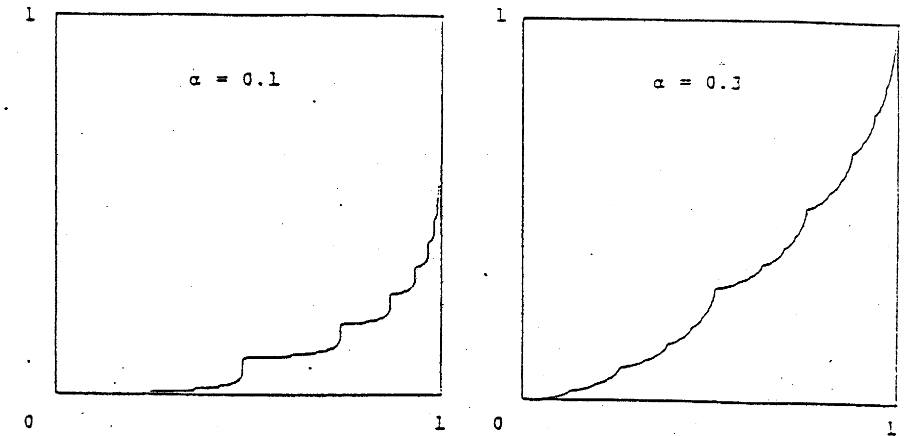


Figure 2. Graphs of Lebesgue's singular function M_α.

5. A relation between $T(x)$ and $M_\alpha(x)$

Using (13) and (19), we get the coefficient $a_{i,k}(M_\alpha)$ in Schauder expansion of M_α:

$$(20) \qquad a_{i,k}(M_\alpha) = (\alpha - \tfrac{1}{2})[M_\alpha(\tfrac{i+1}{2^k}) - M_\alpha(\tfrac{i}{2^k})].$$

With this equality, we can obtain the relation

$$(21) \qquad \begin{cases} a_{2m,k} = \alpha a_{m,k-1} \\ a_{2m+1,k} = (1 - \alpha)a_{m,k-1}. \end{cases}$$

Theorem 4.

$$a_{i,k}(M_\alpha) = (\alpha - \tfrac{1}{2})\alpha^p(1 - \alpha)^q$$

where $p + q = k$, p is the number of 0's in the binary expansion of

$$i = \sum_{j=1}^{k} w_j 2^{j-1} \qquad (0 \le i \le 2^k - 1),$$

q is the number of 1's.

Because of this theorem, the series of Schauder expansion for M_α is holomorphic function of α in neighbourhood of 1/2. We can differentiate (19) with respect to α in some neighbourhood of 1/2, and if we put $\alpha = 1/2$, then we get

$$2T(X) = \frac{\partial M_\alpha(x)}{\partial \alpha}\Big|_{\alpha=\frac{1}{2}}.$$

Theorem 5.

$$\frac{\partial M_\alpha(x)}{\partial \alpha}\Big|_{\alpha=\frac{1}{2}} = 2T(x).$$

6. Physical meaning of $M_\alpha(x)$ and $T(x)$

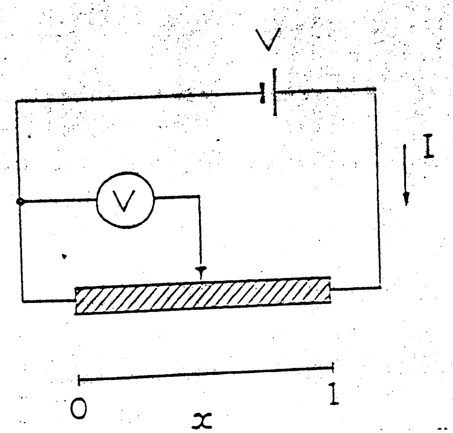


Figure 3.

We are thinking of an electric circuit in which constant voltage V is applied to a 1-dimensional resistance of length unity (Fig. 3). The Ohm's law is the following.

$$E(x) = R(x)I \tag{22}$$

where I is the electric current, $E(x)$ and $R(x)$ are the electric field and the resistivity respectively.

