

Commission Internationale de l'Enseignement Mathématique
(International Commission on Mathematical Instruction)

THE INFLUENCE OF COMPUTERS AND INFORMATICS
ON MATHEMATICS AND ITS TEACHING

* * *
* *
*

Strasbourg 25 - 30 march 1985

DOCUMENT DE TRAVAIL

ulp

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

THE INFLUENCE OF COMPUTERS AND INFORMATICS
ON MATHEMATICS AND ITS TEACHING

Strasbourg 25-30 march 1985

DOCUMENT DE TRAVAIL

1. ATIYAH M.F. Mathematics and the computer revolution	1
2. STRÄSSER R. Comment on the ICMI-paper	9
3. KAPADIA R. Rigour and relevance in mathematics	11
4. YAMAGUTI M. [1. The influence of computer and informatics on mathematics	19
[2. On some multigrid funite difference schemes which describe everywhere non differentiable functions	23
5. KUCK K. Über ergodische, nicht-monotone lernende Automaten	29
6. DE BRUIJN N.G. Contribution to the ICMI discussion	35
7. SALINGER D.L. The effect of computers on the teaching of mathematics students	45
8. MURAKAMI H. et MASATO H. The progress of computers and mathematical education	47
9. STEEN L.A. Living with a new mathematical species	57
10. CALMET J. Introducing computer algebra to users and students	71
11. BITTER G. First year results of the microcomputer assisted mathematics remediation project at Arizona State University	77
12. DUBINSKI E. Computer experience as an aid in learning mathematics concepts	87
13. KLINGEN L.H. Bemerkungen zu "The influence of computers ..."	97
14. THORNE M. Computers as a university mathematics teaching aid : toward a stratagy	101
15. STERN J. Réflexion sur certaines bases mathématiques de l'informatique	109

16. OTTE M. Was ergibt sich aus der Grundlagenkrise der Mathematik ?	113
17. BOIERI P. et SCARAFIOTTI A.R. Personal computers in teaching basic mathematical courses	115
18. BIEHLER R. Comments on the ICMI-Paper	119
19. NIMAN J. The influence of microcomputers on the elementary school mathematics curriculum	125
20. OKON J.S. Computer assisted instruction in undergraduate mathematics	127
21. SEDA A.K. Computer science and the mathematics curriculum	129
22. GUIDY WANDJA J. Contribution on symposium	133
23. MEISSNER. Some comments responding to the ICMI discussion document	137
—————	
24. HARTHONG J. Eléments pour une théorie du continu	139
25. RICE et SEIDMAN. A fundamental course in higher mathematics incorpora- ting discrete and continuous themes	143
26. JENKYN S T.A. et MULLER E.R. Discrete mathematics (2 years experience with an introductory course)	145
27. BOGART K.P. What should a Discrete mathematics course be ?	149
28. LANE K.D. Symbolic manipulators and the teaching of college mathematics	153
29. STOUTEMYER D.R. Using computer symbolic mathematics for learning by discovery	155
30. OLLONGREN. Formules manipulation in teaching perturbation methods	161
31. KIMBERLING C. Conics	167
32. IREM de Strasbourg. Expériences sur les aspects de l'informatique à l'en- seignement des mathématiques	173
33. BURCKHARDT H. Computer aware curricula : ideas and realisation	189
34. TAKENOUCHE O. Computers in the beginner's course of the calculus	197
35. TALL D. Visualizing calculus concepts using a computer	203
36. HODGSON et al. Introductory calculus in 1990	213
37. WINKELMANN B. The impact of the computer on the teaching of analysis	217
38. ENGEL A. Algorithmic aspects of stochastics	233

39. RÅDE L. Report on ISI round table conference in Canberra (in august 1984) 241
40. SAUNDERS D.J. Computer animations in mathematics teaching at the Open
University 245
41. MASON J. What happens when you swith off the machine ? 251

Ce symposium est organisé grâce au support du Fonds mathématique international,
alimenté par les contributions de :

- l'UNESCO,
- IBM Europe,
- Herrmann éditions,
- IBM France

ainsi que de :

- 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.

M.F. Atiyah

§1. A historical perspective

This Orwellian year of 1984 provides an inviting occasion for us to look to the past, present and future of mankind and in particular to consider the constantly changing relations between Science and Society. While George Orwell pin-pointed with great dramatic effect many of the political dangers of "double-think", the perversion of truth for political ends, he underestimated in other ways the enormous changes which Science had in store for us. The major problem we face today is of course the existence of atomic weapons and our capacity to destroy civilization, but even if this problem is solved many other challenges remain and prominent among these is the computer revolution.

It is now commonly acknowledged that we are firmly embarked on an economic and social revolution which will be comparable in scope and effect to the industrial revolution. There are here many significant analogies but also many important differences, notably in the speed of change. Whereas the industrial revolution is usually measured in centuries, the computer revolution is properly measured in decades. Since the human life-span has not fundamentally altered, the impact of the computer revolution will be faster and more acute in sociological terms, and coming to terms with it will be correspondingly more difficult.

Not being an economist or a sociologist I will leave it to others to elaborate on the obvious problems and likely

developments in these areas. As a mathematician I am more concerned with another aspect of the computer revolution and one in which it differs fundamentally from its predecessor the industrial revolution. Whereas the eighteenth and nineteenth centuries witnessed the gradual replacement of manual labour by machines, the late twentieth-century is seeing the mechanization of intellectual activities. It is the brain rather than the hand that is now being made redundant. This means that the challenge which we face is of quite a different order and analogies with the past may therefore be misleading.

The intellectual challenge presented by the computer is, I believe, very far-reaching even if at the present time the problems are only just beginning to emerge. Moreover this challenge is certainly not restricted to mathematics, but will eventually penetrate into almost all aspects of human activity. For example, we are already seeing the introduction of "expert systems" into fields such as medicine and law, and the roles of doctors and judges as we now understand them are unlikely to survive unchanged into the next century. Science fiction in these directions has difficulty in keeping pace with fact.

Exciting though it is to speculate on the computerization of thought and knowledge in such fields I will, for two reasons, restrict myself to mathematics. The first and most important is that I am myself a mathematician and that I can speak about this area at first-hand and with some confidence. The temptation to pretend to expertise which one does not possess should be firmly avoided. The second reason for concentrating on mathematics is that, in the public eye at least, this subject

tends to be naturally, though not always correctly, associated with computers.

It is of course true that, in its early days, computer science grew up alongside mathematics and that famous mathematicians such as Turing and von Neumann were amongst its pioneers. Moreover, of the traditional basic sciences, mathematics is still the closest in spirit to computer science. In fact, it is sometimes asserted, with dry humour, that computer science is the Cuckoo of the mathematical nest with all the unpleasant overtones which that suggests.

In this world of education, mathematics and computer science still go hand in hand even if the relationship is now an uneasy one. In universities, Computing and Mathematics are frequently found together and, at the school level, computing is almost exclusively in the hands of mathematics teachers.

For all these reasons it seems to me that mathematicians have a responsibility to explain, to society at large, intellectual challenges and dangers presented by the computer revolution. This is what I hope to do today. As I have already mentioned I shall restrict myself to describing the impact of computers on mathematics itself, but at a fundamental level I believe that many of the things I shall say will have some relevance to other fields of intellectual study, though I will leave it to each of you to decide how far my remarks pertain to your own discipline or field of interest.

Finally, I should issue a disclaimer. Some of my mathematical colleagues have much more direct experience of computing than I have. In fact, at the technical level I am barely a novice, but I hope that, at a higher level, I am aware of what is happening and that my perception is not too far off the

§2. Mathematics and Theoretical Computer Science

It might be helpful if I began by describing the role which mathematics has played, and continues to play, in the development of the theoretical aspects of computers. Not surprisingly, those parts of mathematics which have been relevant in this respect have themselves received an enormous stimulus in return. On the one hand, this has been beneficial in many ways by suggesting fruitful lines of research, but the sheer scale of the computer field brings dangers in its wake, and I shall return to this later.

Historically it was mathematical logic which provided the theoretical basis for computers. Here, and throughout my lecture, I am referring to the 'software' side of computers, concerned with the development and use of suitable languages rather than the 'hardware' side which is concerned with their physical design and construction. Of course, it is the hardware development - the rise of the minute silicon chip - which has produced the revolution, but this in its turn only re-emphasizes the need for greater sophistication in the language so as to fully exploit the hardware potentialities.

Mathematicians have always been concerned with the notion of "proof", the rigorous deduction of various conclusions from given assumptions, and in the first half of the twentieth-century this notion was subjected to extremely careful analysis. In particular there emerged the notion of a "constructive" proof, where the desired conclusion could be arrived at after a finite number of definite steps. The famous "Turing machine" was a hypothetical ideal machine which could carry out such constructive proofs, and the early computers were in essence its physical realization.

Computers have to be given precise commands and mathematical logic provides the theoretical framework in which such commands can be formulated. Moreover as computers become more and more powerful, so the languages they use become increasingly sophisticated, and the problems of possible errors loom much larger. The errors I refer to are not of course machine errors - a machine can do no wrong - but of human errors in issuing the right commands or in translating into the computer language. Here again mathematical ideas of proof become important - how to prove that a given set of computer instructions are correct.

This very briefly is why mathematical logic is related to theoretical computer science and why students trained in this most abstract of mathematical disciplines find a ready demand for their talents in the computer field.

Closely related to the notion of constructive proof is that of an "algorithm" which in mathematical parlance is the term used for a definite procedure for solving a problem. For example an explicit formula for solving an equation is an especially simple algorithm. If a mathematician wants to use a computer to solve a problem he needs to give the computer an algorithm. Now algorithms can be fast or slow, measured in computer time, and there is clearly a great advantage in devising fast algorithms. Thus the development of computers has stimulated a whole new branch of mathematics, complexity theory, which is essentially concerned with understanding how "complex" an algorithm is and which roughly corresponds to how long a computer will take to give an answer.

Proof theory and complexity are just two examples of the sort of mathematics which has been stimulated or created by the needs of the computer. In general the sort of mathematics involved is quite different from that required by the applications to physical science. Because computers are based on the on/off switch of electrical circuits they involve discrete mathematics exemplified by algebra, whereas, since Newton's time, physical science has been based largely on the application of calculus to the study of continuously varying phenomena. This has led some people to argue that the traditional approach to teaching mathematics, with its heavy emphasis on calculus must, in the age of the computer, be drastically modified.

3

§3. Computers as an aid to mathematical research

Having described the way in which mathematics has helped the development of computer science, let me now consider the flow in the opposite direction. In what ways has the advent of computers assisted and altered mathematical research?

4 The first and most obvious use of computers has been simply as "number crunchers". High-speed machines are excellently adapted to carrying out very large numbers of repetitive calculations, so that explicit numerical answers can rapidly be provided for problems which would otherwise have been too complicated to handle. This use of computers has had a dramatic effect on all of applied mathematics and it has significantly altered our conception of what is a satisfactory solution of a mathematical problem. In pre-computer days mathematicians would work hard to cast the solution of a problem into some elegant algebraic form, involving familiar objects such as algebraic and trigonometric expressions. Nowadays a problem in applied mathematics is regarded as satisfactorily solved if one can find an algorithm to feed into a computer which will generate all the numerical values one is interested in.

Not all of mathematics however is concerned with numbers. Algebra for instance deals with symbolic expressions which may or may not stand for unknown numbers. For example, an expression in mathematical logic does not stand for anything numerical. The manipulation of complicated symbolic expressions can also be performed on computers and there are areas of mathematics where this has already been applied very successfully. For example, the determination of all finite simple groups, the

building blocks of abstract symmetries, was greatly assisted by the use of powerful computers. With the increased availability of micro-computers and the greater computer expertise among the younger generation of mathematicians it seems certain that these symbolic uses of computers will greatly increase.

In Mathematics, as in the Natural Sciences, there are several stages involved in a discovery, and formal proof is only the last. The earliest stage consists in the identification of significant facts, their arrangement into meaningful patterns and the plausible extraction of some law or formula. Next comes the process of testing this proposed formula against new experimental facts, and only then does one consider the question of proof.

In all the earlier stages computers can play a role, particularly when large or complex systems are being considered. For example, in Number Theory interesting questions may involve very large prime numbers, and some of the deepest conjectures being studied at the present time have been based on extensive computer calculations. In the same way problems in differential equations which involve the evolution of some system (e.g. the flow of a liquid) for a very long time have been enormously influenced by experimental facts discovered on computers.

One advantage of present day computers which is only just beginning to be fully appreciated by mathematicians is their ability to display information graphically (and even in colour). For many complicated mathematical problems involving geometrical features, this provides an extremely effective new tool with which to explore phenomena.

To sum up therefore the computer is proving of great practical assistance to mathematicians at all stages of their work, but perhaps most significantly in the exploratory or experimental stage. Great mathematicians of the past such as Euler or Gauss carried out large numbers of tedious calculations by hand in order to provide themselves with the raw material from which they could then guess some general law, or discover some remarkable pattern. As mathematics delves further and we become more ambitious the raw material becomes correspondingly much more messy and complicated. The computer can help us to sift this material and to point the way to further progress and understanding.

§4. The intellectual dangers

Few scientific advances are unmixed blessings and the computer is no exception. Having enumerated the many benefits which mathematicians, amongst others, can derive from the advent of the computer I would like now to draw attention to some possible dangers that lie ahead. Let me begin with the most central and insidious problem which is essentially the challenge the computer presents to the human intellect. Will mathematics continue as one of the highest forms of human endeavour or will it gradually be taken over by the computer? Who will remain in charge of mathematics and what are its criteria to be?

To illustrate the dangers I have in mind, let us consider an event that has already taken place, namely the solution by computer of a famous outstanding mathematical problem. I refer to the 4-colour theorem which says roughly that four colours suffice to colour any conceivable map of the world, the requirement being that adjacent countries must be coloured differently. This problem which dates from the last century was recently solved by a proof which involved a computer check of hundreds of different cases. On the one hand, this was a great triumph, the solution of a hard problem; on the other hand from an aesthetic point of view, it is extremely disappointing, and no new insights are derived from the proof.

Is this to be the way of the future? Will more and more problems be solved by brute force? If this is indeed what is in store for us should we be concerned at the decline of human intellectual activity this represents, or is that simply an archaic view-point which must give way before the forces of "progress"?

To answer such philosophical questions we must be bold and ask ourselves what is the nature and purpose of mathematical and scientific activity. The usual answer is that Science is man's attempt to understand, and perhaps eventually control, the physical world, but this leaves us with the difficult notion of "understanding". Can we be said to "understand" the proof of the 4-colour theorem? I doubt it.

For those who feel that "understanding" is too subjective and restrictive a term, the more limited goal of "description" may be preferred. Certainly I can describe the proof of the 4-colour theorem, though my description entails saying "the computer checked the following facts".

Such a "descriptive" attitude to mathematics could live happily with a gradual take-over by the computer, but I believe that this would lead to the atrophy of mathematics even measured by these modest "descriptive" standards. Mathematics is really an Art - it is the art of avoiding brute-force calculation by developing concepts and techniques which enable one to travel more lightly. Give a mathematician an infinitely powerful machine for doing calculations and you deprive him of his inner driving force. It is at least arguable, though somewhat far-fetched, that if computers had been available in say the fifteenth century, mathematics now would be a pale shadow of itself.

§5. Economic dangers

In addition to the subtle and intangible intellectual threat posed by computers there are much more obvious and practical dangers due to the tremendous economic importance of computers to society at large. Inevitably there will be vast financial pressures which will tend to push mathematics into new directions related to computing. Broadly speaking, more emphasis will be put on discrete mathematics as opposed to Calculus, which is concerned with continuous phenomena. No doubt, some of this pressure will be healthy and will stimulate and generate exciting new branches of mathematics, but the scale and tempo of the computer revolution are such that there is a real danger of the great classical tradition of mathematics being swamped.

Superficially, at first sight, discrete mathematics, which deals only with finite quantities and processes is easier and simpler than Calculus which deals with the infinite in various forms. However, it is one of the greatest triumphs of mathematics that infinity has been tamed and put to use, so that calculus is a tool of enormous power and elegance which has no serious rival or counterpart at the purely finite level. In fact, many important results of a discrete nature are best proved by the use of Calculus.

Until now the central position of Calculus, the Analysis of the infinite, has been unassailable not only in Pure Mathematics, but even more as a foundation for the application of mathematics to the whole of Science and Engineering. Courses in Calculus have provided the bed-rock of University education in the Mathematical Sciences. In recent years however this position has been called into question and there is an increasing call to reduce the role of Calculus

in scientific education and replace it with the kind of discrete mathematics more relevant to Computer Studies. To some extent this has already happened and it represents a necessary response to changing conditions, but I foresee pressures for much more radical changes which might be very damaging but would be difficult to resist.

It is possible that I am being unduly pessimistic on this score and certainly the dichotomy between discrete and continuous mathematics is not as sharp as I have implied. Traditionally we think of using finer and finer discrete quantities to approximate a continuous quantity in the way a continuous curve can be approximated by a large number of straight line segments. However, this procedure can equally well be reversed and continuous quantities can be regarded as approximations to discrete ones, provided the step size is sufficiently small. Thus we can use our knowledge (derived from Calculus) for the length of a circle to get an approximation for the length of a regular polygon with a large number of sides. In this way, as computers become more and more powerful and the numbers they deal with become larger and larger (or the time span for a single operation becomes shorter and shorter) so Calculus may again come into its own.

§6. Educational dangers

As we all know, the present economic scene is of widespread decline of traditional industries and the simultaneous growth of computer related industries. This is the economic side of the revolution. Naturally, this means that the best employment opportunities are linked to computers and this is altering the attitudes and expectations of all the younger generation. In schools and universities traditional studies are having to compete with the excitement and attraction of the computer, but mathematics, as the closest of the older discipline, is inevitably in the front line. This is having an effect at several levels. In the first place the pressure falls on the mathematics teachers in schools. Increasingly, they are having to take on computer studies as an additional responsibility and this means that mathematics teaching as such is suffering. Our educational institutions, for organizational and human reasons, can only change slowly and the sheer speed of the computer revolution is going to put them under very severe strain.

As far as students are concerned mathematics is going to be affected in two different ways. For the abler student who might have gone on to creative work in the higher reaches of mathematics, there is now the attractive alternative of entering a field which is in an explosive stage of its development and where the opportunities to make your mark are much greater. This means that the great creative minds of the past such as Newton, Gauss or Riemann might in future gravitate towards computer science rather than mathematics. For a subject so entirely dependent on brain-power this would be the greatest disaster of all. One has to hope that mathematics, by its power and beauty, will still attract intellects of quality

§7. Conclusion

I have tried to draw attention to the challenges and dangers which the rise of computers presents to mathematics. I am sorry if the picture I have been depicting appears too negative. It is easy to see the benefits which mathematics may derive from its association with computing, and so I have not thought it worthwhile to emphasize these at length. The dangers I think are more subtle and not so well recognized so it seemed appropriate to dwell on them in greater detail. To recognize possible dangers is to be fore-armed, and one can hope to prevent the worst from actually happening. Perhaps I can end on this note by recalling that George Orwell did not view his book on 1984 as a prediction but as a warning, deliberately exaggerated for dramatic effect, of what might happen if we were not careful.

in the future and that not all of them will be seduced by the computer.

For students of mathematics at a lower level, there are other dangers. At their most elementary these are simply that the wide-spread use of computers, or even sophisticated calculators, will lead to the view that arithmetic is no longer a necessary skill to acquire. Why learn your multiplication tables when, at the push of a button, the answer will appear on your screen? Such attitudes are already with us and there is much educational debate on say the advantages and dangers of having calculators in primary schools. As computers become ever cheaper and more powerful, they will flood into our schools and mathematics at all levels will constantly have to justify itself.

The enlightened response to these philistine attacks on mathematics is to say that, even when all the work can be done by pushing a button, you have to teach children which button to push. At the most basic level they have to know when to push the addition sign and when to push the multiplication sign. This means that there has to be more emphasis on understanding the processes involved and less on the performance of routine calculations. Properly interpreted this can be regarded as an educational advantage in which drudgery is removed and appreciation is enhanced. However, life is not quite so simple and any over-reliance on machines can lead to the atrophy of the human faculties involved, much in the way the motorcar has undermined the capacity of people to use their legs. Perhaps the sort of reaction which has made jogging so popular in recent years will in due course make exercise in mental arithmetic a form of mental therapy!

C O M M E N T

on the ICMI-Paper

"The Influence of Computers and Informatics
on Mathematics and its Teaching"

Rudolf Straesser, IDM Bielefeld

1. Introductory Remark

In my comments I will concentrate on an aspect of the ICMI discussion document I know best and I am most interested. I do not aim at a general evaluation of the discussion paper which might be done by other persons more competent in the field.

2. Main Idea of the Comment

On the whole I find it valuable that ICMI starts discussing special issues of international concern. Discussing the "influence of computers and informatics on mathematics" can be a good start of such necessary activities.

Nevertheless I want to point to a restriction in the discussion paper which can be harmful to the results obtainable: Especially questions 2 and 3 are limited "to the curriculum and teaching at university and pre-university level (from the age of 16 years)", as is stated on page 5. This limitation may be helpful at first glance - but totally leaves out the majority of the age group mentioned, namely those 16+-youth heading for non-academic careers. People in technical and vocational education are even more directly confronted with the impact of computers to their training - and these effects also play on the mathematics they have to learn in their technical and vocational education/instruction.

I will try to illustrate my point:

3. Illustration

When studying the "effect of computers on curricula" it is helpful to look for "the needs of society and the state of the discipline" (cf. page 10 of the document). The needs of society cannot be fully identified when only looking for the needs of other "disciplines - physicists, engineers, biologists, economists, etc" (cf. page 11). but also by looking for the needs of trades, qualified workers and professionals in non-academic jobs. The COCKCROFT-report and its research studies provide a valuable example of ways looking in this directions and they also provide valuable information on the impact of technological change - e.g. introduction of computers - on mathematics education.

I would like to add just one further example from the field to discuss here: Analysing (mathematical) needs coming up with CAD (computer assisted design) it seems clear that coordinate geometry (at least in three dimensions) becomes more important than geometry in the Euclidean style (study of figures, congruency etc.) because coordinate geometry underlies CAD-techniques. The case of CNC-machines (computer numerically controlled) seems to underline the argument I stated.

These changes in the needs can be most easily identified if you analyse changes in technology and organisation of work. They are more easily discerned than following the development of certain disciplines as studied in the discussion paper.

4. Final Remark

Question 3 on page 12 reveals another limitation of the discussion paper which should not be forgotten: Most of the arguments in the discussion paper apply for developed, industrialised countries - but forget the problems of mathematics education in rural and developing countries. Nevertheless the existence of computers and informatics in developed countries does affect developing countries. Have they just to follow the line of development of industrialised countries or do they need a preparation to the advent of computers before they are used in their countries for various purposes ?

6

4. AIMS AND GENERAL PHILOSOPHY OF THE SCHEME4.1. Vocational Aims

The pattern of demand for skilled personnel in the areas of information technology and management support services is becoming stabilised now that the worst of the economic recession appears to be ended. Although employment prospects in some traditionally well-established occupations may never return to their previous levels, it seems certain that opportunities for suitable graduates in the aforementioned areas will continue to increase in number. The overall aim of the proposed course structure is, therefore, 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, say, Data Processing, no longer provide an adequate foundation for professional employment in the later 1980's. Therefore, courses need to reflect the changes that have taken place and that are likely to take place in the immediate future.

The authoritative document upon which evaluation of those changes can be based is the Alvey Report. The content of new courses in both Mathematical and Computing Studies acknowledges the recommendations of the Alvey Committee.

Some aspects of the Alvey programme lie outside the sphere of activity of the department and belong more appropriately to the complementary discipline of electronic engineering. In those cases - VLSI providing a good example - there has been no attempt made to force content into the syllabi which does not naturally belong therein. However, the courses reflect the need to introduce mathematical rigour to the teaching of computing subjects, the orientation of applied mathematics to computer based applications and the importance of software engineering and knowledge processing to the development of information systems.

Although new directions in information technology are represented in the courses, 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.

In conclusion, the course scheme 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 places in the Information Technology and Management Support Services teams of those organisations.

The overall vocational aim of the new scheme is to strike an appropriate balance between the generalised knowledge and the 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 multidisciplinary team.
- (iv) Communicate their knowledge to others.

4.2.

Academic Aims and Philosophy

No one course can hope to encompass all that work that has been done and is being done in the computing and mathematical sciences, nor has a standard undergraduate curriculum yet emerged. Indeed, such is the rate of progress that a standardisation is not only not desirable but is also unobtainable.

Each course within the scheme aims to establish a sound theoretical and practical foundation in the broad areas of the computing and mathematical sciences respectively. Many of the staff have commercial and industrial experience and retain close contact with outside bodies (See Vol I, Part 2. 1.3); this background will provide the desirable practical orientation in their teaching.

The first two years of each course provide 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.

4.2.1.

Mathematical Studies

In addition to recognising that mathematical and problem solving techniques are finding wider and wider applications in commerce, the public services and industry the course in mathematical studies 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 the 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.

12
The final year of the course 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.

4.2.2.

Computing Studies

The course in computer studies aims to cover the range of knowledge and skills required by a computing specialist within a Management Information Services group. During the first two years, areas of computer studies are presented as three streams - a Systems Analysis and Design stream, a Programming stream and a Hardware/System Software stream.

Throughout the course the emphasis is laid on well-structured design of both hardware and software. Modern methodologies of System Design and Programming are presented in addition to techniques for rigorously proving and testing those designs (some of which are introduced as part of the core mathematics units). The first two years also act as a springboard for both the industrial year and the final year of the scheme. Students are offered a choice of final year units in which to specialise and the range of choice again covers the spectrum of computer studies. As ideas and technology advances, different final year options will be introduced to cater for whichever direction the industry takes.

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

(i) Have a basic theoretical and practical understanding of computer hardware, and software and of its applications.

(ii) Understand and apply systems analysis and

design methodologies to the production of a specification for a management information system which meets the requirements of a particular organisation.

(iii) Be able to develop, prove and test programs in a number of languages, both high and low level.

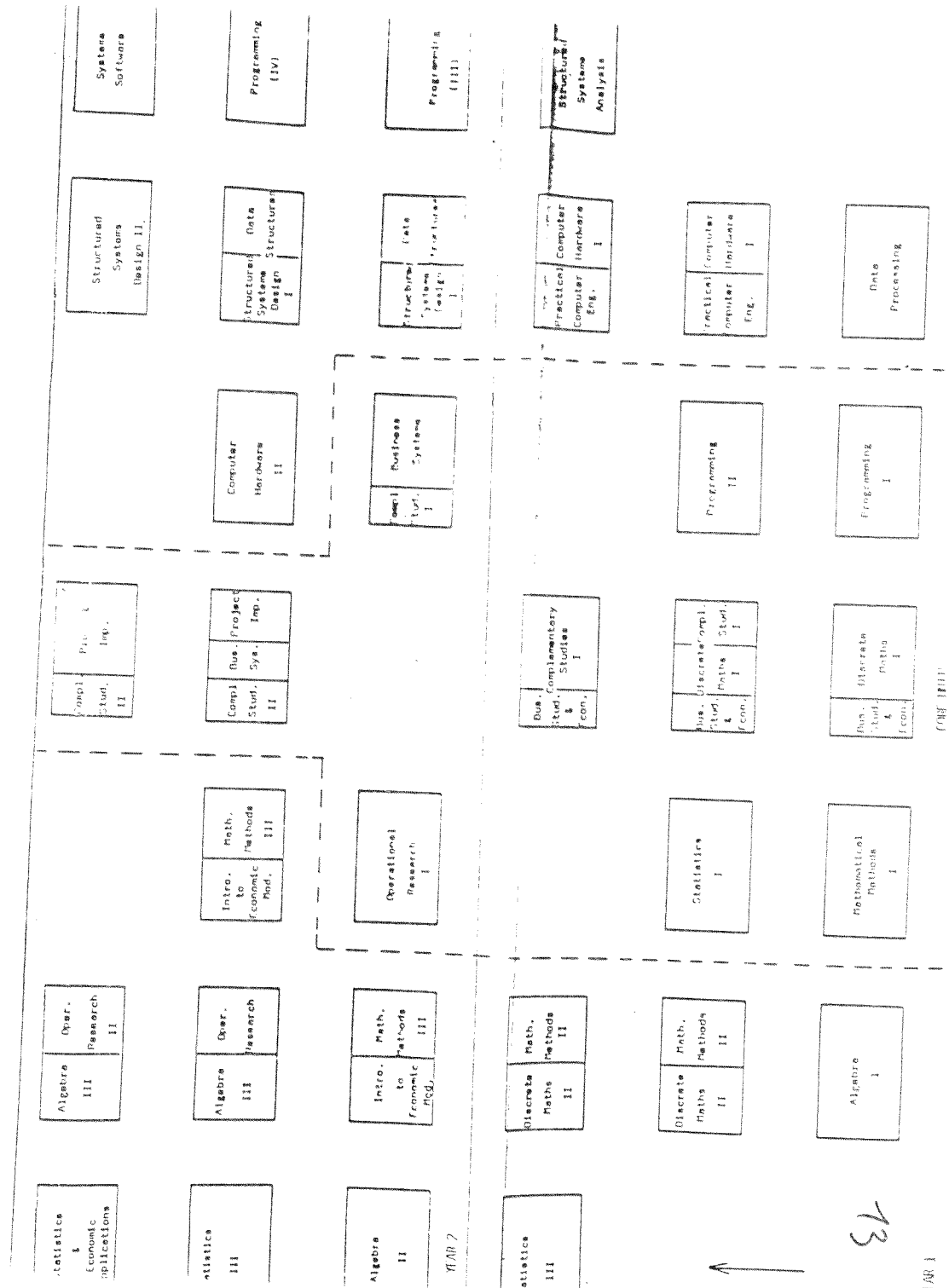
(iv) Understand human and social factors relating to an organisation and its environment and their role in the development of a computerised information system.

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

4.2.3.

The Degree

Although the above aims apply to both Degree and Degree with Honours students, the former will concentrate more in current techniques, technologies and applications, while the latter will study with an emphasis on the theoretical aspects of each area of the scheme. The same subject material will be presented during joint lectures but tutorial work and most examinations will be separate for the two groups.



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.
Algorithms 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:

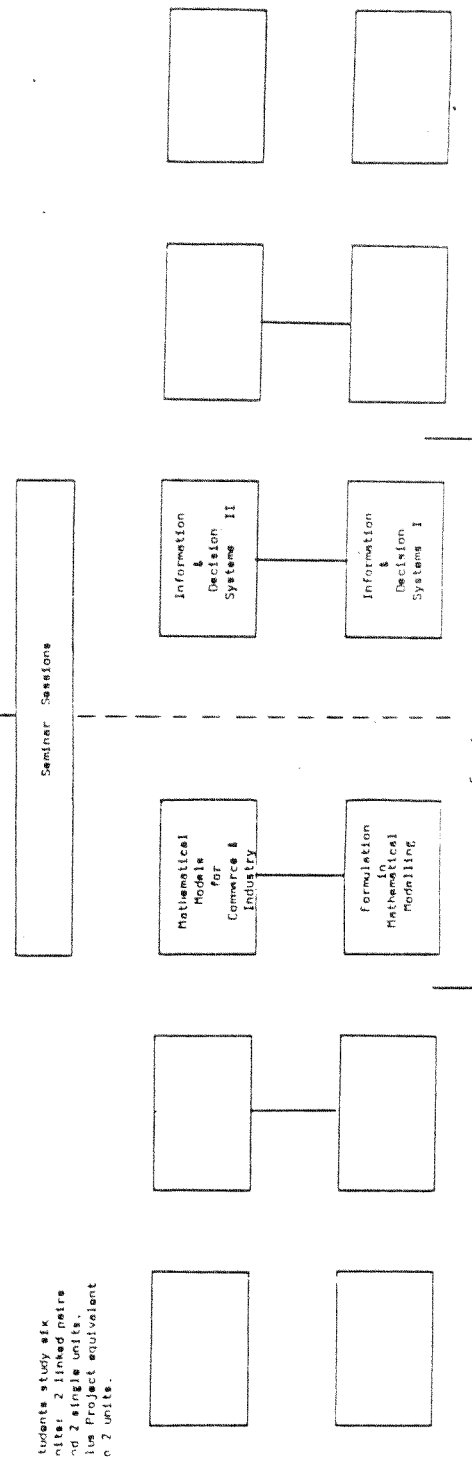
LIPSCITZ, S.,	Discrete Mathematics.
PRATHER, R. E.,	Discrete Mathematics for Computer Science. Houghton Mifflin, 1976.
FISHER, J. L.,	Application Oriented Algebra. Dun-Donnelly. 1977.
BIRKHOFF, G., and BARTEE, T. C.,	Applied Modern Algebra. McGraw Hill. 1970.

14

INTEGRATED DEGREE SCHEME

MATHEMATICAL STUDIES

COMPUTING STUDIES



Students study six units: 2 linked units and 2 single units. The Project equivalent is 2 units.

FINAL YEAR

YEAR 3

INDUSTRIAL PLACEMENT

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.

Lecture (33 hrs), Tutorial (16.5 hrs).

TEACHING METHOD:

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: 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: 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. First order languages. Interpretations. Satisfaction. Truth. Free and bound variables. Validity of formulae.

TEACHING METHOD:

Lecture (33 hrs), Tutorial (16.5 hrs).

See Discrete Mathematics I.

ASSESSMENT:

Examination (50).

READING LIST:

See Discrete Mathematics I.

15

UNIT TITLE: ALGEBRA II
YEAR AND TERM: Second Year. Term 1.
AIMS: See Algebra I.
OUTLINE SYLLABUS: Linear Vector Spaces.

Linear equations.
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: 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 adequacy theorem for L. Formal predicate calculus. The formal system K_1 . 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. Undecidability.

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 INFLUENCE OF COMPUTER AND INFORMATICS ON MATHEMATICS

Masaya Yamaguti

§ 0. I had tried to respond to the question: Did computer change mathematics or not. But at every period of our history, mathematics were always changing. Then it is hard to point out what part are really changed by computer 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 computer 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 brought to mathematics some integrating effect of different specialities in mathematics.

From the middle of our century, we experienced some separation between different specialities in mathematics. It has been continued to differentiate. But this tendency is now stopping by the influence of computers.

i) One of the most famous example is the study of non-linear dispersive wave propagation, particularly the soliton solution of K-d-V equation had been found as a result of numerical experimentation done by Kruskal and Zabusky and Miura. But after this finding, it becomes more and more pure mathematical object and now, we got some very rigorous mathematical argument for the whole structure of exact solutions of this kind of equations via Kac-Moody algebra.

The origin of this series of researches was some visualization performed by some computers.

The merite of this kind of visualization is enormous, because by the visualization of some possible facts, mathematicians in many different fields get common object of research. This causes a recovery of the unity in 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 that moment, I engaged in the study of chaotic dynamical system. 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 exists between Weierstrass func-

tion and this chaotic dynamical system. In reality, it was true that Weierstrass function is a superposition of these general solutions of the above dynamical system. And then I began a collaborative work with M. Hata about Takagi function which is also nowhere differentiable continuous found by T. Takagi just after Weierstrass. This function is more easy to define and it has clear explanation using the dynamical system:

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

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 n-th iterate of ψ . Now we got some general functional equation which has such kind of function as its solution.

Little later, we remarked that these functional equation had been studied by Julia, de Rham, and Moser without using dynamical system. But we also become aware of the related researches done just early period of this century, by Cesàro, Faber, Lebesgue, etc. That was through several works by de Rham. Finally, we found some finite difference scheme (multigrid) which are satisfied by these singular functions (See [1]). By this method, we succeeded to get a nice relation between Takagi func-

tion and Lebesgue's singular functions. Next step is the introduction of the concept "Invariant set of contractions" which was proposed by R.F. Williams and refound by 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, Pólya, Sierpiński, Osgood, ... about singular curves. And moreover, it contains many new figures which is a kind of Fractals (See [2]). Thus we are now feeling an arrival of new analysis without using differential calculus. All our researches was guided by computer experiments before our proof on each above step.

Recently, the advances in the research of Cellular Automaton which achieved by the progress of hardware (See [3]) is very near to our study. Yet we do not know exact interrelation between theirs and ours.

§ 2. Discrete Mathematics

At 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 you consider the mesh length tending to zero. And he wrote that this distinction arises in the study of non-linear problem. He gave as an example, the famous Clairaut differential equation, where usual discretization of Clairaut differential equation has a family of discretized solutions which tend to a smooth function different from any solution of original equation as their mesh size tend to zero.

Sometimes, it is said that difference equations are obviously tech-

nically and intellectually much simpler than their counterparts and that therefore, we can replace differential equation by difference equation and can establish a complete analogy between two things.

I am very much doubtful on these kind of 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 gave a start point to the study of chaotic discrete dynamical systems. 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 size [5]. Little later, we observed that centered difference scheme of the logistic differential equation gives the same phenomena for any mesh size. S. Ushiki completely proved this [6].

Thus, we can say in the case of non-linear differential equation, that the asymptotic behavior of the solution of differential equation

and the solution of the discretization of it are completely different. Then one can not replace each other.

Of course, one can construct complete analogue discretization for the differential equation. But for that, it is necessary to get very sophisticated difference scheme. For example, R. Hirota had a discretization which is very much 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 L_1 norm of the initial data as the original equation and L_2 norm in x of the solution dissipate as t increases. And then Hirota developed some calculus of difference operators which is completely analogous to differential calculus using very sophisticated way the operators Δ and Π . For example, he got

a formula of chain rules for this finite difference calculus and also complete analogue of many special functions. This way of construction of new analysis is seemingly very difficult. I can not recommend to teach for freshmen.

Now, I can propose one thing for the mathematical Education. My proposal is not to reduce elementary education of continuous theory, for example, that of differential equations. I recommend to do some preliminary numerical experiment with some discrete model before introducing differential equations. Because the students who learned the difficulties of a discrete theory can estimate well the easiness of continuous theory.

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.

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:

$$W_{a,b}(x) = \sum_{n=0}^{\infty} a^n \cos(\pi b^n x) \quad (0 \leq x \leq 1),$$

a, b real positive, $0 < a < 1$.

This can be represented, when $b = 2$,

$$(1) \quad W_{a,2}(x) = \sum_{n=0}^{\infty} a^n \cos(\pi \phi^n(x))$$

where $\phi(x) = 2x$ ($0 \leq x \leq 1/2$), $\phi(x) = 2(1-x)$ ($1/2 \leq x \leq 1$) and $\phi^n(x)$ means n -th iterate of $\phi(x)$. (Specially, $\phi^0(x) \equiv x$.)
Next example is Takagi's function [1]

$$(2) \quad T(x) = \sum_{n=1}^{\infty} \left(\frac{1}{2}\right)^n \phi^n(x).$$

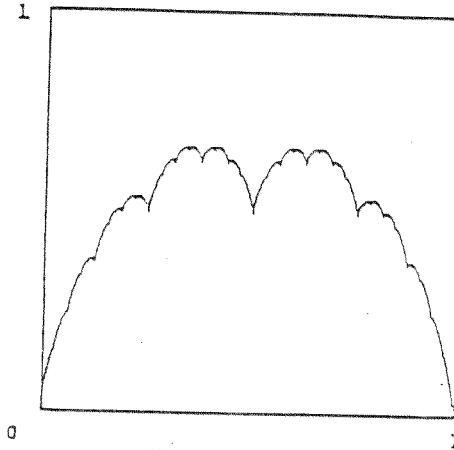


Figure 1. Graph of Takagi's function.

Remark. The above is not the original form of Takagi's function but we interpret the original definition using $\phi(x)$.

Both functions (1) and (2) satisfy the following functional equation:

$$(3) \quad F(t,x) = tF(t,\psi(x)) + g(x), \quad (0 \leq t < 1)$$

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:

$$(4) \quad \begin{cases} \frac{\partial F}{\partial t}(t,x) = \frac{\partial}{\partial t}(tF(t,\psi(x))) & (0 < t \leq 1), \\ F(0,x) = g(x). \end{cases}$$

Then we get the following theorem:

23

Theorem 1. Suppose that $g: [0,1] \rightarrow \mathbb{R}$ is a bounded function and that $\psi: [0,1] \rightarrow [0,1]$ 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

$$(5) \quad F(t,x) = \sum_{n=0}^{\infty} t^n g(\psi^n(x)).$$

We omit the proof because it is so easy.

Remark. This theorem is very general. For example, given functions $f(x)$ and $\psi(x)$ and a real value s ($0 < s < 1$), 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

$$(6) \quad F(s,x) = f(x).$$

Thus, we can obtain an expansion of usual Cantor function:

$$(7) \quad \sum_{n=0}^{\infty} \frac{1}{2^{n+1}} \chi_{[1/3,1]}(\phi^n(x))$$

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:

$$(8) \quad \begin{cases} M(x) = \alpha M(2x) & (0 \leq x \leq \frac{1}{2}) \\ M(x) = (1 - \alpha)M(2x - 1) + \alpha & (\frac{1}{2} \leq x \leq 1) \end{cases}$$

where α is a real number such that $0 < \alpha < 1$.

For comparison, we examine a special case of our equation for Takagi's function:

$$T(x) = \frac{1}{2}T(\phi(x)) + \frac{\phi(x)}{2}$$

which is rewritten in detail as below,

$$(9) \quad \begin{cases} T(x) = \frac{1}{2}T(2x) + x & (0 \leq x \leq \frac{1}{2}) \\ T(x) = \frac{1}{2}T(2(1-x)) + 1-x & (\frac{1}{2} \leq x \leq 1), \end{cases}$$

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 $\alpha \neq 1/2$. 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. Schauder expansion

As an analogy of Fourier expansion, we have Schauder expansion of all continuous function on the closed interval $[0, 1]$. The basis function $F_{\alpha, \beta}^{i/2^k, (i+1)/2^k}(x)$ is obtained from the function $F_{\alpha, \beta}(x)$

which is defined as follows:

$$(10) \quad F_{\alpha, \beta} = \frac{1}{\beta - \alpha} (|x - \alpha| + |x - \beta| - |2x - \alpha - \beta|).$$

Our bases are the following sequence of functions:

$$(11) \quad 1, x, F_{0,1}, F_{0,1/2}, F_{1/2,1}, \dots, F_{i/2^k, (i+1)/2^k}, \dots$$

Theorem 2. Any continuous function $f(x)$ on $[0, 1]$ can be expanded uniquely as follows:

$$(12) \quad f(x) = f(0) + [f(1) - f(0)]x + \sum_{k=0}^{\infty} \sum_{i=1}^{2^k-1} a_{i,k}(f) F_{i/2^k, (i+1)/2^k}(x)$$

where the coefficients $a_{i,k}$ are

$$(13) \quad a_{i,k}(f) = f\left(\frac{2i+1}{2^{k+1}}\right) - \frac{1}{2}\left(f\left(\frac{i}{2^k}\right) + f\left(\frac{i+1}{2^k}\right)\right).$$

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

$$(14) \quad f(x) = f(0) + [f(1) - f(0)]x + \sum_{k=0}^{\infty} b_k(x) \phi^{k+1}(x)$$

where $b_k(x) = \sum_{i=0}^{2^k-1} a_{i,k}(f) \chi_{[i/2^k, (i+1)/2^k]}(x)$.

Therefore, 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

$$(15) \quad \forall k, \quad b_k(x) = \frac{1}{2^{k+1}} \quad (0 \leq x \leq 1),$$

which means that $T(x)$ satisfies the following boundary value problem for a multigrid finite difference scheme because of (13).

$$(16) \quad \begin{cases} T\left(\frac{2i+1}{2^{k+1}}\right) - \frac{1}{2}\left(T\left(\frac{i}{2^k}\right) + T\left(\frac{i+1}{2^k}\right)\right) = \frac{1}{2^{k+1}}, & (0 \leq i \leq 2^k - 1, \quad \forall k) \\ T(0) = T(1) = 0. \end{cases}$$

Remark. 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 $u(0) = u(1) = 0$.

The following theorem suggests that Takagi's function can be generalized to some nice class.

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

$$(17) \quad \sum_{n=1}^{\infty} |c_n| < +\infty.$$

The proof of sufficiency is easy. The necessity 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 [1]). 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 $\lim_{n \rightarrow \infty} |2^n c_n| \neq 0$, then $f(x)$ has no derivative everywhere. 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:

$$(18) \quad \begin{cases} T_G(\frac{2i+1}{2^{k+1}}) = \frac{1}{2}(T_G(\frac{i}{2^k}) + T_G(\frac{i+1}{2^k})) + c_k \\ T_G(0) = T_G(1) = 0, \quad 0 \leq i \leq 2^k - 1, \quad \forall k \end{cases}$$

where $\{c_n\} \in l_1$, and $\lim_{n \rightarrow \infty} |2^n c_n| > 0$.

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:

$$(19) \quad \begin{cases} M_\alpha(\frac{2i+1}{2^{k+1}}) = (1-\alpha)M_\alpha(\frac{i}{2^k}) + \alpha M_\alpha(\frac{i+1}{2^k}) \\ M_\alpha(0) = 0, \quad M_\alpha(1) = 1, \quad 0 \leq i \leq 2^k - 1, \quad \forall k \end{cases}$$

the proof is easy using the equation (8).

This boundary value problem is closely related to some singular perturbation problem:

$$(*) \quad \begin{cases} \epsilon \frac{d^2 u}{dx^2} - \frac{du}{dx} = 0 & \epsilon \text{ small } > 0 \\ u(0) = 0, \quad u(1) = 1. \end{cases}$$

because our problem (18) can be rewritten as follows:

$$(**) \quad \begin{cases} \frac{1}{4 \cdot 2^{k+1}} \frac{M_\alpha((i+1)/2^k) - 2M_\alpha((2i+1)/2^{k+1}) + M_\alpha(i/2^k)}{(1/2^k)^2} \\ + (\alpha - \frac{1}{2}) \frac{M_\alpha((i+1)/2^k) - M_\alpha(i/2^k)}{\frac{1}{2^k}} = 0, \\ M_\alpha(0) = 0, \quad M_\alpha(1) = 1, \quad 0 \leq i \leq 2^k - 1, \quad \forall k \text{ and } 0 < \alpha < \frac{1}{2}. \end{cases}$$

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.

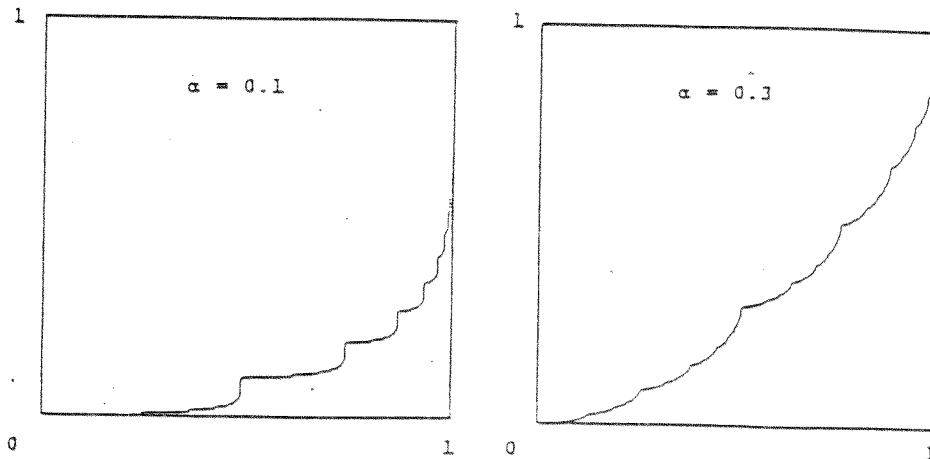


Figure 2. Graphs of Lebesgue's singular function M_α .

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 :

$$(20) \quad a_{i,k}(M_\alpha) = (\alpha - \frac{1}{2}) [M_\alpha(\frac{i+1}{2^k}) - M_\alpha(\frac{i}{2^k})].$$

With this equality, we can obtain the relation

$$(21) \quad \begin{cases} a_{2m,k} = \alpha a_{m,k-1} \\ a_{2m+1,k} = (1-\alpha) a_{m,k-1} \end{cases}$$

Theorem 4.

$$a_{i,k}(M_\alpha) = (\alpha - \frac{1}{2}) \alpha^p (1-\alpha)^q$$

where $p+q=k$, p is the number of 0's in the binary expansion of

$$i = \sum_{j=1}^k w_j 2^{j-1} \quad (0 \leq i \leq 2^k - 1),$$

q is the number of 1's.

25

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

$$2T(x) = \left. \frac{\partial M_\alpha(x)}{\partial \alpha} \right|_{\alpha=1/2}$$

Theorem 5.

$$\left. \frac{\partial M_\alpha(x)}{\partial \alpha} \right|_{\alpha=1/2} = 2T(x).$$

6. Physical meaning of $M_\alpha(x)$ and $T(x)$

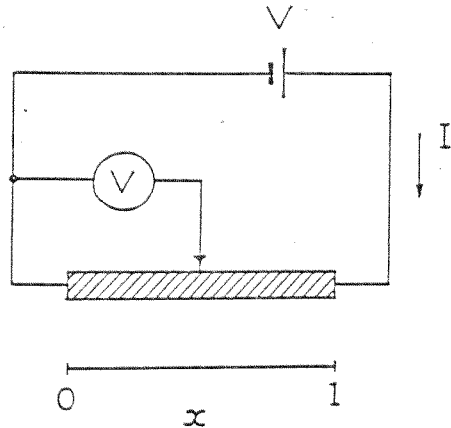


Figure 3.

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.

$$(22) \quad E(x) = R(x)I$$

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

$$R(x) = \kappa \rho(x) \quad \kappa : \text{constant,}$$

then $V(x)$ voltage at point x , becomes

$$(23) \quad V(x) = \int_0^x E(x') dx' = \kappa \int_0^x \rho(x') dx'$$

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 ($0 \leq \alpha \leq 1$). It is defined by a limit of cascade of the coarse grained distribution $\rho_\alpha^{(k)}(x)$ which are defined by

$$(24) \quad \begin{cases} \rho_\alpha^{(k+1)}\left(\frac{2i}{2^{k+1}}\right) = \alpha \rho_\alpha^{(k)}\left(\frac{i}{2^k}\right) \\ \rho_\alpha^{(k+1)}\left(\frac{2i+1}{2^{k+1}}\right) = (1-\alpha) \rho_\alpha^{(k)}\left(\frac{i}{2^k}\right) \\ \rho_\alpha^{(0)}(0) = 1. \end{cases} \quad (0 \leq i \leq 2^k - 1)$$

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),

$$(25) \quad V(x) = \kappa \int_0^x \rho_\alpha(x') dx' = \kappa M_\alpha(x)$$

where $M_\alpha(x)$ is Lebesgue's singular function appeared in preceding section 5.

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

$$j(t,x) = -\int_0^x \frac{\partial \rho}{\partial t}(x',t) dx'.$$

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

$$\begin{aligned} j(x,0) &= \lim_{\Delta t \rightarrow 0} \int_0^x \frac{\rho_{1/2+\alpha'\Delta t}(x') - \rho_{1/2}(x')}{\Delta t} dx' \\ &= -\alpha' \left. \frac{\partial}{\partial \alpha} M_\alpha(x) \right|_{\alpha=1/2} \\ &= -2\alpha' T(x) \end{aligned} \quad (\text{Theorem 5})$$

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

$$I(t) = \prod_{n=1}^{\infty} \left\{ \alpha + (1 - \alpha) e^{\frac{it}{2^n}} \right\}$$

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

$$I(t) = e^{it} \prod_{n=1}^{\infty} \left\{ \cos \frac{t}{2^{n+1}} + (1 - 2\alpha) i \sin \frac{t}{2^{n+1}} \right\}.$$

References

- [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. Jnl. (1983) to appear.
- [3] Hata, M. and Yamaguti, M., On Takagi's function, Japan Jnl. of Applied Math. (New Jnl.) to appear.
- [4] De Rham, G., Sur quelques courbes définies par des équations fonctionnelles, 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.

Über ergodische, nicht-monotone lernende Automaten

C. Kuck, Universität-GHS Paderborn, Fachbereich 17
(Mathematik-Informatik), Warburger Straße 100, D-4790 Paderborn

Zusammenfassung: Ein zeitabhängige Aussagenlogik ist eine Lie Algebra. Kleene's Theorem kann man verallgemeinern: entscheidbar ist äquivalent zu nicht-monoton und rational. Es gibt künstliche Intelligenz. Es gibt nicht-monotone und ergodische Beweissysteme. Es gibt nicht-monotone Kategorien von $\mathbb{1}^e$ -Semimoduln. Gödel's Unvollständigkeitssätze gelten nur in monotonen Beweissystemen. Hilbert's Programm ist vollendet. Widerspruchsbeweise sind nicht zulässig. Wir kennen den Zahlentyp der Quadratwurzel aus 2 nicht. Widerspruchsfreiheit ist äquivalent zur Entscheidbarkeit, das heißt die Prädikatenlogik ist nicht widerspruchsfrei.

Summary: A time dependent propositional logic is a Lie algebra. Kleene's Theorem can be generalized: decidable is equivalent to non-monotonic and rational. Artificial Intelligence exists. There are non-monotonic and ergodic proof systems. There are non-monotonic categories of $\mathbb{1}^e$ -semimoduls. Gödel's Incompleteness Theorems are only valid in monotonic proof systems. Hilbert's Program is completed. Refutation proofs are not permissible. We do not know, what type of number is the square root of 2. Consistency is equivalent to decidability, that means predicate calculus is not consistent.