If we assume that the resistivity is proportional to the density of impurity $\rho(x)$, namely

$$R(x) = \kappa\rho(x) \qquad \kappa : \text{constant},$$

then $V(x)$ voltage at point x, becomes

$$V(x) = \int_0^x E(x')dx' = V\kappa\int_0^x \rho(x')dx' \tag{23}$$

where ρ is normalized as

$$\int_0^1 \rho(x')dx' = 1.$$

Now, we consider the case where $\rho(x)$ is de Wij's fractal [6]. De Wij's fractal $\rho_\alpha(x)$ is a self-similar function specified by only one real parameter α $(0 \le \alpha \le 1)$. It is defined by a limit of cascade of the coarse grained distribution $\rho_\alpha^{(k)}(x)$ which are defined by

$$\begin{cases} \rho_\alpha^{(k+1)}\left(\frac{2i}{2^{k+1}}\right) = \alpha\rho_\alpha^{(k)}\left(\frac{i}{2^k}\right) \\ \rho_\alpha^{(k+1)}\left(\frac{2i+1}{2^{k+1}}\right) = (1-\alpha)\rho_\alpha^{(k)}\left(\frac{i}{2^k}\right) \qquad (0 \le i \le 2^k - 1) \\ \rho_\alpha^{(0)}(0) = 1. \end{cases} \tag{24}$$

Fractal dimension D of ρ_α is known to be

$$D = -\{\alpha\log_2\alpha + (1-\alpha)\log_2(1-\alpha)\}.$$

Now the voltage $V(x)$ is obtained from (23),

$$V(x) = V\kappa\int_0^x \rho_\alpha(x')dx' = V M_\alpha(x) \tag{25}$$

where $M_\alpha(x)$ is Lebesgue's singular function appeared in preceding sections.

Next we treat the case where the density $\rho_\alpha(x)$ of the impurity changes with time.

From the conservation of impurity, the flux of the impurity $j(t,x)$ is determined as

$$j(t,x) = -\int_0^x \frac{\partial}{\partial t}\rho(x',t)dx'.$$

If we assume that the density is uniform at $t = 0$, namely $\rho(x,0) = \rho_{1/2}(x) = 1$, and that it becomes De Wij's fractal after a short time Δt, namely $\rho(x,\Delta t) = \rho_{1/2+\alpha'\Delta t}(x)$,

then the flux at $t = 0$ can be computed as

$$\begin{aligned} j(x,0) &= -\lim_{\Delta t\to 0}\int_0^x \frac{\rho_{1/2+\alpha'\Delta t}(x') - \rho_{1/2}(x')}{\Delta t}dx' \\ &= -\alpha'\frac{\partial}{\partial\alpha}M_\alpha(x)\Big|_{\alpha=\frac{1}{2}} \\ &= -2\alpha' T(x) \qquad \text{(Theorem 5)} \end{aligned}$$

where $T(x)$ is Takagi function.

The above discussion is also valid to other cases, for example, laminar shear flow ρ, V and j represent density of vorticity, velocity of the fluid and flux of vorticity respectively. The third case is just concerning about density of change, electric field and electric current respectively.

Appendix. The Fourier Stieltjes coefficient of $M_\alpha(x)$ can be computed using (7). Let

$$I(t) = \int_0^1 e^{itx} dM_\alpha(x)$$

be this coefficient, then we get

$$I(t) = \prod_{n=1}^{\infty} \{ \alpha + (1-\alpha) e^{\frac{it}{2^n}} \}$$

for all integer $p \geq 2$,

$$I(2^p\pi) = (2\alpha - 1) I(\pi).$$

If $\alpha \neq 1/2$, $I(t)$ never vanish as t tends to $+\infty$.
The other expression of $I(t)$ is

$$I(t) = e^{it} \prod_{n=1}^{\infty} \{\cos\frac{t}{2^{n+1}} + (1-2\alpha) i \sin\frac{t}{2^{n+1}}\}.$$

Refferences

[1] Takagi, T., A simple example of the continuous function without derivative, Proc. Phys.-Math. Japan 1 (1903) 176-177.

[2] Yamaguti, M. and Hata, M., Weierstrass's function and chaos, Hokkaido Math. Jrnl. (1983) to appear.

[3] Hata, M. and Yamaguti, M., On Takagi's function , Japan Jrnl. of Applied Math. (New Jrnl.) to appear.

[4] De Rham, G., Sur quelques courbes définies par des équations functionnelles, Rend. Semi. Math. Torino 16 (1957) 101-113.

[5] Faber, G., Über stetige Funktionen, Math. Ann. 69 (1910) 372-443.

[6] Mandelbrot, B., The fractal geometry of nature, Freeman (1982) 376.

I.C.M.I. SYMPOSIUM

LISTE DES PARTICIPANTS

ALLEN Richard	IRISA - Campus universitaire de Beaulieu - Avenue du Général Leclerc 35042 RENNES CEDEX FRANCE
APERY	Département de Mathématiques - 7, rue R. Descartes 67084 STRASBOURG CEDEX FRANCE
BIEHLER Rolf	Institut für Didakitk der Mathematik - Universität Bielefeld, Postfach 8640 D-4800 BIELEFELD 1 GERMANY
BITTER Gary	Payne 203 - School Science and Mathematics - Arizona State University - Tempe, Arizona 85287 U.S.A.
BOGART Kenneth	Department of Mathematics and Computer Science - Dartmouth College - HANOVER NH 03755 U.S.A.
BRUIJN De Nicolaas	Eikenlaan 2 - 5671 AB NUENEN - THE NETHERLANDS
BURKHARDT Hugh	Shell Center for Mathematical Education - University of Nottingham - NOTTINGHAM 2RD - ENGLAND
BUTTLER Douglas E.	The M.E.I. Schools Project - 41a, West Street, OUNDLE, PETERBOROUGH - PE8 4EJ - ENGLAND
CHURCHHOUSE Robert	University College, Cardiff - Mathematics Institute, Senghennydd Road, CARDIFF - CF2 4AG - PAYS DE GALLES, U.K.
CONZE J.P.	Université de Rennes 1. U.E.R. de Mathématiques Avenue du Général Leclerc - 35042 RENNES CEDEX FRANCE
CORNU Bernard	Institut Fourier - B.P. 74 - 38402 SAINT MARTIN D'HERES FRANCE
DAVENPORT J.H.	School of Mathematics. University of Bath. BATH B.A.2 7 AY ENGLAND
DEBIEVE Camille	123, Chaussée de Bruxelles - 7460 CASTEAU - BELGIQUE
DELMEZ Claude	1A, rue du Moulin - 5991 TOURINNES-LA-GROSSE BELGIQUE
DUBINSKY Ed	Department of Mathematics and Computer Science - Clarkson University - Potsdam, NEW YORK 13676 U.S.A.
DUPUIS Claire	I.R.E.M. de Strasbourg - 10, rue du Général Zimmer 67084 STRASBOURG CEDEX FRANCE
ENGEL Arthur	AM TREUTENGRABEN 49 - 6000 FRANKFURT 90 - WEST GERMANY
FRASER Rosemary	Shell Center for Mathematical Education - University of Nottingham, NOTTINGHAM 2RD - ENGLAND
GRAF Klaus	Kurstraße 5. D. 1000 BERLIN 38 . R.F.A.
GUIDY WANDA Joséphine	Université d'Abidjan - 08 BP 217 - ABIDJAN 08 COTE D'IVOIRE
HEBENSTREIT Jacques	E.S.E. Plateau du Mouton - 91190 GIF/YVETTE FRANCE
HODGSON Bernard	Université de Laval - Cité Universitaire - QUEBEC. P.Q. CANADA GIK 7P4
HOUBEN Jean Paul	78, rue de l'Eglise - 1301 WAVRE - BELGIQUE
HOWSON Albert Geoffrey	Centre for Mathematics Education - The University SOUTHAMPTON - S09 5NH ENGLAND
IGOT Jean Pierre	I.R.E.M. de Strasbourg - 10, rue du Général Zimmer 67084 STRASBOURG CEDEX FRANCE
JAKUBOWICZ Daniel	46, rue Epoigny - 94120 FONTENAY SOUS BOIS FRANCE
KAHANE Jean Pierre	11, rue du Val de Grâce - 75005 PARIS FRANCE
KLINGEN Leo H.	Billrothstraße 2 - D 5300 BONN 1 - BRD
KIVELA Simo Kalervo	Department of Mathematics - Helsinki University of Technology - OTAKAARI 1, SF - 02150 ESPOO FINLAND