1. Definition Wir definieren für die Ersetzung

$$((\alpha_2, \beta_2), (\alpha_1, \beta_1)) \rightarrow (\beta_2, \beta_1)$$

die Zuordnung $Z: (p(\alpha_2, t_1), p(\alpha_1, t_2)) \rightarrow \text{smax}(1 - p(\alpha_2, t_1), p(\alpha_1, t_2))$ geht für den Schwellwert $S = 1$ in die Implikation der Aussagenlogik über [1]. Die Hintereinanderausführung zweier Ersetzungen ist ein nicht kommutatives Produkt, das wir mit min bezeichnen. min ist transitiv [1]

Als zweite Operation führen wir die Auswahl max von Ersetzungen ein. min und max arbeiten auf einer lokal finiten rationalen Menge Σ von Produktionsregeln. Mengen verschiedener Ableitungen des gleichen Satzes mit gleichen Massen bestehen aus äquivalenten Elementen. Die Auswahl ist eine Kongruenzrelation auf dieser Quotientenhalbgruppe mit dem neutralen Element 0 . Nach Einführung einer skalaren Multiplikation spezialisieren wir das kommutative Monoid zu einem Vektorraum [1].

Ersetzungen sind Funktionale. Die Multiplikation zweier Funktionale erfüllt die drei Bedingungen für die Lie Klammern [1]. Damit haben wir aus der zeitabhängigen Aussagenlogik ein Lie Inferenz- und Problemlösesystem $\mathbb{1}_0 = L/K$ konstruiert. K ist ein Ideal, das von den Elementen

$$[h_i, h_j] = 0, [e_i, f_j] = \delta_{ij} q_{ij} h_i, [e_i, h_j] = \beta_i(h_j) e_i,$$

$$[f_i, h_j] = \beta_i(h_j) f_i \quad \text{für alle } i, j, q_{ij} \in [0, 1[$$

erzeugt wird. $q_{ij} h_i$ ist eine Notation, die daran erinnern soll, daß die abgeleiteten Sätze ein Mass haben.

Die zeitabhängige Lie Multiplikation ist eine nicht-monotone Verallgemeinerung der Implikation. Wir beweisen, dass unser Inferenzsystem und unsere nicht-monotonen und rationalen Theorien widerspruchsfrei sind, und dass Widerspruchsfreiheit äquivalent zur Entscheidbarkeit ist. Mathematik ist nicht-monoton. Die Prädikatenlogik ist nicht widerspruchsfrei [1].

Wir zeigen zunächst durch Verallgemeinerung des Kleene'schen Theorems, dass es Übergangsmatrizen E^+ gibt, die zu Cartan Matrizen äquivalent sind. Für Cartan Matrizen gilt $A_{ii} = 2, A_{ij} \in -\mathbb{Z}_+$, $\forall i \neq j$ und $A_{ij} = 0 \Rightarrow A_{ji} = 0$. $-\mathbb{Z}_+$ ist zu \mathbb{Z}_+ isomorph, \mathbb{Z}_+ ist zu $2\mathbb{Z}_+$ und $\mathbb{Q}[0,1[$ isomorph.

Wir definieren nicht-monotone, rationale $(\underline{1}_0)^-$ -Theorien, die die $(\underline{1}_0)^+$ -Operation nicht benutzen [1]

$$\text{Th}^-(\tilde{A}_j) = \bigcup_{i=1}^n \{L\} \cup \{\Sigma_{ci} \mid (\underline{1}_0)^- : \Sigma_{ci} \rightarrow \Sigma_{ci}\}$$

2. Satz $\text{Th}^-(\tilde{A}_j)$ ist entscheidbar, wenn die $S_{ci} = \text{const.}$.

Beweis: Wir ordnen die Teilmengen Σ_{ci} der Sprache L mit Hilfe von Massfunktionen und Schwellwerten $S_{ci} \in [S_{c1}, 1]$, so an, dass

$$\Sigma_{cn} \subset \Sigma_{c,n-1} \subset \Sigma_{c,n-2} \subset \dots \subset \Sigma_{c1} \subset L$$

$$0 \leq S_{c1} \leq S_{c2} \leq \dots \leq S_{cn} \leq 1.$$

(Die 1 kriegen wir später wieder, weil die Stringlängen der Produktionsregeln endlich sind.)

Die Menge $\{\Sigma_{ci}\}$ ist unter $(\underline{1}_0)^-$ abgeschlossen, wenn die Schwellwerte S_{ci} konstant sind, weil die Σ_{ci} endlich $((\underline{1}_0)^+)$ wird nicht benutzt) und die Stringlängen l der Elemente $f \in \Sigma_{ci}$ beschränkt sind. $(\underline{1}_0)^-$ wählt aus allen möglichen Permutationen die Inferenzketten aus, deren Ersetzungen feuern. Also sind auch

$$\bigcup_{i=0}^n \Sigma_{ci} = \Sigma_{cn} \quad \text{und} \quad \bigcup_{i=0}^n \Sigma_{ci} = L \quad (i=0 \text{ steht für } L)$$

abgeschlossen, denn das Alphabet F von L ist endlich und

$$L = \bigcup_{i=0}^n \Sigma_{ci} = F^* = \{f_1, \dots, f_n \mid n \geq 0, f_j \in F\}$$

ist abgeschlossen, weil die Σ_{ci} abgeschlossen sind und n endlich ist. \square

Wir definieren $\text{Th}^+(\tilde{A}_j)$ analog $\text{Th}^-(\tilde{A}_j)$.

3. Satz $(\underline{1}_0)^{(n)}(\Sigma_{ci}) = (\underline{1}_0)^*(n \Sigma_{ci}^-)a$ konvergiert nach n Anwendungen von $(\underline{1}_0)^+$ gegen einen Fixpunkt mit der Multiplizität 1. Die Masse (Gewichte) in $(\underline{1}_0)^+(\Sigma_{ci})$ sind minimal.

Beweis:

Wir gehen davon aus, dass jede Problemlösung mit einer Ersetzung (einem Zustandsübergang) endet

$$(\underline{1}_0)^+(\Sigma_{ci}) = (\underline{1}_0)^*(\Sigma_{ci})a$$

und, dass Problemlösungen shuffle-Produkte sein können ($a \in \Sigma_{ci}$, $b \in \Gamma_{ci}$, c_i Potenzen von a oder 1), die ebenfalls mit Zustandsübergängen enden ($()^+$ ist eine Vereinigung)

$$((\underline{1}_0)^*(\Sigma_{ci})b)^+ = (a + b)*b = c_1 b c_2 b c_3 b \dots c_n b.$$

Aus den beiden Axiomen folgt für $\Sigma_{ci} = \Gamma_{ci}$

$$\begin{aligned} ((\underline{1}_0)^{(n)}(\Sigma_{ci}))^+ &= ((\underline{1}_0)^*(n \Sigma_{ci}^-)a)^+ = (\underline{1}_0)^*(na + a)a = \\ &= (\underline{1}_0)^*((n+1)a)a \quad \text{und daraus mit Hilfe des 1. Axioms} \end{aligned}$$

$$((\underline{1}_0)^*(\Sigma_{ci}^-)a)^+ = (\underline{1}_0)^*(a)a = (\underline{1}_0)^+(a) \quad \square$$

4. Satz Sei $E^+ = E + E^2 + E^3 + \dots + E^n$ die Summe der Übergangsmatrixmultiplikationen mit \min und \max . Die Elemente der Matrizen sind rationale Mengen

$$E_{rq} = \bigcup_{r,q} \underline{1}_0(\Sigma_{ci}).$$

Dann konvergiert jedes Element $E_{rq}^{(n)}$ gegen einen Fixpunkt.

Beweis: Satz 3.

5. Satz Sei X ein lokal finites Monoid, dann ist die Menge Σ , deren Elemente $(\underline{1}_0)$ -Untermengen Σ_{ci} von X sind, unter den nicht-monotonen und rationalen Operationen abgeschlossen.

Beweis: $\Sigma_i \Sigma_{ci}, \Sigma_{ci} \Sigma_{ci}, \Sigma_{ci} \Gamma_{ci}, k \Sigma_{ci}, k + \Sigma_{ci}$ liegen in $(\underline{1}_0)^+$

$$(\underline{1}_0)^+(\Sigma_{ci}^-) = \Sigma_{ci}^- \text{ nach Satz 3.} \quad \square$$

6. Theorem Sei P eine Menge abgeschlossener Intervalle $[S_{ci}, 1]$, $0 \leq S_{ci} \leq 1$ und F ein endliches Alphabet. Dann konvergieren die nicht-freien $\underline{1}_0$ -Teilmengen Σ_{ci} von X^- nach Satz 4. gegen Fixpunkte

$$(\underline{1}_0)^+(\Sigma_{ci})_{rq}$$

Diese Fixpunkte sind genau dann entscheidbar, wenn sie nicht-monoton und rational sind.

Beweis:

1. entscheidbar \Rightarrow nicht-monoton und rational

Die Elemente

$$E_{rq} = \bigcup_{r,q} \underline{1}_0(\Sigma_{ci})$$

von E bilden die Klasse aller $\underline{1}_0$ -Untermengen Σ_{ci} von X^- . $\underline{1}_0(\Sigma_{ci})$ ist nach Satz 5. abgeschlossen.

Die E_{rq} sind nach Satz 2. entscheidbar. Die $(\underline{1}_0)^+(E_{rq})$ sind nicht-monoton und rational. Satz 4. sagt

$$E_{rq} = (\underline{1}_0)^+.$$

2. nicht-monoton und rational \Rightarrow entscheidbar

Sei $\underline{1}_0(\Sigma_{ci})$ ein nla. Seine Zustände seien $Z = \{1, \dots, n\}$. Wir betrachten die nach Satz 2. entscheidbaren $(\underline{1}_0)^-$ -Untermengen

$$(\underline{1}_0)^- = \{(Z, r, q) \mid r, q \in Z\}.$$

Da

$$|\underline{1}_0| = \bigcup_{r,q} (\underline{1}_0)^+(\Sigma_{ci}^-)$$

genügt es zu zeigen, dass alle $(\underline{1}_0)^-$ -Untermengen in $(\underline{1}_0)^+$ liegen. Sei für irgendeine ganze Zahl $0 \leq k \leq n$ $C_{rq}^{(k)}$ die Menge der nicht trivialen Pfade

$$c: r \rightarrow q,$$

so dass für jede Zerlegung

$$r \xrightarrow{c_1} l \xrightarrow{c_2} q$$

die Ungleichung $l \leq k$ erfüllt ist. Sei $B_{rq}^{(k)}$ die Vereinigung der Wörter $\{c\}$ in $C_{rq}^{(k)}$. Dann haben wir

$$\bigcup_{r,q} (\underline{1}_0)^-(\Sigma_{ci}) = B_{rq}^{(0)} = E_{rq}^{(0)}$$

$$\text{und } \bigcup_{r,q} (\underline{1}_0)^{(n)}(\Sigma_{ci}) = B_{rq}^{(n)} = E_{rq}^{(n)}.$$

In den letzten beiden Gleichungen haben wir das k oben weggelassen und dafür einen Index in die Klammer oben gesetzt, der die wiederholte Anwendung von $(\underline{1}_0)^-$ anzeigt.

Die wiederholte Anwendung des Inferenzsystems $(\underline{1}_0)^-$ ergibt $(\underline{1}_0)^+$.

3
7

$(\underline{1}_0)^+$ ist nach Satz 3 ein Fixpunkt und es gilt

$$(\underline{1}_0)^+(a)a = (\underline{1}_0)^+(a).$$

Die letzte Gleichung sagt, dass die rationalen Ausdrücke in $(\underline{1}_0)^+(a)$ mit einer Ersetzung enden. Das bedeutet, dass die wiederholte Anwendung des Inferenzsystems $(\underline{1}_0)^-$ (einschliesslich seiner Negation [1]) eine Faktorisierung des Pfades

$$c: r \rightarrow q \quad \text{in}$$

$$r \xrightarrow{d} k \xrightarrow{c_1} k \xrightarrow{c_2} k \rightarrow \dots \rightarrow k \xrightarrow{c_n} k \xrightarrow{e} q$$

impliziert. Diese Zerlegungsmöglichkeit sagt uns, dass die nicht-monotonen und rationalen Untermengen $B_{rq}^{(n)}$ eindeutig in entscheidbare $\underline{1}_0^-$ -Untermengen $B_{rq}^{(0)}$ zerlegt werden können, wenn $0 \leq k \leq n$. \square

Es gibt 3 Korollarien zu diesem Ergebnis.

1. Es gibt künstliche Intelligenz.
2. Es gibt eine endliche Zahl $n_i \in \mathbb{N}_0$, für die

$$\frac{n_i + 1}{f_i} \cdot \sigma_0 \geq S_{ci},$$

σ_0 ist ein Element mit minimalem Mass aus Σ_{ci} .

3. Jede Inferenzkette in $(\underline{1}_0)$ und jeder Wurzelstring im Dualraum $(\underline{h})^*$ ist endlich, d. h. $(\underline{1}_0)^+(\Sigma_{ci})$ ist S_{ci} -lösbar.

Damit haben wir **gezeigt**, dass es Übergangsmatrizen E^+ gibt, die zu Cartan Matrizen äquivalent sind. E^+ -Matrizen sind symmetrisch, weil die Massfunktionen symmetrisch sind. Wir bauen auf E^+ -Matrizen ein Kac-Moody Lie Inferenz- und Problemlösesystem auf.

32

Ein Kac-Moody Lie Inferenz- und Problemlösesystem ist ein ergodisches, nicht-monotones Beweissystem, in dem die Gödel'schen Unvollständigkeitssätze nicht gelten. Wir beweisen jetzt diese Behauptung.

Dazu bilden wir den Mustervektorraum A der Produktionsregeln mit

$$\tilde{\alpha}_i \left(\sum_{i=1}^n d_i \alpha_i \right) = \phi_i(h)$$

in $(\underline{h})^*$ ab. Die Masse $\mu \in (\underline{h})^*$ sind Entropien der Mustervektoren aus A . Die Ersetzungen nach Def. 1 erzeugen in $(\underline{h})^*$ Folgen rationale Zahlen aus $[0,1]$. Wir erweitern \underline{h} durch abgeleitete Endomorphismen \underline{h}^e zu \underline{h}^e und konstruieren in $(\underline{h}^e)^*$ endliche, irreduzible und aperiodische Markoff-Ketten, d. h. nicht-monotone, ergodische, lernende Automaten (nla). Ein nla ist ein nicht-deterministischer Automat, der gegen einen deterministischen Automaten konvergiert.

7. Satz Wir setzen einen endlichen Zustandsraum C und eine endliche Markoff-Kette MC voraus. Durch Negation [1] der Ersetzung, deren Mass das Minimum erzeugt, geht

$$\lim_{t \rightarrow \infty} p_{ij}(t) = p_{ij}.$$

Beweis: Sei $M_{ij} := \min p_{ij}(t_2)$ das Mass einer Ableitungsfolge und $m_{kl} := \max p_{kl}(t_4)$ das Mass der Ersetzung in der Ableitungsfolge, das durch Negation verbessert wird. Dann gilt

$$(7.1) \quad M_{ij}(t_2) \leq M_{ij}(t_1), \quad m_{kl}(t_4) \geq m_{kl}(t_3), \quad t_4 > t_3 > t_2 > t_1.$$

Das Gleichheitszeichen in der zweiten Ungleichung gilt für das Mass 1. Wir beobachten jetzt abwechselnd eine vollständige Ableitungs- und die zu ihr gehörende Lernfolge, und heben den Schwellwert $S_{ci} \in [0, \frac{1}{2}]$ jeweils auf das verbesserte Mass der Ableitungsfolge an. Dann gibt es nach Satz 6. rationale Zahlen M_{ij} und m_{kl} , so dass

$$\lim_{n \rightarrow \infty} M_{ij}(n) = M_{ij}, \quad \lim_{n \rightarrow \infty} m_{kl}(n) = m_{kl}.$$

Wir müssen jetzt beweisen, dass $m_{kl} = M_{ij}$. Dazu betrachten wir die Abstandsfolge

$$d_{Mm}(n+1) = m_{kl}(n+1) - M_{ij}(n)$$

und zeigen, dass

$$(7.2) \quad \lim_{n \rightarrow \infty} d_{Mm}(n) = 0.$$

Aus den Gl. (7.1) folgt

$$0 \leq d_{Mm}(t_2) \leq d_{Mm}(t_1), \quad t_2 > t_1,$$

dass heisst $\{d_{Mm}(n)\}$ ist eine monoton nicht wachsende Folge. Um Gl. (7.2) zu beweisen, genügt es daher zu zeigen, dass

$$\lim_{n \rightarrow \infty} d_{Mm}(nN) = 0.$$

Da unsere Markoff-Ketten ergodisch sind, gibt es eine natürliche Zahl N , so dass für alle i, j

$$p_{ij}(N) > 0.$$

Sei

$$c = \min_{i,j} p_{ij}(N), \quad 0 < c < \frac{1}{2}.$$

Wir beweisen, dass für jedes $n = 1, 2, \dots$

$$(7.3) \quad d_{Mm}((n+1)N) \leq (1-2c)d_{Mm}(nN),$$

weil aus (7.3) folgt, dass

$$d_{Mm}(nN) \leq (1-2c)^{n-1} d_{Mm}(N).$$

Für jeden Zustand i in C gilt

$$(7.4) \quad p_{ik}((n+1)N) = \sum_{r \in C} p_{ir}(N)p_{rk}(nN),$$

weil \min zum Produkt und \max zur Addition isomorph ist. Gl. (7.4) sagt, dass man jede Ableitung in die Hintereinanderausführung zweier Ersetzungen faktorisieren kann. Jetzt betrachten wir nacheinander a.) eine Ableitungs- und b.) eine Lernfolge

a.) Ableitungsfolge

Da das Minimum mitgenommen wird, gilt für den ersten Zustand i und den letzten Zustand k einer Ableitungsfolge

$$p_{ik}((n+1)N) = M_{ik}((n+1)N)$$

und für irgendeinen Zustand q

$$p_{qk}(nN) = m_{qk}(nN), \quad \text{s.o.}$$

Aus Gl. (7.4) folgt, dass

$$\begin{aligned} M_{ik}((n+1)N) &= p_{iq}(N)p_{qk}(nN) + \sum_{r \neq q} p_{ir}(N)p_{rk}(nN) \\ &\geq p_{iq}(N)m_{qk}(nN) + M_{rk}(nN) \sum_{r \neq q} p_{ir}(N), \end{aligned}$$

weil $M_{rk}(nN) \leq p_{rk}(nN)$ ist, s.o. Weiter folgt (nach geeigneter Auswahl der Ersetzungen und Normierung), dass

$$M_{ik}((n+1)N) \geq p_{iq}(N)m_{qk}(nN) + M_{rk}(nN)(1 - p_{iq}(N)).$$

Schliesslich ergibt sich

$$(7.5) \quad M_{ik}((n+1)N) \geq M_{rk}(nN) - (M_{rk}(nN) - m_{qk}(nN))c.$$

33

b.) Lernfolge

Eine Lernfolge verbessert das Minimum in a.). Wenn q der erste Zustand einer Lernfolge ist (das Minimum der zu ihr gehörenden Ableitungsfolge), dann gilt

$$p_{qk}(nN) = M_{qk}(nN),$$

M_{qk} ist das Minimum der Lernfolge. Weiter ist in

$$p_{ik}((n+1)N) = m_{ik}((n+1)N)$$

m_{ik} das verbesserte Mass der Ableitungsfolge, das entweder durch Negation oder durch ein neues Minimum erzeugt wird. Aus Gl. (7.4) folgt für das verbesserte Mass m_{ik}

$$\begin{aligned} m_{ik}((n+1)N) &= p_{iq}(N)p_{qk}(nN) + \sum_{r \neq q} p_{ir}(N)p_{rk}(nN) \\ &\leq p_{iq}(N)M_{qk}(nN) + m_{rk}(nN) \sum_{r \neq q} p_{ir}(N), \end{aligned}$$

$M_{qk}(nN)$ ist das Minimum der Lernsequenz, der eventuell mehrere andere Lernsequenzen an der Stelle $p_{qk}(nN)$ vorausgingen, das grösser als das neue Minimum $m_{rk}(nN)$ der Ableitungsfolge ist. Damit ergibt sich (nach geeigneter Auswahl der Ersetzungen und Normierung), dass

$$\begin{aligned} m_{ik}((n+1)N) &\leq p_{iq}(N)M_{qk}(nN) + m_{rk}(nN)(1 - p_{iq}(N)) \text{ und} \\ (7.6) \quad m_{ik}((n+1)N) &\leq m_{rk}(nN) + (M_{qk}(nN) - m_{rk}(nN))c. \end{aligned}$$

c.) Wir ziehen Gl. (7.5) von Gl. (7.6) ab und erhalten

$$m_{ik}((n+1)N) - M_{ik}((n+1)N) \leq (m_{rk}(nN) - M_{rk}(nN))(1 - c) - (m_{qk}(nN) - M_{qk}(nN))c.$$

Da unsere MC's transitiv sind und wir das Minimum der Masse mitnehmen, gilt

$$M_{ik} = M_{qk} = M_{rk} = M_k \text{ und } m_{ik} = m_{qk} = m_{rk} = m_k.$$

Die grossen M bezeichnen das Mass einer Ersetzungskette, und die kleinen m bezeichnen das Mass der Ersetzung, die das Mass einer Kette erzeugt. Deshalb ergibt sich

$$m_k((n+1)N) - M_k((n+1)N) \leq (m_k(nN) - M_k(nN))(1 - 2c). \quad \square$$

Die Übergangswahrscheinlichkeiten haben also nach Normierung den Grenzwert 1. Mit anderen Worten unser Kac-Moody Lie Inferenz- und Problemlösesystem ist korrekt und widerspruchsfrei. Der 2. Gödel'sche Unvollständigkeitssatz gilt in unserem nicht-monotonen, ergodischen Inferenzsystem nicht.

$Th^+(A_j)$ - Theorien sind nicht-monoton korrekt [1]. Sie sind entscheidbar, widerspruchsfrei und vollständig [1]. Der 1. Gödel'sche Unvollständigkeitssatz gilt für unser Inferenzsystem nicht. Damit ist Hilbert's Programm vollendet.

Da wir die Falschheit nicht definieren, sind Widerspruchsbeweise nicht zulässig. Wir fragen daher, von welchem Zahlentyp ist die Quadratwurzel aus 2?

Widerspruchsfreiheit ist nach unseren Ergebnissen äquivalent zur Entscheidbarkeit, das heisst die Prädikatenlogik ist nicht widerspruchsfrei.

Literatur

1. C. Kuck, Non-monotonic learning automata, Verlag Ferdinand Schöningh, Postfach 2540, D-4790 Paderborn (Fed. Rep. Germany), 1984

34

Contribution to the ICMI discussion on "The
influence of computers and informatics on
mathematics and its teaching".

(Strasbourg, 25-30 March, 1985)

by N. G. de Bruijn

1. Comments related to part 1 ("The effect on mathematics")

1.1. Automath

Computers influence mathematics in many ways. One of them 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.

1.1.1. 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 mathematically correct.

1.1.2. 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 might be simply tedious routine work, but also possibly the more serious problem solving (i.e., "theorem proving", a branch of artificial intelligence).

1.1.3. 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.

1.1.4. 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, just like mathematical theorems are applied.

1.1.5. Writing in Automath can be tedious. All details of

arguments have to be presented most meticulously. 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 subconsciously. Mathematicians have a vast "experience" in mathematical situations, and that experience may give a strong feeling for how all the little gaps can be filled. We do not know how much of the experience is consulted subconsciously "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 improving it; certainly it is a thankless 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.

1.1.6. 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 well over a million characters, all typed by hand. It is hard to guarantee correctness of such a text without the help of a modern computer.

1.1.7. 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 of 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 not any 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 too.

1.1.8. 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 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.

1.1.9. 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 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 guarantees 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 say, 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.

1.1.10. One thing people like in Automath, and other people

strongly dislike, is the way Automath treats proof 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 1.1.8.

1.1.11. 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 $1 := \text{succ}(0)$, $2 := \text{succ}(1)$, ..., $9 := \text{succ}(8)$, $\text{ten} := \text{succ}(9)$. After having introduced addition and multiplication we can define things like $\text{thirtyseven} := \text{sum}(\text{prod}(3, \text{ten}), 7)$, but the AUT 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 1.1.5).

1.1.12. 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.

1.1.13. 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.

1.1.14. 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.

1.1.15. 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 1.1.13 used the Landau translation (see 1.1.14) as a data bank, and that way they added to the bank.

1.2. 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 they 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.

1.2.1. 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 than 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 ...).

1.2.2. 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.

1.2.3. 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.

1.3. 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.

1.3.1. 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.

1.3.2. 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.

1.3.3. 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, thus clearly demonstrating that the foundations were unnecessary. 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.

1.3.4. 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.

1.3.5. A foundation 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, students, even mathematics and computer science majors, are very weak in this respect.

1.3.6. In many parts of the new math, in particular in algebraic areas, it is quite hard to draw the borderline between mathematics and metamathematics (cf. 1.2.3). 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.

1.3.7. 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.

1.3.8. 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, sets 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. 1.3.4). 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 exercises.

In a sequel of this course (again 18 hours teaching plus

exercises), 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 borderline 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.

1.3.9. A course like the one described in 1.3.8 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.

1.3.10. 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. 1.1.7). And lambda calculus might really help to make school mathematics easier.

1.3.11. Some of the material mentioned in 1.2 was taught at Eindhoven since about 1977 in a course called "Language and structure of mathematics", for those mathematics

43

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.

The effect of computers on the teaching of mathematics students

D.L.Salinger

School of Mathematics, Leeds University

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 possibility and effectiveness of algorithms to solve 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 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 necessary prerequisite to the more theoretical aspects of computing. I feel it does not greatly matter whether a main-frame or micro computer is used, nor do I think it matters much which high-level language is used. 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 at Leeds and elsewhere has shown that the initial stages of learning 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 take 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, recursive functions, unsolvability of the Halting Problem, computational complexity.

Later courses could contain more on these topics, including a form of Godel's incompleteness theorem, as well as automata theory, denotational and operational semantics, theory of correctness of programs, feasibility and intractability. (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. 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. A course which covers both analytic and numerical theory has been running for a number of years at Leeds.

4. Use of packages in statistics teaching

This has changed the nature of statistics teaching in our courses at Leeds, both to specialist mathematicians and to other students. We use one or two of a number of commercially available packages, principally MINITAB 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.

Reference

[1] E.J.Redfern, *Computers, Practicals and Packages in Teaching Statistics*, Department of Statistics Technical Report No. , University of Leeds.

DLS 19DEC84

45

=====
The Progress of Computers and Mathematical Education
=====

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 used 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.

Haruo Murakami

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

Masato Hata

The Graduate School of
Science and Technology

Kobe University

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.,

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

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

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

- 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 astonishing. 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 strategy. 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 hypotheses. 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 $(1+X)^n$, students use the computer algebra system and expand the expression for $n=0,1,2,3,\dots$. 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 addition, 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. Hulzen 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).

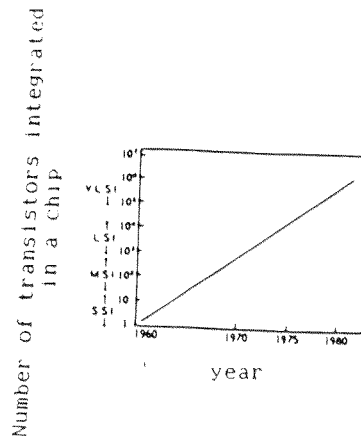


Fig. 2.1 Progress of the number of transistors integrated [5].

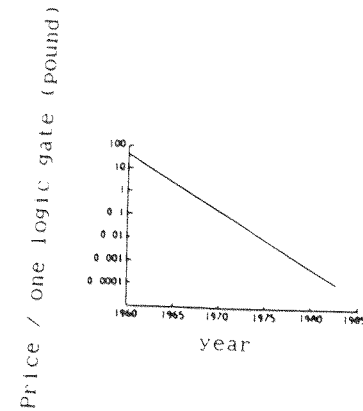


Fig. 2.2 Reduction of cost of LSIs [5].

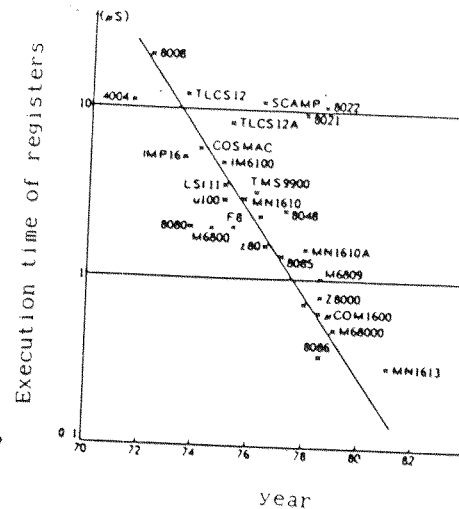


Fig. 2.3 Speed improvement of microprocessors [5].

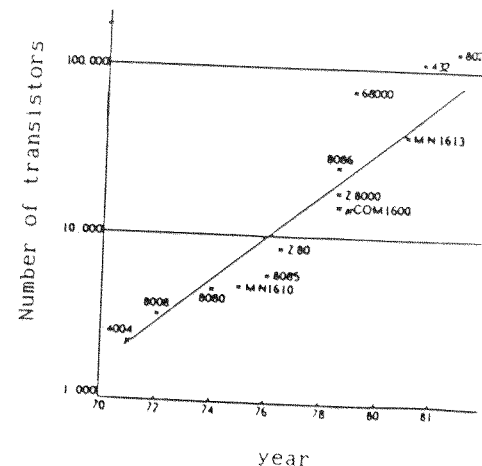


Fig. 2.4 Improvement of integration for microprocessors [5].

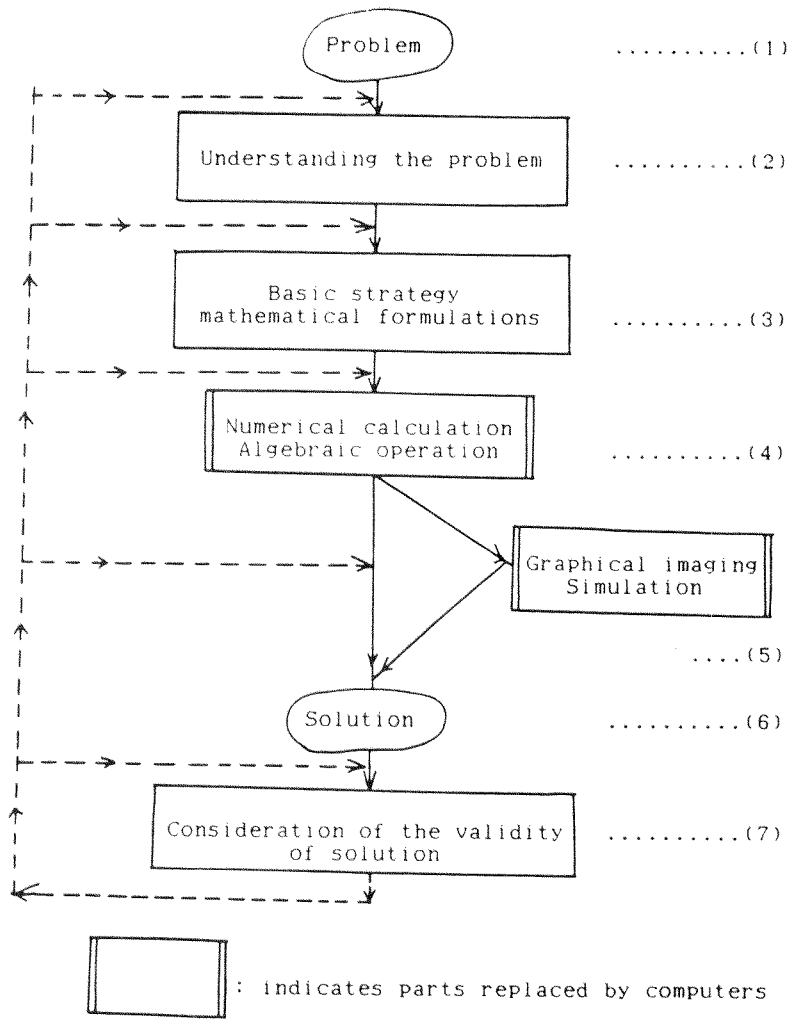


Fig. 5.1 A model of problem solving by the new teaching method incorporating computer algebra systems. (As indicated by dashed lines, it is possible to return from one point to any other.)

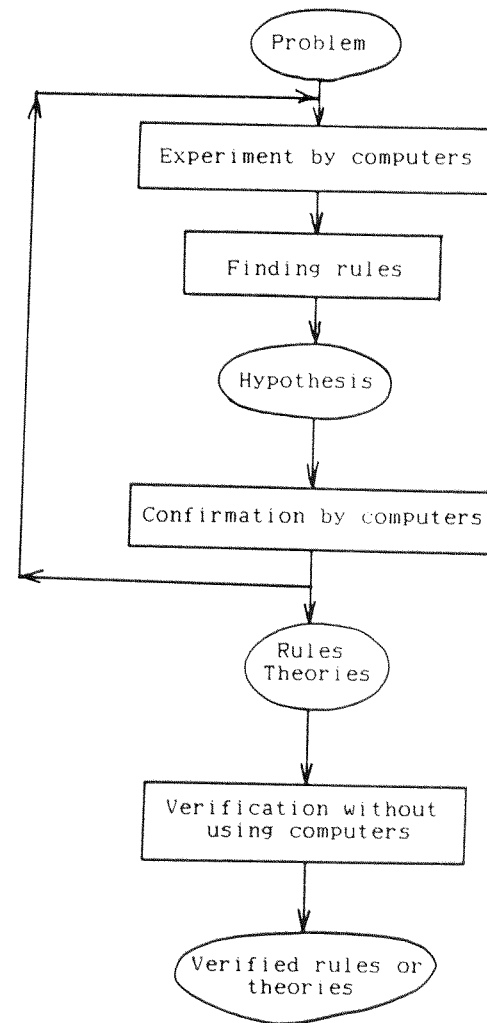


Fig. 5.2 A model of exploratory Mathematics.

Living with a New Mathematical Species

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

The history of mathematics can be viewed as a counterpoint between the finite and the infinite, between the discrete and the continuous. Greek mathematics was finite, concrete, and specific; modern mathematics 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 not a retreat. It is, rather, 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, "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 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 Françlais 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. There 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 first demonstrate 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

59

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.

Where's the Beef?

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 pure 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 needed 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 concensus 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 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

62 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. 61].

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 long 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 spread sheets, 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--recursion and procedures in Pascal, 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 computing joins mathematics as a basic ingredient in secondary and higher education, liberal education 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 mathematics curricula, 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 spread sheets that are marketed under such trade names as Lotus and VisiCalc. Originally designed for accounting, they can as well emulate cellular automata, 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 spread sheet--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 passé; 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 groundrules 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. viii]

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." 3

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 one 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 deus ex machina to rescue teaching from the morass of rules and templates that generations of texts and tests have produced.

Following the Rules

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 differentiating 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.

Let Us Teach Guessing

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.

Metaphors for Mathematics

Two metaphors from science are useful for understanding the relation between computing, 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 computing on mathematics and of mathematics on computing 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 wondrous 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.

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

This paper is a contribution to the session on Symbolic Mathematical Systems and their Effects on the Curriculum to be held at the Fifth International Conference on Mathematical Education, Adelaide, August 24-30, 1984.

(*) Permanent address.

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 heavily 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 organized 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 analyzes 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 summarized 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 easily 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,

This paper is a contribution to the session on Symbolic Mathematical Systems and their Effects on the Curriculum to be held at the Fifth International Conference on Mathematical Education, Adelaide, August 24-30, 1984.

(*) Permanent address.

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 mechanizable 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 realize 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 factorize 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 easily 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 realize 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 organization 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 emphasized 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).

G. BITTER

Abstract

First Year Results of the Microcomputer Assisted Mathematics

Remediation Project at Arizona State University

Introduction

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.

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 education, 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

7
7

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 increased 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.

Statement of the Problem

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 competency and effectively 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 convenient 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 item 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 coefficient).

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 completion. 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 laboratory 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 received 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 statistics 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 conceivably 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 satisfy 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 sensitive 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 development of a microcomputer achievement test of skills relevant 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., Pennema, 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 Considerations," Resource Manual for Counselors/Math Instructors, Middletown, CT, Washington School of Psychiatry, 1980.
- Lockhardt, Kathleen A. and others, "Computer-Managed Instruction 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 Systems 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.

DRAFT

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 approach is 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, "n ... 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 accommodate 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; linking several of these reconstructed structures together 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 is necessary to look a little more closely at the process of acquiring a concept. The last step in completing the construction of a cognitive structure is to apply it to various phenomena. At this point the structure is used implicitly and in order to proceed with the development of higher structures it is 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, he or she is using words and chalk to refer to mental images which the mathematician possesses. Unfortunately, the student very often does not possess such images and so he or she 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.

8
8

Unfortunately this activity does not help the students to encapsulate the function process and make it a cognitive entity that can be manipulated.

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. My idea for teaching a particular concept is to first make sure that the student has had and remembers 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 formed 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 piping 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 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 calculations) 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 7 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 does 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 is very likely the actual psychological decomposition.

SETL AND DISCRETE STRUCTURES

This project is, in several ways, the most extensive activity described in this paper. It is a full semester course designed to implement the ideas about using computer experiences that we have discussed. This is now a regular course at Clarkson University and has been given twice. 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 length), that the following SETL expression

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

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 programming 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.

$A := (\text{expression } \underline{\text{IN}} \ x: \ x \ \underline{\text{IN}} \ S \ \underline{\text{ST}} \ \text{boolean expression } \underline{\text{IN}} \ x;$
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 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 addition, 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

91

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

FORALL x IN S EXISTS y IN T ST FORALL z IN 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, not, 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 IN [1..K] | I IN [1..N]]

The students learn to translate this to

92
$$\left(\sum_{j=1}^k A_{ij} x_j \right)_{i=1}^N$$

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 length 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 score was 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 again 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 over 60%. 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 encouraging. 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 perceived 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 $H = F \circ G$ 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 single activity would be largely a waste of time for students in two of the three stages.

In our future work on this project we will interview the same students 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 verbal reference to computer analogies, there were no actual computer experiences 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 course 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 accommodate 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

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 hope others will 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 tapes 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. Sci. 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).

95

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).

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. Schrankenätze 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 transzender 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 Differenzgleichungen und Diskussion der linearen Differenzgleichung
- 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.

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 people.

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.

103

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 maintenance, 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 $\epsilon - \delta$ 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

105

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_n . 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.

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 réflexion 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. PUECH, 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 bases 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 McCulloch 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 "récurif" et du "calculable".

109

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 (random access machine, cf. Cook et Reckhow [1973]) contrôlée par l'intermédiaire d'un langage simple, type langage machine.

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".

110 S2. 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 . 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 procédures 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.

WAS ERGIBT SICH AUS DER GRUNDLAGENKRISE DER MATHEMATIK?

M. Otte

Im Kursbuch 1978 zum Thema "Lust an der Theorie" (Dezember 1984) beschreibt der Biochemiker Erwin Chargaff sehr erfahrungsreich und anschaulich viele wirkliche und manche scheinbaren Auswüchse und Gefahren moderner Naturwissenschaft und Technik. Als Kern des Übels identifiziert er die Methoden, ohne die Naturwissenschaft für ihn nicht vorstellbar ist, obwohl er aus Erfahrung weiß, "wie oft Methode ein Ersatz fürs Denken ist. Wie viele der Forscher, die ich an den komplizierten und ach so kostspieligen Apparaten sitzen sehe, stehen was sie tun und worauf die von ihnen verzeichneten Ergebnisse beruhen? Sie drücken auf Knöpfe und werden berühmt. Was geschehen ist, das METHODE die Menschen überwältigt und, im Falle des Computers, verdrängt hat" (S. 57).

Die Methode ist das Werkzeug der "Sucht, aus der Naturwissenschaft eine Erklärungswissenschaft zu machen" (S. 51). Chargaff unterscheidet nun zwischen Methoden, die "den direkten Weg zu einem Ziel bedeuten" und solchen, die die Umgehung von Hindernissen zum Ziel haben. Also die direkte Methode als der "Weg zu etwas", oder die indirekte Methode als der "Weg um etwas herum", und er meint, daß "unsere gegenwärtigen Wissenschaften immer mehr durch die Verdrängung direkter Methoden durch indirekte gekennzeichnet sind", sowie "daß die Bevorzugung indirekter Methoden wesentlich einer Wissenschaft abträglich sein kann", was er dann am Beispiel der Chemie zu illustrieren sucht. Der Kern seiner Argumentation ist die Überzeugung: "Wirklichkeit ist das Unmittelbare, das Unvermittelbare" (S. 52). Sein Anliegen ist dabei die Wirklichkeit als Einmaliges, Unwiederholbares. Die Schädlichkeit der Methode liegt eben darin, daß sie diese Unwiederholbarkeit zerstört (S. 54).

Aber ist es nicht gerade umgekehrt so, daß die Einmaligkeit und Unwiederholbarkeit zeigt, daß die menschliche Wirklichkeit vermittelte Wirklichkeit ist? Wirkt denn nicht gerade deshalb eine Abbildung "umso verfälschender je farbenechter" sie ist? (S. 52).

Nun hatte sich speziell in der Mathematik diese Situation zugespitzt. Die Mathematik ohne sichtbaren äußeren Gegenstand, die ganze

Weit ihren Methoden gegenübergestellt, versuchte sich des Sinns, der Einmaligkeit der subjektiven Wirklichkeit durch die Wahl der besonderen Methode zu versichern. "Moralische Argumentationen im Grundlagenstreit der Mathematiker" (Paul Lorenzen) sind in keiner Weise abwegig, aber berauben sie sich nicht selbst, wenn sie sich auf die subjektive Bedeutung zu wählender oder auszuschließender formaler

Methoden konzentrieren? Hermann Weyl hat versucht, sein eigenes Resümee des "Grundlagenstreits" wie folgt zu formulieren: "Nimmt man die Mathematik für sich allein, so beschränke man sich mit Brouwer auf die einsichtigen Wahrheiten ... In der Naturwissenschaft aber beruhren wir eine Sphäre, die der schauenden Evidenz sowieso undurchdringlich ist; ... und es ist darum, wenn die Mathematik durch die Physik in den Prozeß der theoretischen Weltkonstruktion mit hineingenommen wird, auch nicht mehr nötig, daß das Mathematische sich daraus als ein besonderer Bezirk des Anschaulich-Gewissen isolieren lasse: Auf dieser höheren Warte, von der aus die ganze Wissenschaft als eine Einheit erscheint, bin ich geneigt, Hilbert recht zu geben" (vgl. Ges. Werke, Band IV, S. 334).

Heißt das nicht, daß praktisch die gesamte heutige reine Mathematik eigentlich unmöglich ist? Hermann Weyl sagt explizit, daß nur das "Verhältnis zur Physik deutlich macht, was es mit dieser transzendenten, rein symbolischen Mathematik auf sich hat" (a.a.O., S. 332), und in der Zusammenfassung seiner Argumentation schreibt er schließlich: "... selbst in der reinen Mathematik oder Logik können wir die Gültigkeit einer Formel ihr nicht mittels eines deskriptiven Merkmals ansehen, sondern sie wird gewonnen nur durch praktisches Handeln, indem man nämlich von den Axiomen ausgehend in beliebig oftmaliger Wiederholung und Kombination die praktischen Regeln des Schließens anwendet. Man kann darum sprechen von einer ursprünglichen Dunkelheit der Vernunft: Wir haben die Wahrheit nicht, es genügt nicht, große Augen zu machen, sondern sie will durch Handeln gewonnen sein" (a.a.O.).

"Sehen" oder "Handeln" kann das die Alternative sein? Überhaupt ist der Brouwer'sche Standpunkt keine Alternative und das läßt sich gleich doppelt begründen.

Erstens können wir fragen, inwiefern ergibt der Ausschluss bestimmter Methoden und Mittel eine Alternative? Es ergibt sich eine Alternative höchstens aus den Bedingungen des Dialogs insofern ich den anderen überzeugen muß, daß ein bestimmtes Mittel oder eine bestimmte Methode leistet was sie soll. Diese Vorstellung von dem Mittel, das

etwas zu leisten hat, führt nun aber zu der Vorstellung vom Verstehen als einem Anwenden.

Man könnte auch anders vorgehen. Man könnte sich fragen, was will ich denn eigentlich bewirken durch mein Handeln (welches ich dem "Sehen" gegenüberstelle)? Eigentlich will ich nichts Gegenständliches bewirken, werden jedenfalls diejenigen sagen, die mit der Gleichsetzung von Verstehen und Anwenden nicht einverstanden sind. Sie werden fortfahren: "ich will das Handeln selbst aufbauen, entwickeln". Dazu findet sich bei Wittgenstein eine Anmerkung, die das erläutern konnte: "Der rechnet nicht, der, wenn ihm einmal das, einmal jenes herauskommt, und der, wenn er einen Fehler nicht finden kann, sich damit abfindet und sagt: das zeige eben, daß gewisse noch unbekannte Umstände das Ergebnis beeinflussen. Man könnte das so ausdrücken: Wem die Rechnung einen Kausalzusammenhang entdeckt, der rechnet nicht" (Bemerkungen über die Grundlagen der Mathematik, V - 40). Man könnte auch sagen, die reine Prozedur oder Methode hat keine "gegenständliche Wahrnehmung" und ist vollständig auf sich selbst bezogen. Das Gegenständliche ist einem solchen operativen Prozeß höchstens Anlaß, seine Maßnahmen zu wechseln, seine Richtung zu verändern.

Gleichzeitig wird aus dem Zitat auch deutlich, warum das Rechnen, das "Spiel mit formalen Regeln" für die philosophische Selbstreflexion der Mathematik so bedeutsam erscheint. Heute müssen wir allerdings konstatieren, daß der Computer durchaus das durchschimmernde Ideal erfüllt. Warum akzeptiert man diese prinzipielle Bedeutung des Computers nicht? Man antwortet - wie kürzlich auf einer Diskussion in Oberwolfach - der Computer könne die theoretischen Begriffe nicht verstehen. Das Problem des Verstehens sieht dann wie folgt aus. "Man könnte fragen: was könnte ein Kind von zehn Jahren am Beweis des Dedekind'schen Satzes nicht verstehen? - Ist denn dieser Beweis nicht viel einfacher, als alle die Rechnungen, die das Kind beherrschen muß? - Und wenn nun jemand sagte: den tieferen Inhalt des Satzes kann es nicht verstehen - dann frage ich: wie kommt dieser Satz zu einem tiefen Inhalt?" (Wittgenstein a.a.O., IV - 31). Die Antwort würde lauten: durch die Anwendung. Also aufs neue: Verstehen ist Anwenden.

Verstehen kann von den Methoden und Mitteln nur insofern abhängen als dadurch bestimmte Anwendungen ermöglicht werden oder nicht. Verstehen verlangt also als Voraussetzung die Entwicklung der Mittel - und die Entwicklung des Computers hat uns in dieser Hinsicht einen großen Schritt vorangebracht - und die Entwicklung des Subjekts der

Anwendung bzw. des Verstehens.

Was heißt es also, den Anwendungen einen derartigen prinzipiellen erkenntnistheoretischen Status einzuräumen? Es bedeutet, den Gegenstand der Theorie als ihre intendierten Anwendungen aufzufassen. Das wiederum impliziert, daß Theorien nur einen indirekten Gegenstandsbezug haben, daß sie prinzipiell von den Realitäten, auf die sie sich beziehen, unterschieden werden müssen, und es bedeutet weiter, daß ihr Gegenstand von der Entwicklung des Erkenntnissubjekts abhängt. Es muß also die Kommunikation und Kooperation der individuellen Erkenntnissubjekte entwickelt werden, weil nur so die Erfahrung einer Theorie (eines Wissens) und seiner Anwendung oder Entwicklung gleichzeitig realisiert werden können. Die Anwendung ist also auch das Kriterium der Methode.

Am Ende treffe ich mich allerdings doch wieder mit Chargaff, wenn er meint, "daß es dem reinen Forscher nur wenig hilft, wenn er darauf beharrt, daß nicht er, sondern der Ingenieur schuld sei an der Verpestung der Welt. Solche Unterscheidungen sind uninteressant für den Nichtfachmann; denn während Dr. Faust nach oben weist, wo die höheren Erkenntnisse wohnen, zeigt der ungelehrte Finger nach unten auf die Giftdeponie, wo mephistophelisch und mephitisch Synonyme geworden sind. Seltsamerweise verharret die reine Forschung viel zu lange in dem Glauben, daß sie mit Peter Schlemihl die Unfähigkeit gemeinsam habe, einen Schatten zu werfen".

PERSONAL COMPUTERS IN TEACHING BASIC MATHEMATICAL COURSES

P. Boieri, Politecnico di Torino, Italy
L. Chiantini, Politecnico di Torino, Italy
G. Geymonat, Politecnico di Torino, Italy
P. Moroni, Politecnico di Torino, Italy
A. R. Scarafioti, Politecnico di Torino, Italy

SUMMARY

This is a report about an experiment in progress at Politecnico of Torino, aiming at an integration of use of computers in basic mathematical courses.

With the purpose of teaching mathematics through the computer, we introduce the students to a programming language and to fundamentals of computer science.

Then computers are used in a wide range of situations: in conjecturing values of limits and orders of infinitesimals, in studying discrete dynamical systems and local behaviour of functions, in computing sums of series.

Graphic facilities of personal computers are very useful in giving a picture of some abstract situations; for instance, how power or Fourier series approximate a given function or how a linear transformation changes a neighbourhood of the origin in the plane.

This possibility of creating a large number of examples helps the student in understanding the theory, while he becomes more familiar with the capabilities and deficiencies of the machine.

At the end of the mathematical curriculum, personal computers are used also for tutorial programs that introduce the student to more advanced topics usually not treated in class.

1. INTRODUCTION

This note comes out from the experience we are having, since a few years ago, in teaching basic mathematics to engineering students at Politecnico of Torino. In our work we believe that it is important to avoid two opposite mistakes: on one hand to be too abstract and to aim only at an elegant and logically consistent presentation of the theory; on the other to consider only the use of mathematics in applied sciences, making just an exposition of a list of formulas and recipes.

One of the possible ways for saving precision and avoiding excessive abstractedness is to use pocket calculators and personal computers as a teaching aid in basic mathematical courses. This allows us to make some important abstract ideas more concrete; moreover the students learn how to use an essential tool in applied sciences; they are introduced into the problems of numerical analysis and they are stimulated to go deep into it, taking a specialized course later in their curriculum.

A correct use of computers requires a new curriculum, in which some fundamental concepts (i.e. algorithm, flow-chart, floating point computations) and a programming language are introduced; however it also requires a more radical change in teaching basic mathematics.

The constructive and computational aspects of proofs must be emphasized, whenever possible, in order to have methods suitable for computer programming. Let us consider, for instance, the proof of existence of zeros for continuous functions; the method of subsequent subdivisions of the interval is well suited for computers, while the proof that uses connectedness and properties of continuous functions is not.

Graphic facilities of personal computers are very useful to help the understanding of some abstract concepts, usually considered rather difficult (i.e. the pointwise and uniform con-

vergence of series, the transformations $R^2 \rightarrow R^2$).

The early exposition of students to computers can be helpful for a "critical" use of the machine, i.e. for emphasizing the limits of this tool and the necessity of a good understanding and position of the problem, before beginning the program; the computer, in this way, does not become a "myth" as unfortunately happens today too often, because of the fast growing use of computers in research and everyday life.

From these premises, it is clear that our experience cannot be described as a "course about computers" but as "teaching mathematics through computers", i.e. an integration of machines in traditional mathematical curriculum; this integration is realized in different ways in the three semesters in which basic mathematics is taught; this is described below in more detail.

2. THE USE OF COMPUTERS IN THE COURSES

In the first semester the basic ideas of calculus (limits, derivatives, integrals) are introduced; the fundamental problem in this course is to give the students a correct method of approaching mathematics.

This applies also to the use of computers; first BASIC language and some fundamental notions such as algorithm, flow diagram, floating point computations are introduced; then problems are solved using a method divided into three steps: conjecture, proof, refutation (see /1/).

With the help of a computer we make a numerical conjecture about the solution of a problem; then we prove the result in a rigorous way; eventually we go back to the numerical result we got. In this way we can emphasize that numerical conjectures may be wrong and that a global analysis of all errors intervening in the numerical process is necessary.

Let us consider a couple of examples concerning computation of

limits and orders of infinitesimals.

Example 1.

Compute $\lim_{x \rightarrow 0} f(x)$ for $x \rightarrow 0$, where

$$f(x) = (x - \sin^{-1} x) / \sin^3 x.$$

With a pocket calculator (TI 58) we obtain this table, with a four digit approximation:

x	f(x)
0.1	-1.6826 10^{-1}
0.05	-1.6706 10^{-1}
0.02	-1.6673 10^{-1}
0.01	-1.6668 10^{-1}
0.005	-1.6667 10^{-1}

The numerical conjecture is that the value of the limit is -0.16666...; this result is correct and can be proved using Taylor's expansions.