KUCK Conrad	Universität - Gesamthochschule Paderborn, Fachbereich 17 Warburgerstraße 100, D-479 PADERBORN R.F.A.
LANE Kenneth	Math. Department - Harvey Mudd College - CLAREMONT, CA 91711 U.S.A.
LEHNING Hervé	84, boulevard Joffre - 92340 BOURG LA REINE FRANCE
LIM Chong-Keang	Department of Mathematics - University of Malaya - KUALA LUMPUR , MALAYSIA
MANDELBROT	IBM Research, P.O. Box 218, Yorktown Heights 10598 NEW YORK U.S.A.
MARZOLLO	Représentant l'UNESCO
MASCARELLO Maria	Dipartimento di Matematica - Politecnico di Torino - Corso Duca Degli ABRUSSI 24 - I 10129 ITALY
MASSOBRIO F.	C. Agnelli, 60 - 10137 TORINO ITALY
MEISSNER Hartwig	Westfälische Wilhelms Universität - Fliednerstraße 21 D-4400 MUNSTER WEST GERMANY
MORIN	Département de Mathématiques - 7, rue R. Descartes 67084 STRASBOURG CEDEX FRANCE
MULLER Eric	Department of Mathematics - Brock University - St CATHARINES, ONTARIO - L2S 3A1 CANADA
MURAKAMI Haruo	Department of Applied Mathematics - Faculty of Engineering Kobe University - Rokko, Nada, Kobe 657 JAPAN
NEWMAN Michael	Department of Mathematics - IAS - Australian National University - GPO Box 4 - CANBERRA ACT - Z 603 AUSTRALIA
NIMAN John	Long Hill Rd. - Hopewell Junction - NEW YORK 12533 U.S.A.
OLLONGREN Alexander	Department of Computer Science - Wassenarseweg 80 - LEIDEN - 2333 AL HOLLAND
OKON James	Mathematics Department - California State University San Bernadino - CALIFORNIA 92407 U.S.A.
OTTE Michael	Universität Bielefeld - Institut für Didakitik der Mathematik - Postfach 8640 - D-48000 BIELEFELD1 GERMANY
PLUVINAGE François	Département de Mathématiques - 7, rue R. Descartes 67084 STRASBOURG CEDEX FRANCE
POLLAK Henry	Room 2Q-386, Bell Communications - Research 435 South Stre MORRISTOWN , NEW JERSEY - 07960 U.S.A.
RALSTON Anthony	Department of Computer Science - University at Buffalo 226 Bell Hall - Buffalo, NEW YORK 14260 U.S.A.
SALINGER David	School of Mathematics - Leeds University - LEEDS LS2 9JT - ENGLAND
SARGENT Robert	42 Springhead Ferndale - TUNBRIDGE WELLES - KENT ENGLAND
SAUNDERS David	BBC-Open University Production Centre - Walton Hall MILTON KEYNES - MK7 6BH - U.K.
SCHMITT Bruno	U.E.R. Des Sciences Exactes et Naturelles 57000 METZ FRANCE
SCHUYTEN Gilberte	Jemappesstraat 8 - GENT 9000 BELGIQUE
SEIDMAN Stephen	Department of Computer Sciences - George Mason University FAIRFAX, VIRGINIA 22030 U.S.A.
STEEN Lynn Arthur	Department of Mathematics. St Olaf College - NORTHFIELD, MINESOTA 55057 U.S.A.
STERN Jacques	Université de Caen - Esplanade de la Paix 14032 CAEN FRANCE
STOUTEMEYER D.R.	University of Hawaî - HONOLULU U.S.A.
STRASSER Rudolf	Akazienallee 2a, 4700 Hamm 1, R.F.A.
TAKENOUCHI Osamu	2-4, D5-308, Shinsenri - Higashi - TOYONAKA-SHI 565 JAPAN

TALL David Orme	Mathematics Education Research Centre - Univerity of Warwick - COVENTRY CV4 7AL - U.K.
THORNE Michael	Department of Computing Mathematics - University College CARDIFF CF2 4AE WALES U.K.
VANHAMME Jacqueline	Rue Firmin Martin 2, B -1160 BRUXELLES BELGIQUE
VAN LINT Jacobus	Department of Mathematics and Computing Science Eindhoven University of Technology. P.O. BOX 513 EINDHOVEN - 56000 MB THE NETHERLANDS
VEZINA	Université de Québec. Chicoutimi - 555 boulevard de l'Université CHICOUTIMI - QUEBEC CANADA
WEST Beverly	Department of Mathematics. White Hall, Cornell University ITHACA, NY 14853 U.S.A.
WINKELMANN Bernard	Institut für Didaktik der Mathematik. Universität Bielefeld. D.4800 BIELEFELD 1 WEST GERMANY
WINTER Mary Jean	Department of Mathematics. Michigan state University EAST LANSING, MI 48824 U.S.A.
YAMAGUTI	Department of Mathematics. Faculty of Sciences KYOTO University T 606 - SAKYOKU KYOTO JAPAN

NOTES SUR LE FONCTIONNEMENT DU SYMPOSIUM

Le symposium s'est déroulé au Centre Saint-Thomas, qui offre des conditions d'hébergement et de travail excellentes, propices au déroulement d'un programme en définitive très soutenu durant toute la semaine.