If we continue our numerical experiment for smaller values of x, we have these outputs:

x	f(x)
0.001	-1.6660 10^{-1}
0.0001	0

This table shows how we can be led to wrong conclusions; in this way we are warned against the occurrence of rounding errors.

Example 2.

Compute the order of infinitesimal of

$$f(x) = \sin x - x \cos(x/\sqrt{3}) \quad \text{for } x \rightarrow 0$$

in comparison with the order of $g(x) = x$.

The numerical results, obtained with an HP 34C, are, with a four digit approximation:

116

x	$f(x)$
$5 \cdot 10^{-1}$	$1.1460 \cdot 10^{-4}$
$1 \cdot 10^{-1}$	$3.7030 \cdot 10^{-8}$
$5 \cdot 10^{-2}$	$1.1500 \cdot 10^{-9}$
$1 \cdot 10^{-2}$	0.0000
$5 \cdot 10^{-3}$	0.0000
$1 \cdot 10^{-3}$	0.0000

In this case no reasonable conjecture can be made using the numerical results; using calculus we have that

$$f(x) = (1/270)x^5 + O(x^7) = 3.7037 \cdot 10^{-3}x^5 + O(x^7).$$

The theoretic result can help to understand the failure of numerical experiment; this improves the capability of students of reading critically computer outputs.

Students work at personal computers too, trying to solve some problems suggested to them; here are two examples aiming at a better understanding of the concept of real number.

Problem 1. Given a number $a = p/q$ with $0 < a < 1$, write a program that gives the first n digits of its representation with respect to an assigned base b .

Problem 2. Write a routine that gives the first n digits of the square root of a positive real number, using a base $b \geq 2$.

Also local behaviour of functions, Taylor's expansions and discrete dynamical systems have been studied with personal computers (see /2/).

The second semester is concerned with the following subjects:

- 1) Linear algebra
- 2) Geometry of the plane and the space
- 3) Differential geometry of curves and surfaces.

Linear algebra is an interesting field of application of computer programming, mainly because of its combinatorial aspects;

however our first aim (to teach mathematics through the computer) can be easily missed if we point our attention only to the research of combinatorial algorithms; instead we try to develop with the students some programs that take care of the underlying geometrical interpretation of linear algebra. So, for instance, together with a program that computes the determinant and the eigenvalues of a 2×2 matrix, we always ask for a routine that reproduces the deformation of a neighbourhood of the origin represented by a matrix.

The general philosophy is that computers can provide a large number of examples of how a given algebraic procedure applies to geometry. This should improve the intuition of the students, their skill to recognize at the first glance for a given problem the geometric situation that is likely to be expected.

In the second and third part of the semester the above concepts are emphasized; computers give here a tool for refining the knowledge of the basic methods of geometry of plane and space. They allow to see which change in the geometry of a given problem is introduced by a slight variation of the data.

Another interesting aspect of the application of computers for teaching geometry is that a general algorithm for the solution of a problem often fails when applied to some particular (and degenerate) case; indeed such degenerate case must be treated by a "ad hoc" routine. This often implies a better knowledge of the geometry of the problem; in fact learning geometry through the degeneration of standard problems is a fascinating way to understand the abstract theory.

The main topics treated in the third semester are:

- 1) Multiple integration
- 2) Numerical and function series
- 3) Ordinary differential equations.

71

Many numerical experiments can be made in order to show how the definition of integral (in one or several variables) works; the student can write a routine where a given function is approximated, by above and below, by step functions and the convergence of the integrals of these step functions is studied.

In this process, it is natural to introduce a better approximation of the function: we obtain a class of numerical integration formulas that can be compared each other.

On the contrary, the evaluation of the sum of numerical series is useful to get the student aware of the effects of different kinds of numerical errors; moreover, the impossibility of discriminating divergent from slowly convergent series points out the necessity of knowing the theoretical results.

Several examples can be provided by personal computers to clarify the theory of function series; this is a good background for introducing the different concepts of convergence (pointwise, uniform and in L^p norms) and comparing them each other.

But the field where the use of computers changes more radically traditional educational patterns is that of differential equations.

There are very few equations relevant in applied sciences for which the solution can be given in an analytic form; the knowledge of simple numerical methods is sufficient for the student to see the qualitative behaviour of the solution, helping the physical and mechanical understanding of the phenomenon.

The study in the phase plane of planar autonomous systems with constant coefficients is emphasized (see /3/); thanks to the display it is possible to visualize the concept of stability of an equilibrium point for linear and non linear systems.

This is an example of the possible use of computers in the realisation of tutorial programs on more advanced topics that are useful in the further curriculum of an engineering student and

are habitually neglected in basic courses.

REFERENCES

- /1/ G. Geymonat *Congettute e dimostrazioni: possibili usi didattici dei calcolatori tascabili nell'insegnamento della matematica* La Fisica nella Scuola, XV,4 (1981), 129-136.
- /2/ P. Boieri *An experiment of use of computers in basic mathematical courses* To appear in the Proceedings of 2nd SEFI European Seminar on Mathematics in Engineering Education. Developments and Innovation." Kassel, 1984.
- /3/ A. Bacciotti, P. Boieri, P. Moroni *An introduction to linear differential systems with the aid of personal computers* To appear.

Rolf Biehler
Institut für Didaktik der Mathematik
Universität Bielefeld
Postfach 8640
48000 BIELEFELD

Bielefeld, September 1984

Comments (preliminary version) on the ICMI-Paper:
The Influence of Computers and Informatics on Mathematics and Its Teaching

The following comments on the ICMI-paper are primarily made on the background of my experience with my main field of interest, i.e. statistics/data analysis and its teaching. The ideas presented owe much to the discussions I had with some colleagues of mine at the IDM in Bielefeld.

I agree that it is necessary to study the effect computers have on mathematics although, as is well stated in the paper, the developments in mathematics can only be one important reference point for designing new curricula, for it is equally important to consider social needs. From reading chapter 1 (effects on mathematics) I got the impression that there might have been a certain bias towards pure mathematics in selecting the examples. I think it would be important to include applied mathematics more systematically in two respects at least: applied mathematics as a field of knowledge and research, which has undergone a profound change due to the availability of computers, and the processes of applying mathematics, which have changed dramatically because of the availability of computer software. Computer software has changed the cognitive dimension of applying mathematics (how problems are solved) as well as the social-communicative dimension, i.e. the division of labour between mathematicians and subject matter experts and the tools (e.g. computer graphics) of communication about problems and their solutions. I think, a detailed study of the development in particular areas of (applied) mathematics is highly desirable and could serve two ends: to identify

trends working in mathematics in an inductive way and to identify the particular conceptual and "philosophical" change in particular fields in order to give advice with which new emphasis these particular fields should be taught.

In what follows, I shall comment on some selected topics which are partly mentioned in the paper and partly seem to me worth considering to mention in more detail in a revised version of the paper, namely:

- Algorithmics
- The emergence of new mathematical concepts
- Visualizations
- Experimentation and the status of proof
- The process of applying mathematics
- Distinction between secondary and tertiary level

Algorithmics

On page 5, the influence of fundamental concepts of informatics on mathematics is stressed; algorithmics is considered to be found in the forefront. But the impact of this concept on mathematics has been at least contradictory: on the one hand, it gave rise to new theoretical research on the design and analysis of algorithms, e.g., from numerical analysis, complexity theory to stochastic analysis of algorithms. The design of effective algorithms, which for a long time had been considered to be only second order mathematics, is now appreciated as a challenging field in its own right. On the other hand, for most people concerned with mathematics, the availability of computer software liberates them from having to deal with algorithmic details. They are free to concentrate on other aspects of their problems. In a sense, mathematics has become less algorithmic due to computers. This is well illustrated in the field of statistics where classical statistics was designed mainly for routine (algorithmic) analyses when the model was given, whereas the modern "interactive data analysis" relies heavily on human intervention

in many stages of the process. These changes should have consequences for future curricula, but it is far from being clear which relation between intelligent use of prefabricated software and design of algorithms by students is desirable. A subproblem is how to develop criteria for designing the user interface for mathematical software to be used in educational contexts.

The emergence of new mathematical concepts

On page 5, it is said, that one "can expect" that the new methods of computation will lead to the emergence of new mathematical concepts. This has already occurred to a large extent. The development of statistics during the last, say, 25 years is a very rich source of examples for this thesis, take for instance topics concerned with robustness and all the techniques and concepts for the analysis of multivariate data (see e.g. Efron 1979 for some other examples). The conscious realization of the revolutionary impact of computers on statistics seem to be rather old compared to other fields of mathematics (see e.g. Yates 1966, Tukey 1968, Tukey 1965), and I think much can be learned from the discussions on the relation of statistics and computer science and the role of computers in statistics, which can be found, for instance, in the Proceedings of the Annual Symposia on Computer Science and Statistics (e.g. Heiner et al. 1983). I think that the change cannot be adequately described merely by the emergence of new mathematical concepts. The modern computing systems gave rise to completely new strategies or "philosophies" for the handling of data. This trend is exemplified by the development of the so-called Exploratory Data Analysis (see e.g. Tukey 1977, Biehler 1982) in the United States and the movement for "l'analyse des données" in France (see e.g. Benzécri 1980). These changes are particularly relevant for the curricula on probability and statistics (see Biehler 1984). The developments in statistics also illustrate the presumable general phenomenon that the computer has lead to a redefinition of what is "simple" and not only to an extension of relevant concepts and methods (see also Efron 1979, 439);

of course, this is highly relevant for the sequence of the topics to be taught in schools.

I think the phenomenon that the computer leads to the emergence of new mathematical concepts is relevant in yet another aspect. It would certainly be insufficient to merely imitate the scientific use of computers in the classroom. Several suggestions, notable Råde (1982) illustrate how computers can be used as a means for the development of new concepts (relative to the students knowledge) especially by using simulation in stochastics.

Visualizations

The new possibilities of visualizations are mentioned in the text (p. 8), but I think they deserve a far deeper treatment. In my opinion, it is only slightly exaggerated that the visualizing capacities of computers will/can/should be one of the main contributions to mathematics and its teaching. Let me comment on that from two points of view.

Statistics has seen a renaissance of graphical methods during the last, say, 15 years (see e.g. Beniger/Robyn 1978, Wainer/Thissen 1981, Biehler 1984a), which would have been nearly impossible without the modern computing systems. Graphical (or semigraphical) displays serve two main ends: communication and exploration. The use of graphics to communicate information effectively to a broad public has a longstanding tradition in statistics, but the increasing amount of information available in science and society together with the new opportunities has led to many efforts to invent new representations and to use them more effectively. In a similar vein, this trend applies to other mathematical (or partly mathematical) sciences, and modern curricula should reflect this trend by offering an adequate "graphico-mathematical education". What I consider even more important is the use of graphs for exploration and analysis. Graphical display has become a research tool in its own right, whereas using it was mainly seen as an unintellectual business

for quite a long time, at best useful for purposes of communication. The new appreciation for graphics shows up not only in statistics, but also in domains such as dynamical systems, which is mentioned in the text. A careful assessment and analysis of the use of computer graphics in mathematics and its applications seems to be of considerable importance. As to statistics or data analysis, the main purpose of graphics is to reveal structure in data sets, to aid the process of discovery, related to data, namely in situations where trustworthy analytical (probability) models are not available. This might be slightly different compared to other fields, where graphics are also employed to study features of models that are analytically not tractable and where graphics allow to handle the vast amount of numerical output of simulation studies effectively, in order to discover new (qualitative) features of the system under study.

Visualizations are certainly important from another point of view, as well. They have always played a role in the day-to-day practice of mathematics teaching as means of facilitating the learning of abstract mathematics. The new possibilities of computers should also be explored in what they can contribute to this important aim. But this aim has to be redefined if one takes into account the changed attitude towards computer graphics as a genuine research tool in mathematics (see Biehler 1984b).

Experimentation and the status of proof

The paper well states that computers have greatly increased the possibilities of experimentation in mathematics and affected the status of proof (p. 7ff). Euler is cited 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.

For one thing, computers have increased the amount of "mathematics without proof". Let me take one example from statistics: the case of

robust estimators. The famous Princeton Robustness Study (Andrews et al. 1972) studied sets of estimators under a system of different modelling assumptions by means of Monte Carlo simulation. Although the results also led to new mathematical concepts and theorems, the results in itself 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 analysing data. I guess that this is quite similar in other areas where simulation studies are conducted. The results are important in themselves and can, to a certain extent, be validated by their success in applications outside mathematics. Even if mathematical results on the performance of statistical procedures can be proved, such proofs should be interpreted as "test" of procedures under idealized conditions, which are never fully met in practice. It is therefore necessary to study the performance also under real conditions, i.e., to evaluate procedures on real data (see Tukey 1962). Another case in point may be numerical analysis, where algorithms are applied often without complete proofs, and where their performance on selected "exemplars" is equally important for the user as are some mathematically proved results.

For the natural sciences, N.F. Lane, the president of the consulting committee of the National Science Foundation for "Advanced Scientific Computing" has stated a fundamental methodological change comparable 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 computer simulation and related methods of computer science (see Physics Today, May 1984, 61). This development seems to be closely related to the rise of experimental mathematics and has led to changed relationships between mathematics and other sciences.

Let me summarize. The interplay between experimentation and proof in mathematics will certainly continue, but the experimental results have become more important and cannot be expected to be completely substitutable by mathematical proofs. Although this might have always been a

feature of applied (engineering-like) mathematics, the computer has enhanced this aspect. In my opinion, it will be important to discuss the pedagogical relevance of such a broader understanding of "experimental mathematics" in depth.

I should like to add that Papert's (1980) pedagogical and epistemological concept of "microworld" which is increasingly discussed, at least in circles of math educators, is highly relevant for this issue, because it shows an interesting way how mathematical concepts and theories can be learned by experimental interaction with a well-designed microworld. The relationship of this approach to the developments in "experimental mathematics" in a "scientific" context should be further explored.

The process of applying mathematics

There has been quite a long-standing discussion on how to teach applying mathematics or mathematical modelling during the last 15 years, and that related to the secondary level and to the tertiary level as well. I think it would be reasonable to take up this point more systematically than the paper does, i.e., how should/could the teaching of applying mathematics be changed because the use of computers while applying mathematics is becoming the rule rather than the exception in most application domains of mathematics. Certainly, a shift in the competence required of an intelligent applier of mathematics can be observed. On the one hand, less knowledge seems to be required because the "mathematical intelligence" of existing software can be exploited. On the other hand, new qualifications are necessary for the intelligent use of software, but these qualifications are still to be defined more precisely, and one can expect that these qualifications cannot be reduced to mathematical qualifications in a traditional sense.

The use of computers while applying mathematics has not only led to new educational demands but also to new educational possibilities. It is not unplausible that the use of computers allows an education in applying

mathematics which has been requested programmatically for several years but has hardly been put into practice because the complexities of more realistic problems were not feasible for the average student. The computer provides leisure to concentrate on the more difficult and important aspects of realistic problems. Joiner (1982) presents some interesting suggestions for teaching the processes of applying statistics, which I do not intend to reproduce here. But I think much more experience has to be gathered before it can be judged whether such an approach can be effectively implemented in curricula.

Besides, the attempts to teach modelling or problem solving often explicitly include teaching heuristics, modelling strategies or principles of scientific research while hoping that this will provide a substitute for the students' own problem solving experiences. I have the impression that these approaches have had only a rather limited success, and I agree with Kelman et al. (1983, pp. 65) that the computer might provide a "problem solving environment" which allows the students to gather their own cognitive experiences more effectively and systematically so that the "scientific method-substitute" is no longer necessary.

Distinction between secondary and tertiary level

Last not least, I find it very important to make a clearer distinction between the secondary and tertiary level of education than the paper does. Secondary education has to provide general education, and 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 mathematization go hand in hand with "computerization". The curricula on the secondary level certainly should also reflect the use of computers in other domains of society such as business and industry.

Therefore, it seems to be necessary to widen the background of the ICMI-paper not only by including applied mathematics, as I have tried to do, but also by considering the use of computers and mathematics in other sciences and non-scientific contexts more systematically.

Besides, the motivational and cognitive prerequisites of students at the secondary level in general are different from those at the tertiary level.

In consequence, similarities and differences between teaching mathematics in the computer age at secondary and tertiary level should be elaborated.

References

- 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
- Beniger, J.R./Robyn, D.L.: 1978. Quantitative graphics in statistics: a brief history. In: Amer. Statist. 32, 1-11
- Benzècri, 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 Vol. 24
- 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 Vol. 30, 116-154, or as IDM Occasional Paper No. 48
- Biehler, R.: 1984a. Die Renaissance graphischer Methoden in der angewandten Statistik. To be published in the Proceedings of the 4. Workshop "Visualisierung in der Mathematik", Klagenfurt, July
- Biehler, R.: 1984b. Graphische Darstellungen. Bielefeld: IDM Occasional Paper No. 50
- Efron, B.: 1979. Computers and the theory of statistics: thinking the unthinkable. In: SIAM Review 21, 460-480
- Heiner, K.W./Sacher, R.S./Wilkinson, J.W. (eds.): 1983. Computer Science and Statistics: Proceedings of the 14th Symposium on the Interface. New York: Springer
- Joiner, B.L.: 1982. The case of using computers in teaching statistics. In: Proceedings of the First International Conference on Teaching Statistics, ed. by Grey, D.R. et al. Sheffield: University of Sheffield Printing Unit 1983, Vol. 1, 307-312
- Kelman, P. et al.: 1983. Computers in Teaching Mathematics. Reading (Mass.): Addison-Wesley
- Papert, S.: 1980. Mindstorms, Children, Computer, and Powerful Ideas. New York: Basic Books Inc.
- 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 Grey, D.R. et al. Sheffield: University of Sheffield Printing Unit, Vol. 1, 19-33
- 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.: 1968. Is statistics a computing science? In: Watts, D.G. (ed.): The Future of Statistics. New York: Academic Press, 19-28
- Tukey, J.W.: 1977. Exploratory Data Analysis. Reading (Mass.): Addison-Wesley
- Wainer, H./Thissen, D.: 1981. Graphical data analysis. In: Annual Review of Psychology 32, 191-241
- Yates, F.: 1966. Computers, the second revolution in statistics. In: Biometrics 22, 233-251

ABSTRACT

John NIMAN

The Influence of Microcomputers on the Elementary School Mathematics Curriculum.

Analysis of how programming with LOGO is changing the teaching of two-dimensional geometry. CAI role in emphasizing problem solving. Introduction of new vocabulary and concepts e.g. iteration and recursion. Changing students' and teachers' attitudes towards mathematics. Negative influence of microcomputers - other areas are being neglected, e.g. 3-dimensional geometry. overdependency and unrealistic expectations.

COMPUTER ASSISTED INSTRUCTION IN UNDERGRADUATE MATHEMATICS

The California State University system has funded a project to develop software that will assist in the teaching of undergraduate and remedial mathematics. I would like to discuss and demonstrate, if possible, the software that has been developed by this project in mathematics.

In the area of remediation, "Recalling Algebra" by J.W. Kinch is designed to help a student who has learned algebra at one time but needs a refresher course before entering a college mathematics course. It covers many of the topics on the Entry Level Mathematics examination required of all students who wish to enter a mathematics class at a California State University. There are also some topics which perhaps can be best illustrated by use of a computer. Professor Dan Rinne and I have written a computer aided instruction module which shows the relationship between the graphs of $f(x+b)+c$ and $f(x)$. In one particular segment a student selects a basic function, say $f(x) = x$. Then, by using various keys, the graph can be translated to the left, right, or reflected about the axis. Each time the graph is moved, the function is changed accordingly. The goal of this module is to dynamically illustrate the effect translation/reflection has on a function and enable the student to graph a complex appearing function by recognizing it as a translation/reflection of a familiar function.

BIBLIOGRAPHY

Marchisotto, Elena Ann; Prime Factorization and Least Common Multiples, The California State University, to appear.

Murphy, Mark; Perimeters, Area's and Circumferences, The California State University, to appear.

Kinch, John W.; Recalling Algebra, The California State University, 1984.

_____ ; Recalling Math, The California State University, to appear.

Okon, J. S. and Rinne, Dan; Graphing Techniques, The California State University, to appear.

Sundar, Viji; Geometric Measurement Skills, The California State University, to appear.

All of the above courseware is distributed through McGraw-Hill Book Company.

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.

130

- 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,
Conk.*

COMMUNICATION

Par Mme le Prof. Dr Josephine GUIDY WANDJA
Université Nationale
Département de Mathématiques
ABIDJAN (CÔTE 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 contexte actuel, on englobe dans l'informatique les activités de conception, de mise en place, d'évaluation, d'application et de mise à jour des systèmes de traitement, de stockage et de communication des données, en ce qui concerne tant les matériels que les logiciels, les aspects organisationnels et humains."

Il apparaît 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 méthodes 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 industrielles, la télécommunication, l'enseignement.

Pendant longtemps en Afrique, l'idée de l'informatique est restée liée à la notion de comptabilité, de gestion des entreprises; cependant l'avènement des microprocesseurs, en entraînant une chute des coûts, a permis de généraliser l'utilisation de l'informatique. Mais son introduction dans le système éducatif 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 développement en Informatique

De façon générale, le tiers-monde accuse un retard important dans le domaine de l'informatique. Le tiers-monde n'occupe qu'environ 5% du marché mondial; un déséquilibre 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 Etats africains surtout francophones ont dès 1960 entamé la mécanisation des activités de certains départements ministériels. Des centres informatiques furent créés et en 1971 les chefs d'Etat réunis à Njamena (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 créés par certains Etats tels que l'ISI (Institut Supérieur d'Informatique) en Côte d'Ivoire, l'IUT de Dakar (Sénégal), créé 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 croissant dans les pays en voie de développement. En effet, l'utilisation des moyens et des méthodes de l'informatique dans les pays développés a fait de

L'informatique un élément important de prise de décision au niveau les
plus divers. De plus en plus maîtrisée par ces pays, l'informatique
apparaît comme un instrument indispensable du développement 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'activités. 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
jusqu'à 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
développement doit être accompagnée de la formation du personnel apte
à utiliser .L'expansion 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 Secrétariat Général de
l'Organisation des Nations Unies : " L'acquisition d'un ordinateur
neut fort bien au départ créer au moins autant de problèmes qu'elle
en résoud. Entre les mains d'utilisateurs compétents, les ordinateurs peuvent
être à la fois utiles et efficaces. Mais c'est à des hommes qu'il re-
vient de fixer leur rôle et d'exploiter les possibilités qu'ils offrent
Leur efficacité dépend donc de la compétence et des connaissances de ceux
qui planifient, approuvent, mettent en oeuvre, dirigent, surveillent et
évaluent les activités pour lesquelles on utilise l'ordinateur."

Il apparaît donc que la formation constitue une priorité dans tous les
pays. L'enseignement a donc un rôle à jouer pour remédier au manque de
personnel spécialisé auquel se heurtent les pays mêmes développés.

L'Informatique et l'enseignement

L'introduction de l'informatique dans les lycées et Universités suppo-
se l'élaboration d'un vaste programme pédagogique, et la formation de
spécialistes pour l'éducation dans les lycées et Universités.

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
nécessaire à l'introduction de l'informatique à l'école car elle est
la garantie la plus sûre de l'introduction de nouvelles méthodes
d'enseignement et du développement des didacticiens.

Dans l'Ouest-africain, l'introduction systématique de l'informatique
à l'école n'est pas encore envisagée, à part quelques rares cas (3 collè-
ges catholiques au Sénégal, le Lycée français d'Abidjan, l'expérience
Logo à Dakar). Aucun établissement primaire ou secondaire ne possède de
matériel informatique. Cependant, beaucoup d'établissements de l'enseigne-
ment supérieur possède du matériel informatique et presque tous dispensent
des cours d'initiation en informatique.

Au Sénégal, l'expérience du Laboratoire Informatique-éducation consiste
à initier de jeunes enfants à l'ordinateur et au langage informatique
afin de permettre les adaptations socio-culturelles nécessaires à l'intro-
duction de cette nouvelle technologie dans l'enseignement. A Dakar
le centre mondial informatique et ressources humaines a démarré un projet
en Mars 1982 au centre de Recherche pédagogique de l'École Normale
Supérieure (ENS). Une équipe pluridisciplinaire composée d'1 mathémati-
cien, 1 informaticien, 1 sociologue, 2 instituteurs, 1 psychologue envoyés
en formation au "Logo computer center de New-York" dirige le projet.

Le langage Logo créé 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'exécute et
vérifie la logique et l'exactitude. Ce langage permet de passer de
l'intuition à l'abstraction; beaucoup de notions de la géométrie euclidienne
ne sont perçues intuitivement à l'aide de sa "géométrie de tortue". Ce
langage permet de saisir intuitivement la notion mathématique de l'infini.
L'expérience porte sur un échantillon 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 géométrie et les mathématiques modernes à l'école primaire. Il est envisagé actuellement l'installation de 10 micro ordinateurs par école expérimentale et la formation de futurs maître-instituteurs a été 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 nécessaire de créer des structures permanentes de recherche et de formation des enseignants.

Hartwig Meissner

Westfälische Wilhelms-Universität, 4400 Münster, Fliegerstr. 21

West Germany

Some comments responding to the "ICMI discussion document"

The Influence of Computers and Informatics on Mathematics
and its Teaching (L'Enseignement Mathématique, tome 30, 1984)

The ICMI study gives a well balanced view upon all important aspects of the topic. It is an encouraging paper starting a new discussion upon the changing foundations of mathematics, upon the philosophy of mathematicians, and upon the impact of both on teaching mathematics. Perhaps the paper still concentrates a bit too much upon a traditional view of mathematics. As a researcher in mathematics education (learning process, impact of technology on mathematics education) and a specialist in training teachers and teaching mathematics I would give some more emphasis to the following aspects.

1. The effect on mathematics

The computer will affect the selfcomprehension (in German: "Selbstverständnis") of mathematics and (pure) mathematicians in a radical way. The picture of "a mathematician working alone" (p. 164) will become less and less typical. Complete chains of reasoning from the question to the answer (from the axioms to the theorem) or explicit sequences of deducing will become too

complex and too artificial to be understood by a single person. Mathematicians will lose their independence. Using computers effectively they will become dependent on computer specialists. Using the computer they have to create team work and working groups. The group then will "be able to follow and verify every step" (p. 164), but not the individual.

A changing working style also will change basic ideas of traditional mathematics: "Chains of reasoning" will be replaced by "hierarchies of reasoning", the "sequence of logical steps" will be replaced by a "logical sequence of black boxes", "computing" will be replaced by "computability" (in German: "Berechenbarkeit").

Symbolic systems (p. 165) also may have a dramatic effect on mathematics. There will be a second progress to mathematical ideas and insight after the first being caused by numerical calculations. "An entirely new art of experimentation" (p. 163) also is possible with symbolic systems. Manipulations and simulations with software packages using spreadsheets and graphics for visualization will create cognitive dimensions not known yet to mathematical ideas and relationships.

2. The effect of computers on curricula

I am not sure if the present mathematics curricula perfectly meet the needs of the society. Many of our students (age > 16 years) probably will use the computer after their examinations just like a car, a washing machine, a typewriter or like television or telephone. Did we teach them the

137

appropriate mentality for that kind of use, did we teach them how to use the computer most effectively under these circumstances? The creative use of computers probably will be restricted to a small minority of our population. Only those might "demand more mathematics, better understood" (p. 166).

Maybe that we have to accept changes. Mathematics in the curricula has to become more dynamic and more algorithmic. A static view of mathematics is no help for producing a computer using mentality. Also the working style in mathematics education might change. No longer fights one against all the others, but more team work is necessary. Discussing the dangers of the computer use also might be part of a curriculum.

Using the computer most effectively the majority only may need

1. an introduction into the basics (little programming, little hardware and software knowledge, ...),
2. powerful experiences in solving successfully complex problems from different subject areas (using hardware and software mainly as black boxes),
3. explicit knowledge and experiences upon the possibilities and the limits of a computer use.

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

There are at least three advantages using a computer as a teaching aid: visualization, interaction, and speed.

The visualization of mathematics on a plotter or on a screen enriches the learning process. Aspects of Gestalt psychology may enter into the teaching process by using graphics, spreadsheets, etc. Also the knowledge about the recognition of shapes and patterns (either graphic or symbolic representations) may become more important.

The interaction between the learner and mathematics is one of the fundamental assumptions of the cognitive learning psychology. Learners do not simply add new informations to their store of knowledge. New knowledge has to be "constructed" by the learner, has to be "invented" by himself. The computer (with appropriate software learning packages) can become a powerful tool within the process of learning.

Also the speed of a computer is essential. Each answer of the computer gives a feed back to the learner. Many computer tutorials give emphasis on that feed back. (Our research project concentrates on feed back by using guess-and-test procedures). Since the years of Programmed Instruction it is well known, that reinforcement is the more effective the quicker it is.

Extrait de :

Société Mathématique de France
Astérisque 109-110 (1983) p.235-244

139

ÉLÉMENTS POUR UNE THÉORIE DU CONTINU

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 sont 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 : 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

(*) 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

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 Keeb 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.

Posons alors les définitions suivantes :

a) Nous dirons que deux éléments k et l de \mathbb{Z} sont équivalents, $k \simeq l$, si pour tout naïf n , $n|k-l| \leq \omega$.

b) Nous dirons qu'un élément k de \mathbb{Z} est limité (2) s'il existe un naïf n tel que $|k| \leq n\omega$.

A propos de ces deux définitions, il faut être bien conscient de leur caractère externe : elles ne peuvent être posées que par un intuitionniste qui observe l'arithmétique formelle de l'extérieur, et avec une bienveillance presque paternelle. Pour un esprit dogmatique qui refuse de sortir de sa chère théorie formelle des ensembles, ces définitions n'ont aucun sens.

La définition a) nous fournit, entre éléments de \mathbb{Z} , une relation d'équivalence externe (qui n'existe pas dans la théorie des ensembles) et donc aussi des classes d'équivalence (qui ne sont pas des ensembles). Nous appellerons halos ces classes d'équivalence. Puis, nous posons encore la définition :

c) Un nombre réel est le halo d'un élément limité de \mathbb{Z} .

Donnons tout de suite des exemples de nombres réels :

1. Les entiers $\omega, 2\omega, 3\omega$, et plus généralement les multiples naïfs de ω , sont limités. Leurs halos sont donc par définition des nombres réels.

2. Considérons l'ensemble $D = \{(k, l) \in \mathbb{Z} \times \mathbb{Z} \mid k^2 + l^2 < \omega\}$. Cet ensemble est fini ; donc, d'après un théorème de la théorie des ensembles, son cardinal est un élément a de \mathbb{N} ; on vérifie facilement que $a \leq 4\omega$, c'est à dire que a est limité. Son halo est donc un nombre réel, que nous appellerons π .

3. ω^ω et $(\omega+1)^{\omega+1}$ sont deux éléments de \mathbb{N} ; on peut faire la division euclidienne du second par le premier : $(\omega+1)^{\omega+1} = \omega^\omega b + r$, $0 \leq r < \omega^\omega$; en développant $(\omega+1)^\omega$ suivant la formule du binôme, un raisonnement d'arithmétique très simple montre que $(\omega+1)^\omega \leq 3\omega^\omega$, donc b est limité ; son halo est un nombre réel, désigné par le symbole e .

En analyse combinatoire, il est souvent nécessaire de diviser des entiers, et par conséquent il est commode de recourir aux fractions si on veut éviter que l'écriture ne devienne trop lourde. De même que sur \mathbb{Z} , on peut définir sur Q une relation d'équivalence externe, des éléments limités, etc :

DÉFINITIONS. Soit $r \in Q$. Nous dirons que r est infinitement petit, ou négligeable, si pour tout naIf n , $n|r| \leq 1$. Nous dirons que r est limité s'il existe un naIf n tel que $|r| \leq n$. Nous dirons que deux éléments r et s de Q sont infinitement voisins (et noterons $r \approx s$) si $r-s$ est infinitement petit.

A tout nombre rationnel r limité, on peut associer un nombre réel, que nous appellerons son ombre, et que nous noterons $St(r)$: en effet, r s'écrit sous forme de fraction irréductible $r = \frac{p}{q}$ avec $q > 0$. Soit $m(r)$ le quotient euclidien de $p\omega$ par q ; si r est limité dans Q , $m(r)$ est limité dans \mathbb{Z} , donc son halo est un nombre réel qui par définition sera l'ombre de r .

Deux rationnels limités et infinitement voisins ont la même ombre (Exercice : le démontrer).

- [1] Georges REEB : La mathématique non-standard, vieille de soixante ans ?
Publ. IRMA, 1978.
- [2] G. TAKEUTI : Two applications of Logic to Mathematics (Princeton University Press, 1976).
- [3] Les entretiens de Zürich sur les fondements des sciences mathématiques.
Editeur : S.A. Lehmann frères & Cie, Zürich, 1941.
- [4] D. HILBERT : Ueber das Unendlichen. Mathematische Annalen, vol.95, 1926,
p 161 - 190.

NOTES

(1) On peut considérer que cette question demeure posée depuis Brouwer, quoique sous une forme moins directe (ou moins brutale).

(2) Plus loin, dans Q nous donnerons à ce mot un sens légèrement différent, plus proche du sens usuel. Dans Q , l'échelle sera ω fois plus petite.

A FUNDAMENTAL COURSE IN HIGHER MATHEMATICS
INCORPORATING DISCRETE AND CONTINUOUS THEMES

by

Michael D. Rice and Stephen B. Seidman
George Mason University
Fairfax, Va 22030, USA

Traditionally, students' first exposure to higher mathematics comes in the form of the differential and integral calculus, either in secondary school or in the first year of university studies. This initial exposure emphasizes the fundamental role of continuous mathematics in contemporary mathematics curricula, which can be attributed to the historical success of continuous mathematical models in the physical sciences and engineering. In recent decades, increasing attention has been paid to discrete mathematical models, motivated by the ever greater role played by computers both in a wide variety of applications and as a study of intrinsic importance. The widespread interest in these models has led to the introduction of discrete mathematics courses into university curricula. Unfortunately, these courses are generally offered at an advanced stage in students' academic careers, so that the discrete mathematical tools are acquired too late to be effectively used in the study of computer science. In addition, these tools are not well integrated with the more traditional continuous mathematical tools learned earlier. Several proposals have been made in the United States to address this problem by restructuring the first two years of university

mathematics. These proposals usually place discrete mathematics courses before continuous mathematics courses, at the beginning of students' university careers.

In this paper, it will be argued that these proposals do not respond to students' perceptions that discrete and continuous mathematics are unrelated subjects, and therefore will not provide students with a flexible collection of mathematical tools that can be applied in both discrete and continuous situations. Such a collection of tools is absolutely essential for the study of computer science, as well as for the modern study of the physical sciences and engineering. The paper will outline a fundamental course in higher mathematics that introduces both discrete and continuous ideas in a synergistic, mutually reinforcing manner, providing students with both the discrete models that are needed for the study of computer science and the continuous models that are needed for the more traditional study of the physical sciences and engineering.

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

$$D = Q*d + R \text{ and } 0 \leq R < d.$$

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 the stack.
			Output Q = 3, R = 6				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

$$D = 0, 1, 2, \dots, k.$$

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.

147

APPENDIX

SECTION 1 - 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, $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$). 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 [Fibonacci] 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.

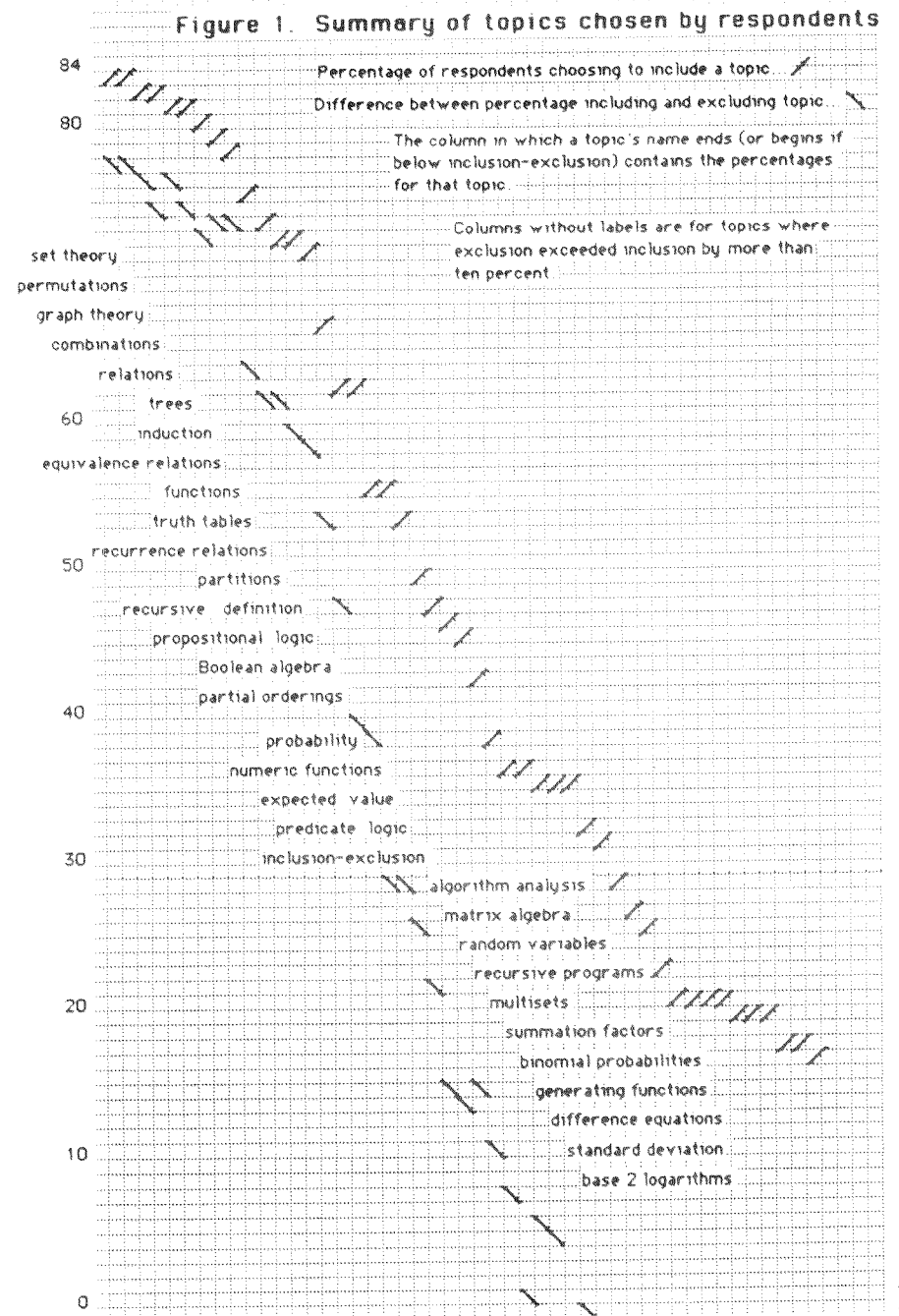
What Should a Discrete Mathematics Course Be?

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 questionnaires 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. These courses are likely no more sophisticated or proof-oriented in their approach than freshman calculus, and are 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, combinatorics and probability. We also asked respondents to indicate which topics on the menu should not 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 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 College Algebra topics 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 the number system. Because the panel is recommending a one year freshman course (which means that weak students might not be able 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 suggests 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 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 this 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 three 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 semester in which it may require students to take a course in abstract algebra, combinatorics, graph theory, linear algebra and linear programming, logic, probability or statistics. 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.

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 for the student. Our course at Dartmouth leaves the second half of Unit 6 and all of Unit 7 for a later (optional) course in combinatorics. With regrets, we plan to cover only as much of Unit 10 as time allows; I expect time may allow little or none of Unit 10. Since our computer science majors do not flock to logic courses, we may yet try to redesign the course to allow coverage of Unit 10. Since our course devotes only three quarter hours to discrete mathematics and one quarter hour to computer programming and computer science topics such as sorting and searching, and since our stronger students take a more rigorous course, I believe it is reasonable to base a course for average American students on this outline.

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 logarithmic 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)*

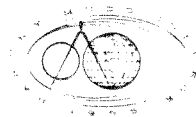
Unit 9 **Matrix Algebra*** Matrix operations, matrix equations and systems of linear equations, inverse matrices, determinants, applications of matrix powers to graphs and Markov Chains, Hamming codes.

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.

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 career. 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



HARVEY
MUDD
COLLEGE
CLAREMONT,
CALIFORNIA
91711

DEPARTMENT
OF
MATHEMATICS
714-621-8023

Symbolic Manipulators
and the
Teaching of College Mathematics:
Some Possible Consequences and Opportunities

Kenneth D. Lane
Department of Mathematics
Harvey Mudd College
Claremont, CA 91711 USA

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.

November 27, 1984

Dr. F. Pluinage
IREM
10 rue du General Zimmer
67084 Strasbourg, Cedex
France

Dear Professor Pluinage:

Enclosed is an abstract for consideration for the March meeting. The work described in my paper is part of a curriculum project at Colby College being funded by the Alfred P. Sloan Foundation. I am on the faculty at Colby but am on leave during this academic year. The overall goal of the Sloan project is to build a two-year curriculum which provides a balance between calculus and discrete mathematics.

We are very excited about our experiences using these new technologies. However, the problems and issues that accompany the uses of these systems are enormously complex. A vigorous discussion among far-sighted individuals who are interested in curricula matters is needed to help identify and solve these problems. I suspect that this meeting would provide a forum from which this discussion might begin.

I have made a preliminary inquiry to A. Ralston with respect to the appropriateness of this paper for the meeting. During my discussion with him he indicated that some funding may be available. In the event that you are interested in my paper, please consider me for any funding that may be available.

I hope that you find my proposal useful and I look forward from hearing from you.

Sincerely,

Kenneth D. Lane

Kenneth D. Lane
Visiting Assistant Professor
of Mathematics

KDL/ssc

153

USING COMPUTER SYMBOLIC MATH FOR LEARNING BY DISCOVERY

by
David R. Stoutemyer
University of Hawaii
Honolulu

December, 1984

"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.

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 a 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:

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™ 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:

- the number of terms;
- relations among the exponents in successive terms;
- symmetries among the coefficients for a particular n ;
- relations among coefficients for two successive values of n ;
- relations between a coefficient and factorials involving the corresponding exponents;
- the asymptotic growth of the largest coefficient with n .
- 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;           "Let's set the expansion control variable to
o1: TRUE                   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 o_6 equals the sum of the coefficient 3 directly above and the coefficient 3 to its left in o_5 . 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: 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
```

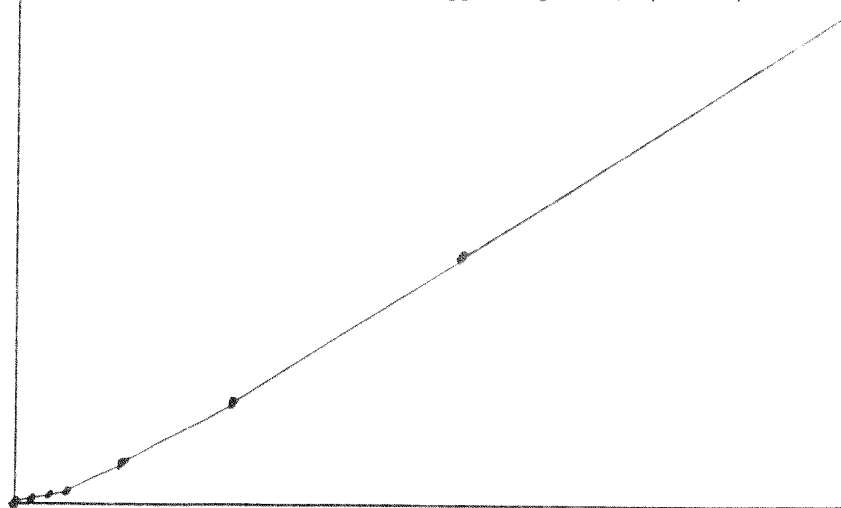
"This sequence will soon become too time consuming for interactive exploration. If I decide to do more, I'll write a procedure containing a loop and run it overnight. A vague pattern of sorts has already emerged anyway:"

"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 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(60,10)],
[16, LOG(180,10)], [32, LOG(180,10)], [64, LOG(180,10)]);
```

```
o20: lower left corner = (1, 0), upper right = (64, 18.26):
```



154

"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:"

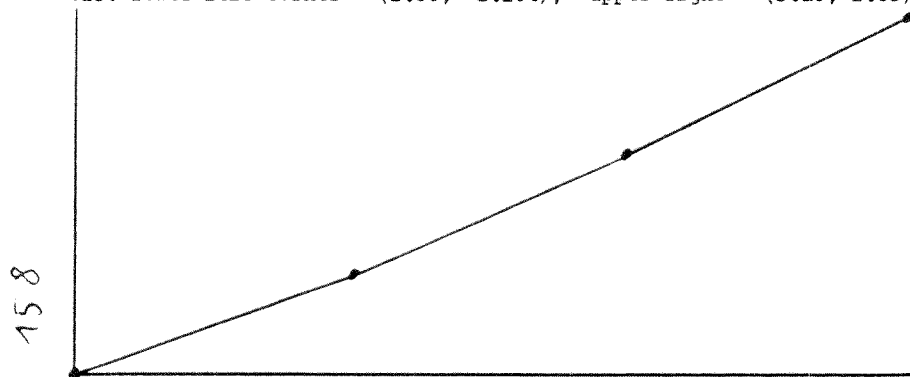
```
i23: SUBST (o22, n: 128);
o23: 4765.61
```

"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

$$\frac{1}{(1 - x^2/(3 - x^2/5))},$$

$$\frac{1}{(1 - x^2/(3 - x^2/(5 - x^2/7)))},$$

$$\frac{1}{(1 - x^2/(3 - x^2/(5 - x^2/(7 - x^2/9))))},$$

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

$$\begin{bmatrix} a & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} b & 1 \\ 1 & 0 \end{bmatrix}, \quad \begin{bmatrix} a & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} b & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} c & 1 \\ 1 & 0 \end{bmatrix}, \quad \text{etc.,}$$

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.

$$\text{a) } \begin{vmatrix} 1 & 1 & 1 & 1 \\ 1 & a+1 & 1 & 1 \\ 1 & 1 & b+1 & 1 \\ 1 & 1 & 1 & c+1 \end{vmatrix}$$

$$\text{d) } \begin{vmatrix} 1 & -1 & 0 & 0 & 0 \\ x & h & -1 & 0 & 0 \\ x^2 & hx & h & -1 & 0 \\ x^3 & hx^2 & hx & h & -1 \\ x^4 & hx^3 & hx^2 & hx & h \end{vmatrix}$$

$$\text{b) } \begin{vmatrix} 1 & a & b & c \\ a & 1 & 0 & 0 \\ b & 0 & 1 & 0 \\ c & 0 & 0 & 1 \end{vmatrix}$$

$$\text{e) } \begin{vmatrix} 1 & a & a^2 & a^3 \\ 1 & b & b^2 & b^3 \\ 1 & c & c^2 & c^3 \\ 1 & d & d^2 & d^3 \end{vmatrix}$$

$$\text{c) } \begin{vmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & b & b \\ 1 & b & 0 & b \\ 1 & b & b & 0 \end{vmatrix}$$

$$\text{f) } \begin{vmatrix} x & 0 & 0 & y \\ y & x & 0 & 0 \\ 0 & y & x & 0 \\ 0 & 0 & y & x \end{vmatrix}$$

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 algorithm 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:

$$(1 - x)^{-1} = 1 + x + x^2 + x^3 + \dots$$

$$(1 - x)^{-2} = 1 + 2x + 3x^2 + 4x^3 + \dots$$

$$(1 + x)/(1 + x + x^2) = 1 - x^2 + x^3 - x^5 + x^6 - x^8 + x^9 - \dots$$

$$(1 + x)/(1 - x)^3 = 1 + 4x + 9x^2 + 16x^3 + 25x^4 + \dots$$

Then, using these as building blocks or inspirations, see if you can experimentally discover rational expressions having the following power series expansions:

a) $1 - x + x^2 - x^3 + x^4 - x^5 + \dots$

b) $1 + 2x + 4x^2 + 8x^3 + 16x^4 + 32x^5 + \dots$

c) $1 + 2x + 3x^2 + x^3 + 2x^4 + 3x^5 + x^6 + 2x^7 + 3x^8 + \dots$

d) $1 + 2x + 3x^2 + 2x^3 + x^4 + 2x^5 + 3x^6 + 2x^7 + x^8 + \dots$

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!

160

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

$$y'' = -y + \delta y^2 \tag{1}$$

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

$$y(0) = 0 \quad y'(0) = 1 \quad (2)$$

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 :

$$u = x(1 + c_1 \delta + c_2 \delta^2 + \dots)$$

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

$$\ddot{y}(1 + c_1 \delta + c_2 \delta^2 + \dots)^2 = -y + \delta y^2 \quad (3)$$

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

$$y = y_0 + y_1 \delta + y_2 \delta^2 + \dots$$

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) *  $\delta$  * * i;
y := for i := 0 : n sum y(i,u) *  $\delta$  * * i;
l := d f (y,u,2) * s * * 2;
```

The righthand side of the equation is assigned to a variable r by

```
r := -y +  $\delta$  * 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, $\delta$ ,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

$$y(0,u) := \sin(u); \quad c(0) := 1;$$

168

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

$$\text{for all } x \text{ let } \sin(x) ** 2 = (1 - \cos(2 * x))/2;$$

We are then left with

$$ar(1) = -2c_1 \sin(u) + \frac{1}{2} \cos(2u) + y_1 + \ddot{y}_1 - \frac{1}{2}$$

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 :

$$a_{11} \sin(u) + b_{11} \cos(u) + b_{12} \cos(2u) + \text{const}_1$$

After substituting this into the formula for $ar(1)$ it is easily seen that $ar(1)$ vanishes if and only if

$$c_1 = 0 \quad b_{12} = \frac{1}{6} \quad \text{const}_1 = \frac{1}{2}$$

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 u say

$$y(0) = 0 \quad \dot{y}(0) = 1.$$

This is satisfied up to the first degree of δ if and only if sub ($u = 0, y(1,u)$) and sub ($u = 0, \frac{df}{du}(y(1,u), u, 1)$) 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

$$-2c_2 \sin(u) - \sin(u) - \frac{1}{3} \cos(2u) \sin(u) + \frac{4}{3} \cos(u) \sin(u) + y_2 + \ddot{y}_2$$

As there are two products of sines and cosines in this expression we introduce the rules

$$\begin{aligned} \text{for all } x \text{ let } \cos(2 * x) * \sin(x) &= (\sin(3 * x) - \sin(x))/2; \\ \text{for all } x \text{ let } \cos(x) * \sin(x) &= (\sin(2 * x))/2; \end{aligned}$$

This gives for $ar(2)$

$$-2c_2 \sin(u) - \frac{5}{6} \sin(u) + \frac{2}{3} \sin(2u) - \frac{1}{6} \sin(3u) + y_2 + \ddot{y}_2$$

From this formula it is seen that we must use the following form for y_2 :

$$a_{21} \sin(u) + b_{21} \cos(u) + a_{22} \sin(2u) + a_{23} \sin(3u) + \text{const}_2$$

Upon substitution in the expression for $\text{ar}(2)$ it is seen that $\text{ar}(2)$ vanishes if and only if

$$c_2 = 5/12 \quad a_{22} = 2/9 \quad a_{23} = -1/48 \quad \text{const}_2 = 0$$

The remaining integration constants a_{21} and b_{21} are again found from the integration conditions (2). We find from the requirements that $\text{sub}(u=0, y(2,u))$ and $\text{sub}(u=0, \text{df}(y(2,u), u, 1))$ must yield zero:

$$a_{21} = -\frac{55}{144} \quad b_{21} = 0$$

We have therefore

$$y_2 = \frac{55}{144} \sin(u) + \frac{2}{9} \sin(2u) - \frac{1}{48} \sin(3u)$$

Continuing in the same way we find

$$c_3 = 0 \\ y_3 = \frac{17}{216} \cos(u) - \frac{7}{108} \cos(2u) + \frac{1}{24} \cos(3u) - \frac{1}{432} \cos(4u) + \frac{5}{48}$$

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 l, 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 another in the "new" $\text{ar}(0) \dots \text{ar}(3)$ so that no new for all 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 $y(0) = 0 \quad y'(0) = 1$ as a truncated Fourier series in u , where

$$u = x (1 + c_1 \delta + c_2 \delta^2 + \dots)$$

by a semi-automatic method. We have used an interactive formula-manipulation system (REDUCE in fact) and we have determined step by step $c_i (i \geq 0)$ and $y_i (i \geq 0)$. Each step consisted of the following substeps:

- inspection of $\text{ar}(i)$ in which the coefficients of δ^i in $\ddot{y}^* (1 + c_1 \delta + \dots + c_i \delta^i)^2 + y - \delta y^2$ 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

165

condition $ar(i) = 0$

- determination of the two remaining coefficients (the integration constants) from the conditions $y_i(0) = 0$ $\dot{y}_i(0) = 0$

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 ICME5, Adelaide, forthcoming.