Afin de permettre l'élaboration des documents d'orientations qui figureront dans le livre des Proceedings (cf. p. 6), un canevas général d'organisation avait été mis au point, mais le programme précis des interventions était déterminé en cours de symposium. Trois types de séances étaient prévus :

- les sessions plénières,
- les groupes d'élaboration,
- les démonstrations de réalisations.

Pour chaque session plénière, un président de session était responsable du choix des intervenants sur les thèmes fixés. Afin qu'une bonne partie de la session donne lieu à des discussions et des débats, les participants avaient reçu, environ un mois avant le début du symposium, un "document de travail" leur permettant de se préparer. De ce fait, les discussions prévues ont pu être menées de façon constructive par les présidents de session. Voici quelles ont été les sessions (la place nous fait défaut pour mentionner les intervenants) :

Lundi — Thème : L'influence de l'informatique sur les mathématiques
Session présidée par J.P. KAHANE.

Mardi — Thème : Mathématiques discrètes et continues
Session présidée par A. RALSTON.

Mercredi — Thèmes : Problèmes d'implémentation ; visualisation dans les études d'équations différentielles
Session présidée par H. POLLAK

Jeudi — Thèmes : Algèbre de l'informatique et logique
Session présidée par J. Van LINT.

Vendredi — Thèmes : Géométrie ; Statistiques ; Expérimentations dans l'enseignement
Session présidée par L.A. STEEN.

Les groupes d'élaboration, respectivement placés sous la responsabilité de R.F. Churchhouse, A. Ralston et B. Cornu, ont régulièrement travaillé pendant la semaine dans chacun des trois domaines qui sont dégagés dans le document préparatoire de l'ICMI (cf. Page 5). Le résultat de leurs travaux a été présenté en session plenière spéciale, le vendredi après-midi. Après révision d'ensemble par les responsables, ce sont ces travaux qui donneront lieu aux articles d'orientation du livre des "Proceedings", à paraître en 1986.

En fin d'après-midi et en soirée, la présentation de réalisations sur ordinateurs ont permis à chacun d'être confronté à un large éventail de possibilités concrètes. Des présentations ont été proposées par : F. Apéry, B. Schmidt, B. Mandelbrot, H. Burkhardt, D. Sargent, B. West, Delmez et Debièvre, J.P. Houben, M. Mascarello, D. Saunders. Par le texte, il n'est pas possible, malheureusement, de faire ressortir le dynamisme et l'esthétique de ces présentations. Les figures d'accompagnement, pour celles qui font l'objet d'un article dans ce document, permettent au lecteur de se faire déjà une certaine idée.

En outre, deux groupes de travail spéciaux ont été institués, sur deux thèmes dont l'approfondissement a paru nécessaire. L'un a traité de l'analyse ("calculus") ; son "noyau" était constitué de : B. Winkelmann, B. West, D. Tall, O. Takenouchi, M. Mascarello et B. Hodgson. L'autre a traité de logique et systèmes symboliques, avec notamment : A. Ollongren, K. Kuck, J. Davenport et N.G. de Bruijn.

Le fonctionnement de tels groupes, non programmés d'avance, illustre de façon éloquente un désir général de participation active aux travaux du symposium. Dans une telle ambiance, le travail de l'équipe d'organisation locale (essentiellement : C. Dupuis, R. Flach, F. Pluvinage et E. Ramirez) ne pouvait être que gratifiant et payé de retour.

Le Secrétaire du Comité de Programme

François PLUVINAGE

Note sur le choix des articles de cette brochure :
Vu l'abondance des propositions, nous avons dû nous limiter assez strictement, à ne retenir que les articles correspondant bien au sujet abordé. Certains articles non retenus méritent d'être soumis à d'autres colloques.