Appendix: REDUCE script

Below follows a set of instructions written in REDUCE for the symbolic solution of the differential equation

$$\ddot{y} = (-y + \delta \cdot y^2) s^{-2}$$

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 are currently engaged in preparing a VHS-video tape displaying an actual session based on the script.

```

COMMENT: REDUCE DEMO;
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,1)*S**2 $
R:=-Y+D*Y**2 $
ARRAY AR(3*N) $
NFACTORS := COEFF(L-R,U,AR);
CLEAR S,Y,L,R $
FOR I:=(N+1):NFACTORS DO AR(I):=0 $
AR(0);
COMMENT: AFTER INSPECTION CONCLUSIONS ARE DRAWN
AND NEW INFORMATION IS SUPPLIED TO THE
SYSTEM (SHOWN BY INDENTATION);
C(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)+L(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>> $
AR(1);
CUB(U=0,Y(1,U));
A(1,1):=-2/3 $
SUB(U=0,DF(Y(1,U),U,1));
A(1,1):=0 $
AR(2);
LET COS(2*U)*SIN(U) = (SIN(3*U)-SIN(U))/2 $
LET COS(U)*SIN(U) = (SIN(2*U))/2 $
AR(-);
Y(L,U):=A(L,1)*SIN(U)+B(2,1)*COS(U)+A(2,2)*
SIN(2*U)+B(2,3)*SIN(3*U)+CONST(2) $
AR(2);
<<C(2):=-3/12; CONST(2):=0; A(2,2):=2/9;
A(2,3):=-1/4>> $
AR(2);
SUB(U=0,Y(2,U));
A(2,1):=-55/144 $
SU(U=0,DF(Y(2,U),U,1));
A(2,1):=0 $
AR(0);
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(2*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):=3/48; B(3,2):=-23/432;
B(3,3):=42/1152; B(3,4):=-5/2160>> $
AR(3);
SUB(U=0,Y(3,U));
B(3,1):=-17/216 $
SUB(U=0,DF(Y(3,U),U,1));
A(3,1):=0 $
COMMENT: NOW THE FINAL CHECKING;
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,1)*S**2 $
R:=-Y+D*Y**2 $
ARRAY BR(3*N) $
COEFF(S/3248*(L-R),0,BR) $
COMMENT: THE FACTOR 3248 RENDERS THE FIRST ARGUMENT
OF COEFF A POLYNOMIAL WITH INTEGER COEFF.;
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));
S;
Y(0,U);
Y(1,U);
Y(2,U);
Y(3,U);
END;

```

166

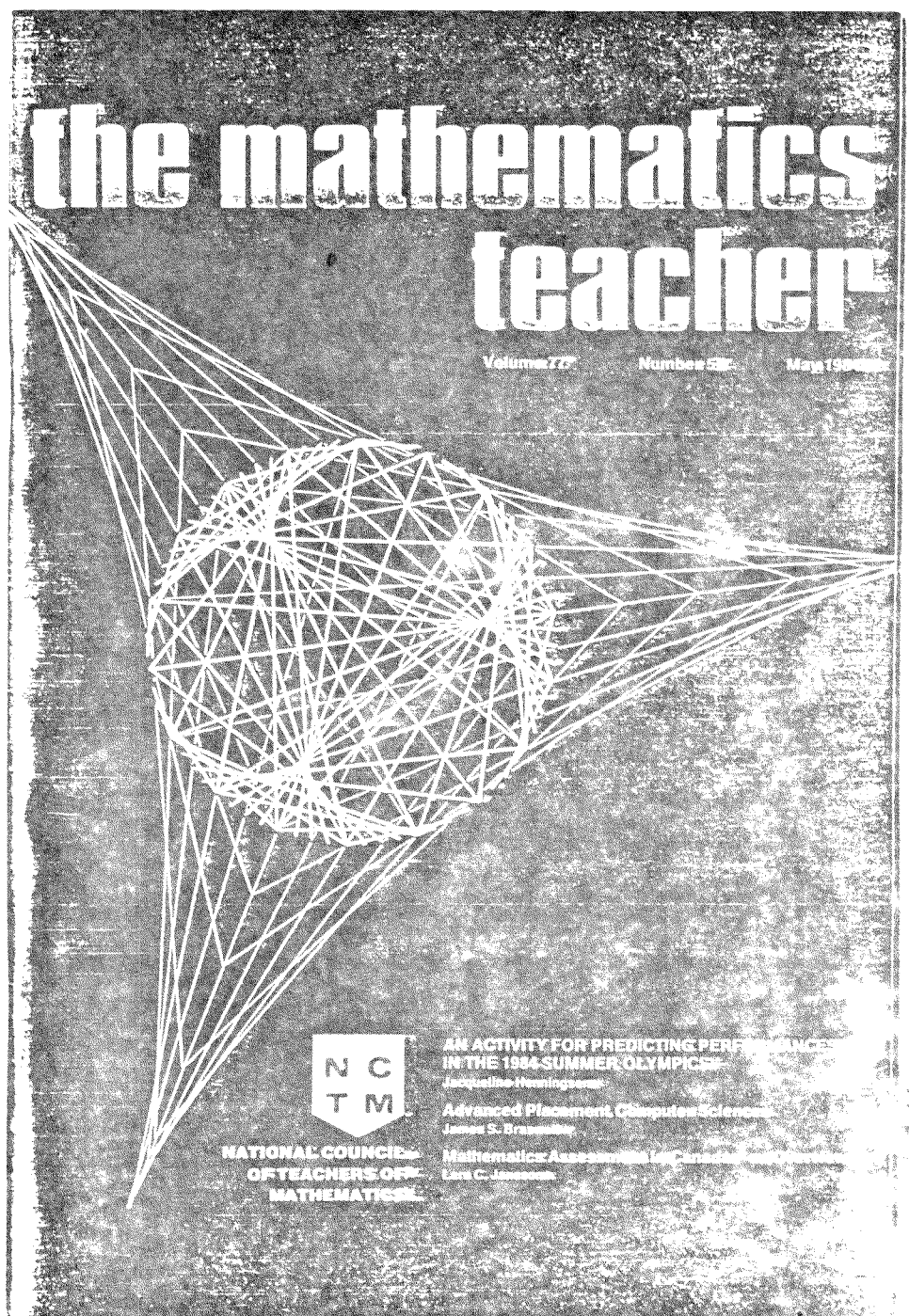
Contributor: Clark Kimberling
Professor of Mathematics
University of Evansville
Evansville, IN 47702

Topic: Microcomputer Programming in BASIC for
Mathematics Students

Abstract: This work examines (1) topics that lend themselves especially well to microcomputer assistance, both through student-writing of programs, and through student use of programs, and (2) the balance between textbook and microcomputer. Examples will be drawn from the author's series, "Microcomputer-assisted Discoveries" in The Mathematics Teacher.

the mathematics teacher

Volume 77 Number 2 May 1984



NATIONAL COUNCIL
OF TEACHERS OF
MATHEMATICS

AN ACTIVITY FOR PREDICTING PERFORMANCE IN THE 1984 SUMMER OLYMPICS

Jackie L. Henningson

Advanced Placement Computer Science

James S. Bransford

Mathematical Association for Children

Lara C. Johnson

TABLE 1

```

10 REM GRAPH MANY CONICS
20 DEF FN I(X) = INT (.5 + X)
30 DEF FN X(U) = (U - 140) / S
40 DEF FN Y(V) = (96 - V) / T
50 DEF FN U(X) = 140 + S * X
60 DEF FN V(Y) = 96 - T * Y
70 S = 10; T = 10
80 HOME : PRINT "THIS PROGRAM GRAPHS
EQUATIONS OF CONICS"
90 PRINT : POKE 36,2
100 PRINT "AX^2 + BXY + CY^2 + DX + EY
+ F = 0"
110 PRINT : POKE 38,15; PRINT "EXAMPLES:"
PRINT
120 PRINT "A B C D E F GRAPH"
130 FOR I = 0 TO 39: PRINT "I:"; NEXT I:
PRINT
140 PRINT "1 0 1 0 0 -25 CIRCLE"
150 PRINT "1 0 2 0 0 -25 ELLIPSE,
VERT."
160 PRINT "2 0 1 0 0 -25 ELLIPSE,
HORIZ."
170 PRINT "1 0 -1 0 0 -25
HYPERBOLA, VERT."
180 PRINT "1 0 -1 0 0 25
HYPERBOLA, HORIZ."
190 PRINT "0 1 0 0 0 -25
HYPERBOLA, 45 DEG."
200 PRINT "1 0 0 0 -9 0
PARABOLA, UP"
210 PRINT "0 0 1 9 0 0
PARABOLA, LEFT"
220 PRINT "0 0 0 >0 >0 <0 LINE
DX+BY+C=0"
230 PRINT "1 0 -1 0 0 0 TWO
LINES"
240 PRINT : PRINT "YOU REALLY SHOULD TRY
MANY OTHERS, TOO."
250 PRINT : PRINT "PLEASE INPUT
COEFFICIENTS A,B,C,D,E,F:"
260 INPUT "A,B,C,D,E,F"
270 HOME : HGR : HCOLOR = 3
280 IF B = 0 THEN 320
290 IF A = C THEN G = ATN (1): GOTO 310
300 G = .5 * ATN (B / (A - C))
310 U = COS (G); V = SIN (G)
320 A1 = A; B1 = B; C1 = C; D1 = D; E1 = E; F1 = F
330 IF B = 0 THEN 390
340 A = A1 * U + B1 * U * V + C1 * V * V
350 B = B1 * (U * U - V * V) + 2 * (C1 - A1) * U *
V
360 C = A1 * V * V - B1 * U * V + C1 * U * U
370 D = D1 * U + E1 * V
380 E = -D1 * V + E1 * U; F = F1
390 IF C = 0 THEN 430
400 Q = D * D - E * E * A / C - 4 * A * F
410 Q1 = SQR (ABS (Q))
420 K = -E / (2 * C)
430 IF A = 0 THEN 460
440 H = -D / (2 * A)
450 R = ABS (Q1 / (2 * A))
460 A = A1; B = B1; C = C1; D = D1; E = E1; F = F1
470 FOR U = 0 TO 279: HPLLOT U,96: NEXT U
480 FOR V = 190 TO 0 STEP -1: HPLLOT 140,V:
NEXT V
490 FOR U = 140 TO 279 STEP S: HPLLOT U,96
TO U,98
500 HPLLOT 280 - U,96 TO 280 - U,98: NEXT U
510 FOR V = 96 TO 191 STEP T: HPLLOT 138,V TO
140,V
520 HPLLOT 138,192 - V TO 140,192 - V: NEXT V

```

```

530 VTAB 21: PRINT "PLEASE WAIT. IF NO
GRAPH APPEARS, IT"
540 PRINT "IS OFF THE SCREEN." CURRENT
COEFFICI:"
550 PRINT "ENTS ARE
"A","B","C","D","E","F"
560 IF C = 0 THEN 710
570 FOR U = 0 TO 279
580 X = FN X(U)
590 B1 = B * X + E
600 D1 = B1 * B1 - 4 * C * (A * X * X + D *
X + F)
610 IF D1 < 0 THEN 690
620 Y1 = (-B1 - SQR (D1)) / (2 * C)
630 Y2 = (-B1 + SQR (D1)) / (2 * C)
640 V1 = FN I (FN V(Y1)); V2 = FN I (FN V(Y2))
650 IF V1 < 0 OR V1 > 159 THEN 670
660 HPLLOT U,V1
670 IF V2 < 0 OR V2 > 159 THEN 690
680 HPLLOT U,V2
690 NEXT U
700 GOTO 1090
710 IF A = 0 THEN 860
720 FOR V = 159 TO 0 STEP -1
730 Y = FN Y(V)
740 B1 = B * Y + D
750 D1 = B1 * B1 - 4 * A * (E * Y + F)
760 IF D1 < 0 THEN 840
770 X1 = (-B1 - SQR (D1)) / (2 * A)
780 X2 = (-B1 + SQR (D1)) / (2 * A)
790 U1 = FN I (FN U(X1)); U2 = FN I (FN U(X2))
800 IF U1 < 0 OR U1 > 279 THEN 820
810 HPLLOT U1,V
820 IF U2 < 0 OR U2 > 279 THEN 840
830 HPLLOT U2,V
840 NEXT V
850 GOTO 1090
860 IF B = 0 THEN 960
870 FOR U = 0 TO 279
880 X = FN X(U)
890 IF X = -E / B THEN 940
900 Y = (-D * X - F) / (B * X + E)
910 V = FN I (FN V(Y))
920 IF V < 0 OR V > 191 THEN 940
930 HPLLOT U,V
940 NEXT U
950 GOTO 1090
960 IF E = 0 THEN 1050
970 FOR U = 0 TO 279
980 X = FN X(U)
990 Y = -D * X / E - F / E
1000 V = FN I (FN V(Y))
1010 IF V < 0 OR V > 159 THEN 1030
1020 HPLLOT U,V
1030 NEXT U
1040 GOTO 1090
1050 IF D = 0 THEN 1090
1060 FOR V = 159 TO 0 STEP -1
1070 HPLLOT FN U(-F / D),V
1080 NEXT V
1090 HOME : VTAB 21
1100 PRINT "READY TO GRAPH YOUR NEXT
EQUATION."
1110 PRINT "INPUT A,B,C,D,E,F:"
1120 INPUT "A,B,C,D,E,F"
1130 GOTO 280
1140 END
]RUN
THIS PROGRAM GRAPHS EQUATIONS OF
CONICS
AX^2 + BXY + CY^2 + DX + EY + F = 0

```

TABLE 1—Continued

EXAMPLES:							YOU REALLY SHOULD TRY MANY OTHERS, TOO.
A	B	C	D	E	F	GRAPH	
1	0	1	0	0	-25	CIRCLE	PLEASE INPUT COEFFICIENTS A,B,C,D,E,F: 1.0,1.0,0,-25
1	0	2	0	0	-25	ELLIPSE, VERT.	PLEASE WAIT. IF NO GRAPH APPEARS, IT IS OFF THE SCREEN. CURRENT COEF- FICIENTS ARE 1.0,1.0,0,-25
2	0	1	0	0	-25	ELLIPSE, HORIZ.	READY TO GRAPH YOUR NEXT EQUATION. INPUT A,B,C,D,E,F: 1.0,2.0,0,-25
1	0	-1	0	0	-25	HYPERBOLA, VERT.	PLEASE WAIT. IF NO GRAPH APPEARS, IT IS OFF THE SCREEN. CURRENT COEF- FICIENTS ARE 1.0,2.0,0,-25
1	0	-1	0	0	25	HYPERBOLA, HORIZ.	READY TO GRAPH YOUR NEXT EQUATION. INPUT A,B,C,D,E,F: 1.0,2.0,0,-25
0	1	0	0	0	-25	HYPERBOLA, 45 DEG.	
1	0	0	0	-9	0	PARABOLA, UP	
0	0	1	9	0	0	PARABOLA, LEFT	
0	0	0	>0	>0	<0	LINE DX+BY+C=0	
1	0	-1	0	0	0	TWO LINES	

ola, and hyperbola—can serve as subjects of many discovery-oriented programs for microcomputers. This article offers two such programs, "Graph Many Conics" (table 1) and "Conic through Five Points" (table 2). Both programs enable students to simulate the "explosion" of ellipses into parabolas and then into hyperbolas.

These Applesoft BASIC programs contain PRINT statements that explain how students can enter their own ideas into the programs.

"Graph Many Conics"

Every conic section has an equation of the form

$$AX^2 + BXY + CY^2 + DX + EY + F = 0.$$

The program "Graph Many Conics" (table 1) allows the user to input the constants A, B, C, D, E, and F; then it graphs the conic section and invites another choice of constants. Thus, many different conics can be seen on the screen.

Unfortunately, in many algebra textbooks, the so-called mixed term BXY is not discussed, so that the only conics that are discussed there are those having axes parallel to the x- and y- axes. The reason for this, I suppose, is that the formula for the "angle of rotation," in case $B \neq 0$, uses the tangent or cotangent function, which are not usually discussed in algebra courses. However, microcomputer graphics enables students to discover on their own the effect of the

mixed term.

To keep the program in table 1 to a minimal length and running time, REM statements have been omitted. However, the following could be inserted:

```

25 REM TRANSFORMATION FORMULAS
BETWEEN (U,V) APPLE
COORDINATES AND
TRADITIONAL (X,Y)
CARTESIAN COORDINATES
335 REM TRANSFORM CONIC SECTION
BY ROTATING
465 REM PLOT X,Y AXES AND
CALIBRATE THEM
565 REM IF C <> 0 THEN FOR EACH X,
SOLVE FOR LOWER AND
UPPER Y, AND PLOT THE
CORRESPONDING
POINTS (U,V)
715 REM IF C = 0 AND A <> 0 THEN FOR
EACH Y, SOLVE FOR LEFT AND
RIGHT X, AND PLOT THE
CORRESPONDING
POINTS (U,V)
865 REM IF A = 0 AND C = 0 AND B <> 0
THEN FOR EACH X, SOLVE FOR
Y, AND PLOT THE ROTATED
HYPERBOLA
965 REM IF A = 0, C = 0, B = 0, AND
E <> 0 THEN FOR EACH X,
SOLVE FOR Y, AND PLOT THE
LINE
1045 REM IF A,B,C,E ALL = 0 AND D <> 0
THEN PLOT THE VERTICAL
LINE

```

Here is a list of experiments my students have conducted using the program in table 1.

769

```

TABLE 2
0  P5V  CONIC THROUGH 5 POINTS
10 DEF FN H(X) = INT (.5 + X)
20 DEF FN Y(V) = (96 - V) * T
30 DEF FN U(X) = 140 + S * X
40 DEF FN V(Y) = 96 - T * Y
5  S = 4 : S = 10 : T = 10
6  Z9 = 1 + INT (3 * RND (1)) + 4 * INT (2 * RND
  1)
20 HOME : PRINT "THIS PROGRAM GRAPHS
  THE UNIQUE CONIC"
30 PRINT "SECTION THAT PASSES THROUGH
  5 POINTS"
100 PRINT "WHICH YOU MAY INPUT." : PRINT
110 PRINT "IN ORDER TO SEE YOUR 5
  POINTS."
120 PRINT "AS WELL AS THE CONIC. KEEP"
130 PRINT : PRINT "      -13 < X < 13"
140 PRINT "      -9 < Y < 9"
150 PRINT : PRINT "FOR 1ST RUN, INPUT THE
  5 POINTS IN"
160 PRINT "ONE OF THESE": INVERSE :
  PRINT "COLUMNS": NORMAL
170 PRINT : PRINT
180 PRINT "      5.0      5.0      5.0      4.5"
190 PRINT "      0.5      0.5      0.5      5.4"
200 PRINT "     -5.0     -5.0     -5.0     2.4"
210 PRINT "      0.-5     0.-5     0.-5     4.2"
220 PRINT "      4.4      2.2      2.-2     -5.7"
230 PRINT "N = 5: FOR I = 1 TO N
240 PRINT "INPUT POINT #":
  X(I), Y(I) =
  INPUT "X(I), Y(I)";
250 PRINT "X(I), Y(I)";
260 X(I) = X(I) + RND (1) / (2 * S)
270 Y(I) = Y(I) + RND (1) / (2 * T)
280 W(I, 1) = X(I) * X(I)
290 W(I, 2) = X(I) * Y(I)
300 W(I, 3) = Y(I) * Y(I)
310 W(I, 4) = X(I) * W(I, 5) = Y(I): NEXT I: PRINT
320 HOME : PR "TO AVOID A SINGULAR
  MATRIX IN THE METH."
330 PRINT "OD OF SOLUTION, YOUR CHOICES
  OF X, Y ARE"
340 PRINT "SLIGHTLY RANDOMIZED TO NEW
  VALUES": PRINT
350 FOR I = 1 TO N: PRINT
  "X(I)", Y(I)" = "X(I)", "Y(I)": NEXT I
360 FOR I TO 1 TO N
370 W(I, 1) = W(I, 1) + 1: NEXT I
380 FOR M = 1 TO N: P = W(M, M) - 1
390 FOR J = 1 TO N: W(M, J) = W(M, J) / P:
  NEXT J
400 FOR I = 1 TO N: IF I = M THEN 430
410 Q = W(I, M): FOR J = 1 TO N
420 W(I, J) = W(I, J) - Q * W(M, J): NEXT J
430 NEXT I: NEXT M
440 FOR I = 1 TO N: W(I, 1) = W(I, 1) - 1: NEXT I
450 FOR I = 1 TO N: FOR J = 1 TO N
460 Z(I) = Z(I) - W(I, J): NEXT J
470 PRINT : PRINT "THE CONIC THROUGH
  YOUR 5 POINTS HAS"
480 PRINT : PRINT "COEF. OF X^2": POKE
  36, 17: PRINT Z(1)
490 PRINT "COEF. OF XY": POKE 36, 17:
  PRINT Z(2)
500 PRINT "COEF. OF Y^2": POKE 36, 17:
  PRINT Z(3)
510 PRINT "COEF. OF X": POKE 36, 17: PRINT
  Z(4)
520 PRINT "COEF. OF Y": POKE 36, 17: PRINT
  Z(5)
530 PRINT "CONSTANT TERM": POKE 36, 17:
  PRINT 1
540 PRINT : PRINT "PRESS 'RETURN'": GET
  AS: HOME
550 A = Z(1): B = Z(2): C = Z(3): D = Z(4): E =
  Z(5): F = 1
560 HGR : POKE 49234, 0: HCOLOR = 3
570 FOR I = 1 TO N
580 FOR V = 1 TO N
590 IF V < 0 OR V > 191 THEN 630
600 U1 = FN U(X(I)) - 3: U2 = FN U(X(I)) + 3
610 IF U1 < 0 OR U2 > 279 THEN 630
620 HPLLOT U1, V TO U2, V
630 NEXT V: NEXT I
640 FOR V = 1 TO 190 - R STEP R
650 HCOLOR = 3
660 Y1 = FN V(Y): Y2 = FN V(Y + R)
670 B1 = B * Y1 + D: B2 = B * Y2 + D
680 D1 = B1 * B1 - 4 * A * (C * Y1 * Y1 + E *
  Y1 + F)
690 D2 = B2 * B2 - 4 * A * (C * Y2 * Y2 + E *
  Y2 + F)
700 IF D1 < 0 OR D2 < 0 THEN 950
710 X1 = (-B1 - SQR (D1)) / (2 * A)
720 X2 = (-B1 + SQR (D1)) / (2 * A)
730 U1 = FN I (FN U(X1)): U2 = FN I (FN U(X2))
740 Z1 = (-B2 - SQR (D2)) / (2 * A)
750 Z2 = (-B2 + SQR (D2)) / (2 * A)
760 W1 = FN I (FN U(Z1)): W2 = FN I (FN U(Z2))
770 IF U1 < 0 OR U1 > 279 THEN 800
780 IF W1 < 0 OR W1 > 279 THEN 800
790 HPLLOT U1, V TO W1, V + R
800 IF U2 < 0 OR U2 > 279 THEN 830
810 IF W2 < 0 OR W2 > 279 THEN 830
820 HPLLOT U2, V TO W2, V + R
830 L = 5 * SGN (U2 - U1)
840 IF U1 < 0 THEN U1 = 0
850 IF U1 > 279 THEN U1 = 279
860 IF U2 < 0 THEN U2 = 0
870 IF U2 > 279 THEN U2 = 279
880 HCOLOR = Z9
890 IF U1 + L < 0 OR U1 + L > 279 THEN 920
900 IF U2 - L < 0 OR U2 - L > 279 THEN 920
910 HPLLOT U1 + L, V TO U2 - L, V
920 IF U1 + 2 * L < 0 OR U1 + 2 * L > 279 THEN
  950
930 IF U2 - 2 * L < 0 OR U2 - 2 * L > 279 THEN
  950
940 HPLLOT U1 + 2 * L, V + R TO U2 - 2 * L, V + R
950 NEXT V: FOR J = 1 TO 2000: NEXT J: PRINT
  : PRINT
960 TEXT : HOME : VTAB 9: PRINT "TO SEE
  THIS GRAPH AGAIN, TYPE 'Y'"
970 PRINT : PRINT "TO START ANOTHER RUN,
  TYPE 'N'"
980 GET CS: IF CS = "Y" THEN GOTO 560
990 RUN
]RUN
THIS PROGRAM GRAPHS THE UNIQUE CONIC
SECTION THAT PASSES THROUGH 5
POINTS WHICH YOU MAY INPUT.

```

170

TABLE 2—Continued

```

IN ORDER TO SEE YOUR 5 POINTS,
AS WELL AS THE CONIC, KEEP
      -13 < X < 13
      -9 < Y < 9
FOR 1ST RUN, INPUT THE 5 POINTS IN
ONE OF THESE COLUMNS:
      5.0      5.0      5.0      4.5
      0.5      0.5      0.5      5.4
     -5.0     -5.0     -5.0     2.4
      0.-5     0.-5     0.-5     4.2
      4.4      2.2      2.-2     -5.7
INPUT POINT #1: X(1), Y(1) = 5.0
INPUT POINT #2: X(2), Y(2) = 0.5
INPUT POINT #3: X(3), Y(3) = -5.0
INPUT POINT #4: X(4), Y(4) = 0.-5
INPUT POINT #5: X(5), Y(5) = 4.4
TO AVOID A SINGULAR MATRIX IN THE METHOD
OF SOLUTION, YOUR CHOICES OF X, Y
      ARE SLIGHTLY RANDOMIZED TO NEW
      VALUES:
      X(1), Y(1) = 5.00455083 0427708492
      X(2), Y(2) = 037902205 5.00340015
      X(3), Y(3) = -4.98755085 .0414218715
      X(4), Y(4) = 0445542131 -4.96199983
      X(5), Y(5) = 4.01284451, 4.02524676
      THE CONIC THROUGH YOUR 5 POINTS HAS
      COEF. OF X^2: -.0400637151
      COEF. OF XY: 0182138207
      COEF. OF Y^2: -.0402638506
      COEF. OF X: -8.93456891E-05
      COEF. OF Y: 9.13905503E-04
      CONSTANT TERM: 1
      PRESS 'RETURN'
      TO SEE THIS GRAPH AGAIN, TYPE 'Y'
      TO START ANOTHER RUN, TYPE 'N'

```

1. Plot $X^2 + CY^2 - Y = 0$ for $C = 2, 1, 1/2, 1/3, 0, -1/3, -1/2, -1,$ and -2 , corresponding to a plane slicing a cone so as to produce ellipses ($C > 0$) that suddenly become a parabola ($C = 0$).
 2. Plot $X^2 - Y^2 = 16, Y^2 - X^2 = 16, XY = 16,$ and $XY = -16$: these points give the same hyperbola in four different positions. What other equations give this same hyperbola in still other positions?
 3. Plot various pairs for $AX^2 + BY^2 = 36$ and $AX^2 - BY^2 = 36$, and compare each pair. Then compare these with $BX^2 + AY^2 = 36$ and $BX^2 - AY^2 = 36$.
 4. Plot families of concentric circles, ellipses, and so on, that are not necessarily centered at $(0, 0)$.
 5. Plot a hyperbola and its asymptotes. (They are a conic section, too.)
 6. Insert the line
465 GOTO 560
- and then plot the Olympics emblem shown in figure 3.



Fig 3. Olympic circles—symbols for continents

Suppose a computer user inputs five points in a plane and wishes to see the (unique) conic section passing through all of them. For many choices of points, it is not at all obvious what sort of conic will work—an ellipse, parabola, hyperbola, or one of the degenerate cases, such as a line or point. Since parabolas and the degenerate cases are "borderline" cases, one could, to simplify programming, very slightly randomize the five chosen points, thereby assuring that the conic section through them could be only an ellipse or a hyperbola. Also, the randomizing would assure that $F \neq 0$ the equation $AX^2 + BXY + CY^2 + DX + EY + F = 0$. Then, after dividing both sides by F , the resulting equation has the form $AX^2 + BXY + CY^2 + DX + EY + 1 = 0$; this equation has only five constants, to be determined by the choice of five points. The program can solve the system of five equations in five unknowns using matrix inversion, since because of the slight randomization it is almost certain that the matrix can be inverted. Table 2 is such a program. Although the graphics output is perhaps the most striking feature of this program, the printed equation of the conic also offers students opportunities for making discoveries. One assignment I have used has five

steps:

1. After running the examples provided by the program in lines 180 to 220, write an equation in the form

$$AX^2 + BXY + CY^2 + DX + EY + 1 = 0$$

using specific numbers for A, B, C, D, and E.

2. Determine five different points that satisfy your equation. (For various values of X, such as 0, -1, +1, use the quadratic formula to find Y, or vice versa.)

3. Input the five points. The equation of the conic through your points (after they are slightly randomized) will be printed. If its coefficients are nearly equal to the input coefficients, then step 2 was successful.

4. If the graph is an ellipse, can you move just *one* of the five points for another run to get a hyperbola, or vice versa? Experiment to find the *least* change in the five input points needed to change from one

kind of conic to the other.

5. Select five points whose conic is an ellipse, and call them P, Q, R, S, and T. Move P to a new location P' such that the conic through P', Q, R, S, and T is a hyperbola. Note that if P, Q, R, S, T, and P' all have *integral* coordinates, then the conics' equations will have only *integral* coefficients. Now, imagine P moving along a line to P'. The associated conic begins as an ellipse but eventually becomes a hyperbola. At some special point P' between P and P', the conic is a parabola. Use the "Graph Many Conics" program to find its equation.

Students should be encouraged to use the two programs interchangeably. For example, after step 3 in the previous assignment, students can input the equation into "Graph Many Conics" and check that the graph matches that given by "Conic through Five Points." ■

**Practical Resources for Teaching Mathematics
in the Secondary School Classroom**

■ **Activities from the MATHEMATICS TEACHER**, by Evan Miletzky and Christian Hirsch. Full of discovery lessons, laboratory experiences, games and puzzles, etc. Some topics included are computational skills, calculators, and geometry. Perforated pages, with teacher's guide and student worksheets. 5 1/2 x 11 in. 1981, 140 pp., #53, \$7.00.

■ **Films in the Mathematics Classroom**, by Barbara J. Bestgen and Robert E. Revs. You probably had no idea there are so many available in the U.S. and Canada. This book lists them and shows you how to use films effectively in teaching math. Includes candid reviews and ratings from teachers. 1982, 90 pp., #292, \$3.85.

■ **How to Evaluate Mathematics Textbooks**. Variable help for you during those times when you have to select textbooks. Contains a simple rating system for objective evaluation and discusses various criteria. 1982, 5 pp., #139, \$1.00.


■ **How to Evaluate Your Mathematics Program**. A guide for assessing your program's effectiveness and setting directions for change. Lists 21 standards to go by; prepared by a committee of 15 teachers, principals, and supervisors. 1981, 24 pp., #301, \$2.00.

■ **Mathematics: Gateway to Future Careers**. A full-color sound filmstrip that illustrates numerous mathematics-related careers. Should be seen by students, teachers, counselors, and parents. 1981, 15 min., #218, \$25.00.

■ **Mathematics Projects Handbook**, by Adrien L. Hess. Revised and updated; consult this book when choosing intriguing subject matter for math projects. Now has a section on calculators; extensive bibliography. 1982, 48 pp., #91, \$1.25.

See NCTM Materials Order Form in "New Publications"

NATIONAL COUNCIL OF TEACHERS OF MATHEMATICS
1906 Association Drive • Reston, Virginia • 22091



EXPERIENCES sur les APPORTS de L'INFORMATIQUE

à L'ENSEIGNEMENT DES MATHÉMATIQUES

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

Parallélisme entre le processus de résolution d'un problème mathématique et la construction d'un algorithme

Les méthodes de construction d'algorithmes et de raisonnement mathématique ont en commun :

- la définition d'un nombre restreint d'outils (symboles, actions élémentaires et procédures soigneusement formalisées),

- 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, à un 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.

173

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 1)

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 : $U_{n+1} = 2U_n + 1$ pour $U_n = 2^{n-1}$).

3. 2. Expérience LOGO (classe de CM2)

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 résout - laissant à l'enseignant le choix des situations-problèmes et son rôle de guide des enfants.

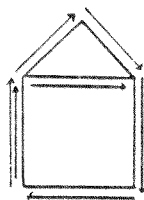
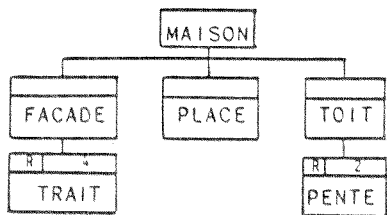
L'intérêt principal d'une programmation active réside dans la démarche algorithmique qu'elle nécessite. L'idée qui nous intéresse est celle de l'aide que peut apporter à l'enfant cette démarche dans la résolution de problème. Amener l'élève à avoir une démarche structurée et à décomposer les difficultés qu'il rencontre est le but que doit permettre d'atteindre l'utilisation des procédures. En cela LOGO est un langage particulièrement adapté.

174

3. 2. 1. Approche graphique

Le traitement du projet MAISON a servi de support pour formaliser une opération déjà plusieurs fois effectuée : analyser le problème, décomposer le projet en une succession ordonnée de petites difficultés pouvant être traitées indépendamment l'une de l'autre.

Les enfants se familiarisent rapidement avec cette nouvelle variété d'arbres :

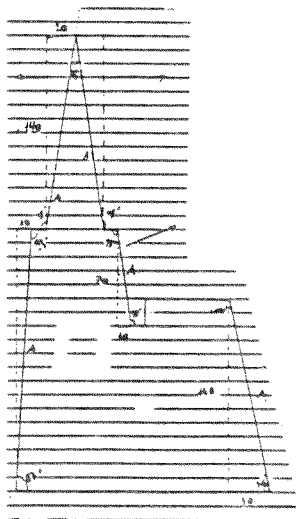


```

POUR MAISON
FACADE
PLACE
TOIT
FIN
POUR FACADE
REPETE 4 [TRAIT]
FIN
POUR TRAIT
AV 80 DR 90
FIN
POUR PLACE
LC AV 80 DR 45 BC
FIN
POUR TOIT
REPETE 2 [PENTE]
FIN
POUR PENTE
AV 56 (1) DR 90
FIN
  
```

Munis d'une méthode, les élèves ont choisi librement leurs projets. Notre démarche pédagogique consiste à partir des problèmes particuliers que les enfants rencontrent pour les amener à réfléchir sur des problèmes plus généraux. Voici quelques exemples : polygones réguliers, symétrie, rotation, théorème de Pythagore, récursivité (cf. brochure).

Voici juste un exemple de projet qui a permis d'introduire le théorème de Pythagore et la notion de racine carrée. Il nous paraît intéressant dans le fait qu'il révèle comment le micro-ordinateur peut motiver un travail mathématique classique.



3. 2. 2. La résolution d'un problème :

Comme pour la partie graphique, nous avons dirigé l'activité des enfants afin d'essayer de leur donner une méthode de recherche pour la résolution de problèmes. Il s'agit d'écrire le programme fil permettant de résoudre le problème suivant : un paysan veut entourer un pré d'une rangée de fil de fer. Le pré est rectangulaire. On connaît la longueur, la largeur et le prix du mètre de fil de fer. Quelle sera la dépense ?

A) La première étape consiste à établir un lexique où l'élève doit mettre en évidence, d'une part les données (il fixe déjà le nom des variables utilisées) et d'autre part le résultat recherché.

Je vais t'aider à résoudre ton problème en te posant des questions :

- Combien y a-t-il de données numériques dans le texte du problème ?
- 3.
- Tape une donnée numérique du texte du problème.
- 5.
- Comment veux-tu appeler le tiroir dans lequel tu ranges ce nombre ?
- Prix.
- Veux-tu choisir une unité pour cette donnée ? (0- N)
- 0.
- Donne l'unité de prix.
- F.

Lexique		
Nom du tiroir	Contenu	Uni
Prix	5	F
Long	8	M
Largeur	4	M

B) La recherche emprunte des éléments à la méthode de programmation descendante. On part de la question demandée : ici DEPENSE.

Comment l'obtient-on ?

$$DEPENSE =: PRIX * PERI$$

PRIX est une donnée, comment obtenir PERI ? etc... jusqu'à ce qu'on remonte aux données du problème.

A chaque opération correspond une procédure. On utilise largement les couleurs pour différencier dans le tableau RECHERCHE :

- données (exemple : PRIX),
- variables intermédiaires (exemple : DPRI),
- procédures (exemple : MULTIPLICATION).

175

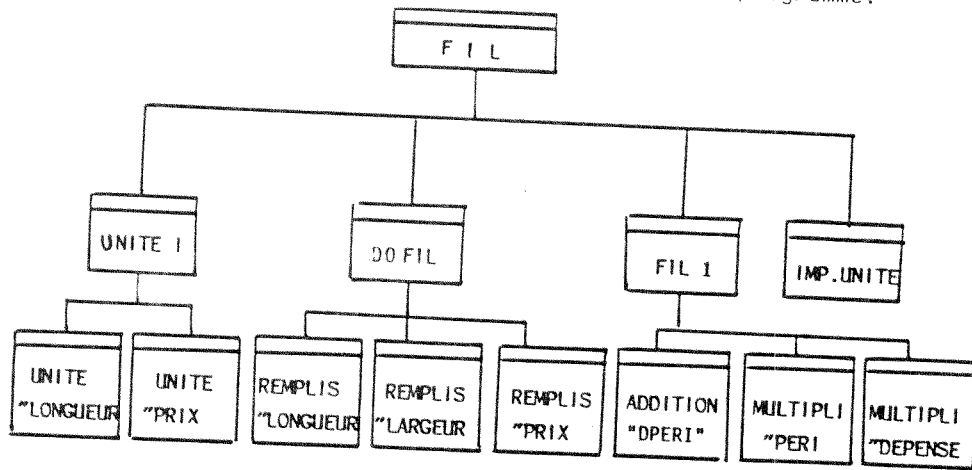
Le problème des unités est à ce niveau primordial. Souvent la première question d'un enfant face à un énoncé de problème est : Quelle est l'unité ? Pour aider l'enfant, nous avons mis à sa disposition une procédure UNITE qui lui permet de traiter le problème des unités "à part" : la procédure UNITE crée pour chaque variable (par exemple ici LONGUEUR) un tiroir de nom UNI (nom du tiroir ici UNITE LONGUEUR) où est placée l'unité tapée par l'enfant (ici M). Notez que les unités se retrouvent dans le lexique et qu'elles permettent à l'enfant de visualiser le problème.

RECHERCHE

Remarquons qu'un programme analogue à celui du lexique permet à l'élève d'établir les deux premières colonnes du tableau. Il complète ensuite la colonne des procédures.

Nom du tiroir	Opération	Procédure
DEPENSE	: DEPENSE = : PRIX * : PERI	MULTIPLICATION : PRIX : PERI "DEPENSE
PERI	: PERI = 2 * : DPERI	MULTIPLICATION 2 : DPERI "PERI
DPERI	: DPERI = : LONGUEUR + : LARGEUR	ADDITION : LONGUEUR : LARGEUR "DPERI

c) La dernière phase consiste à écrire l'arbre du programme.



L'écriture du programme est terminée ! Remarquons qu'après la phase de recherche, l'élève a tous les éléments pour rédiger une solution classique du problème. On peut soulever l'hypothèse que cette méthode serait utile à l'élève en dehors de tout contexte de programmation (seule la phase c) nécessite une initiation à la programmation). Cette démarche demande à être expérimentée sur une période durable afin de mieux déterminer l'aide qu'elle peut représenter pour les élèves. Nous abordons cette étude avec la collaboration de l'équipe de didactique des mathématiques de Strasbourg.

3.2.3. Remarques : - la langage LOGO apparaît comme particulièrement adapté à la structuration de la démarche par l'utilisation des procédures qui permettent de décomposer les difficultés,
- l'expérience devrait être poursuivie sur une plus longue durée afin de mieux cerner l'aide qu'elle peut représenter pour les élèves.

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 âgés normalement de 17 à 18 ans.

La section A2 est une section littéraire à dominante philo-sophie 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.

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 justifient même l'activité.

En même temps, le test par ordinateur est un contrôle bien plus efficace que le contrôle du professeur qui semble toujours avec des références arbitraires. La réussite devient objective et la volée 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)

- 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, c
 - . 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 pourrions, 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 :



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 T₁, T₂, T₃, T₄, 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 $\left[\begin{array}{l} T_3 \leftarrow T_2 \\ T_2 \leftarrow T_1 \\ T_1 \leftarrow T_4 \end{array} \right.$

SINON $\left[\begin{array}{l} \text{SI } T_4 \leq T_2 \text{ ALORS } \left[\begin{array}{l} T_3 \leftarrow T_2 \\ T_2 \leftarrow T_4 \end{array} \right. \\ \text{SINON } \left[\begin{array}{l} \text{SI } T_4 \leq T_3 \text{ ALORS } \left[T_3 \leftarrow T_4 \end{array} \right. \end{array} \right.$

Question 1 : on donne les contenus suivants

T ₁	T ₂	T ₃	T ₄
4	7	5	2

et on exécute l'algorithme ci-dessus.

Indiquez, dans un tableau, les contenus de T₁, T₂, T₃ et T₄ 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 \leftarrow 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 $M_2 < M_1$ 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,

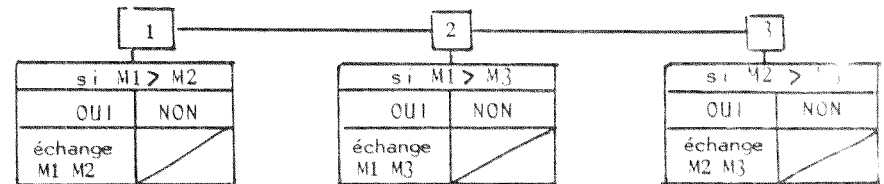
2°) recommencer avec les deux cases restantes M₂ et M₃.

d'où l'algorithme :

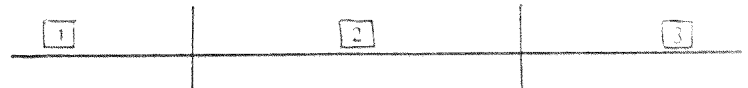
SI	$M_2 < M_1$	alors	ECHANGE (M ₁ , M ₂)
SI	$M_3 < M_1$	alors	ECHANGE (M ₁ , M ₃)
		(* M ₁ contient alors le plus petit nombre)	
SI	$M_2 < M_3$	alors	ECHANGE (M ₂ , M ₃)
		(* M ₂ contient le plus petit des deux restants)	

* 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 :



b) donner un exemple de contenu de M1, M2, M3 tel que l'on ait :



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 :

A N N E X E S

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 (Se 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.

871

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 succes sion d'actions simples, 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>=i) 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 conditions portant sur les objets précédents et déterminer si elles sont vraies ou fausses .

On pourra donner un nom à une liste bien déterminée d'actions simples .

On pourra en demander l'exécution selon qu'une condition est vraie ou fausse :

si condition | alors exécuter liste1 .
 | sinon exécuter liste2

L'exécution d'une liste pourra être répétée tant qu'une condition reste vraie

tant que condition | exécuter liste

exemples :

* SOMME (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 $i \leq N$ exécuter CUMUL
- écrire le nombre N
- écrire la somme S

* CUMUL

- 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

* OCCUR (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 $i \leq l(T)$ exécuter TEST
- écrire N

* TEST

- si $T_{i,1} = "E"$ | alors calculer N+1, ranger dans N
- || sinon rien
- calculer i+1, ranger dans i

IREM - Groupe Informatique et enseignement des mathématiques
Séance du 25 novembre 1981

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, $A \leftarrow AMAX$ 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.

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 leur 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 mathématique 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.

Répondez 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 ?
- 2. Le grand Rabbin de France est-il nommé ou élu, et par qui ?
- 3. Le Bonheur, d'Agnès Varda, fut un film primé. De quand date-t-il ?
- 4. On parle souvent de "puces", à propos des ordinateurs. En quoi est faite une "puce" et quelle quantité d'information peut-elle contenir ?

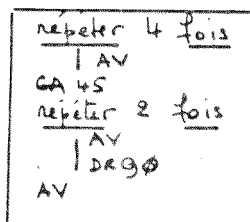
Vous disposez d'un robot qui peut exécuter un certain nombre d'actions élémentaires

- AV avancer d'une unité
- DR 45° pivoter à droite de 45°
- DR 90° pivoter à droite de 90°
- GA 45° pivoter à gauche de 45°
- GA 90° pivoter à gauche de 90°

En outre, ce robot sait reconnaître si la condition BLEU : « le voyant bleu est allumé » est vraie ou fautive

Pour commander le robot, il faut lui indiquer les actions élémentaires à effectuer, dans l'ordre voulu

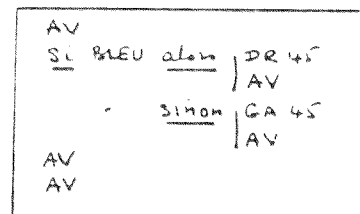
exemple 1



donne :



exemple 2

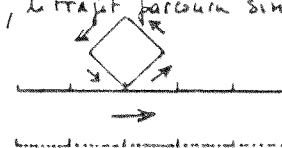


donne :



Voici un trajet défini à être parcouru par le robot :

- en trait plein, le trajet parcouru si le voyant bleu est allumé
- en pointillé, le trajet parcouru sinon (sens des flèches à respecter)



Ecrivez la succession des actions élémentaires à effectuer par le robot pour parcourir ce trajet.

Feuille 1-C | Algorithmes de 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)

Ecrivez, sous forme d'algorithme exécutable par l'un d'entre vous, la confection d'une recette de cuisine que vous connaissez.

181 Annexe 4

③ Dans trois cases mémoires notées M1, M2, M3 sont rangés trois nombres.

L'opération notée "M1 ← M2" signifie maintenant "recopier le contenu de la mémoire M2 dans M1", en lieu et place du contenu actuel de cette dernière, à la manière d'un copier de cassette magnétique.

Exécutez l'algorithme suivant, les contenus initiaux de M1, M2 et M3 étant 2, 4 et -1.

- R ← M2
- M2 ← M1
- M1 ← M3
- M3 ← R

④ Si les contenus initiaux de M1, M2, M3 sont 0, -16, -20, écrivez un algorithme permettant de trouver ces trois nombres dans l'ordre croissant en allant de M1 à M3.

Y a-t-il d'autres contenus initiaux que cet algorithme permet de classer dans l'ordre croissant? Donnez un exemple où l'algorithme fonctionne, un autre où il ne fonctionne pas.

⑤ Quelles valeurs peut-on mettre au départ dans M1, M2, M3 pour que l'algorithme suivant réalise le classement par ordre croissant?

- R ← M1
- M1 ← M2
- M2 ← M3
- M3 ← R

Même question pour l'algorithme :

- R ← M1
- M1 ← M2
- M2 ← R

182

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, noté ←"

p.ex. 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

Illustrer le déroulement de l'algorithme en représentant le contenu successif de 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?

Revenez l'un de ces algorithmes en vous servant de l'opération ECHANGE (x, y), vue au ①, où x, y désignent deux bols quelconques.

②

a) Exécutez l'algorithme ci-contre, sur la mémoire

M1, M2, M3 contenant respectivement -5, 0, -3.

Que contiennent-elles après cette exécution?

b) Même question si les contenus initiaux sont

-3, -5, 0.

$R \leftarrow M_2$
si $M_2 < M_1$ alors $R \leftarrow M_1$
$M_2 \leftarrow M_3$
$M_3 \leftarrow R$

③ On désigne par x, y les contenus inconnus de deux mémoires M1, M2. Écrivez un algorithme permettant de classer ces deux nombres, dans l'ordre croissant, dans les mémoires M1, M2. (à utiliser ← et si ... alors ...)

④ Écrivez l'algorithme correspondant pour trois mémoires M1, M2, M3 contenant trois nombres inconnus x, y, z .

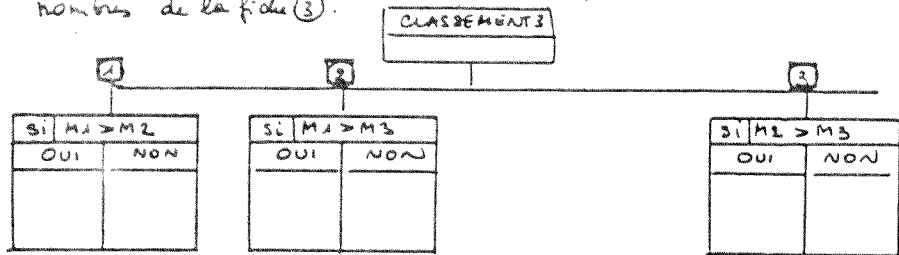
⑤ Décrivez l'algorithme du ④ si les mémoires M1, M2, M3 contiennent au départ 2, 1, 0 en indiquant le contenu initial de chacune de toutes les mémoires utilisées, à chaque étape du déroulement. Combien d'essais analogues faut-il faire pour vérifier que l'algorithme fonctionne bien dans tous les cas?

① On va vous dicter dix nombres.

Pendant cette dictée vous ne notez aucun nombre.Vous ne pouvez garder en mémoire que deux nombres.a) A la fin de la dictée, vous devez donner la moyenne de dix nombres. Comment procédez-vous?b) A la fin de la dictée, vous devez donner le plus petit de dix nombres. Comment faites-vous?

c) Même question pour dix mots en utilisant l'ordre alphabétique.

Le schéma suivant représente l'algorithme de classement de 3 nombres de la fiche ③.



① Complétez les cases oui-non respectives avec les instructions de l'algorithme de classement.

② M_1, M_2, M_3 contiennent, au départ $-2, 4, -5$.

Indiquez, pour chacun des tests ①, ②, ③, le quelle de case oui ou NON est parcourue pour effectuer le classement de ces trois nombres.

③ Même question pour $2, 1, 0$.

④ Quelles étaient les valeurs initiales possibles de M_1, M_2 et M_3 pour que l'algorithme se déroule de la façon suivante :

1 2 3
 OUI NON OUI

⑤ Quelles sont tous les autres déroulements possibles ? Donnez un exemple de contenus pour chaque cas.

① Écrivez l'algorithme permettant de classer 4 nombres, dans l'ordre croissant, dans les mémoires M_1, M_2, M_3 et M_4 , en utilisant la même méthode qu'au ④ de la fiche 3-B.

② exécutez l'algorithme sur les contenus suivants :

M_1 M_2 M_3 M_4
 a) -2 4 -5 -3
 b) 2 1 0 -2

Exercice 1

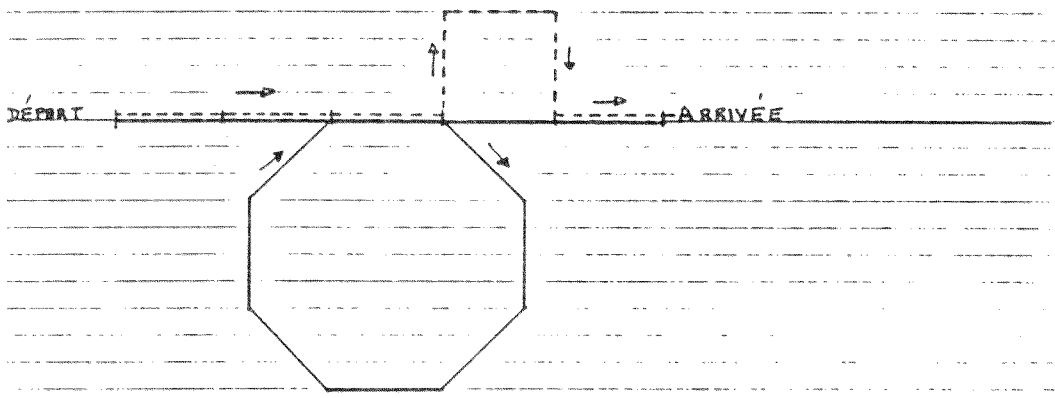
On dispose d'un robot qui peut exécuter les actions élémentaires :

- A.V : avancer d'une unité
- GA 45 : tourner à gauche de 45°
- GA 90 : tourner à gauche de 90°
- DR 45 : tourner à droite de 45°
- DR 90 : tourner à droite de 90°

et qui sait reconnaître si la condition :

① : le feu est bleu est vraie ou fausse.

Ecrivez la succession des ordres élémentaires à donner pour que le robot parcoure le trajet en continu si le feu est bleu et le trajet en pointillé si le feu n'est pas bleu.



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	m

Les nombres

x, y et z vérifient la relation $x < y < z$.

ALGORITHME

SI $T_4 \leq T_1$ ALORS

$T_3 \leftarrow T_2$
$T_2 \leftarrow T_1$
$T_1 \leftarrow T_4$

SINON

SI $T_4 \leq T_2$ ALORS

$T_3 \leftarrow T_2$
$T_2 \leftarrow T_4$

SINON

SI $T_4 \leq T_3$ ALORS $T_3 \leftarrow T_4$
--

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, T_4 à chaque étape de l'exécution.

question 2 Même question dans le cas où au départ les

contenus sont

T_1	T_2	T_3	T_4
-2	-1	6	3

question 3 Plus généralement les contenus au départ sont :

T_1	T_2	T_3	T_4
x	y	z	m

avec $x < y < z$.

- a) si $m \leq x$, quels sont les contenus de T_1, T_2, T_3 à la fin de l'exécution de l'algorithme ?
- b) si vous ignorez la position de m par rapport à x, y et z , que pouvez-vous dire des contenus de T_1, T_2, T_3 à la fin de l'exécution de l'algorithme ?

① Que fait cet algorithme pour $N > 2$. Donnez-lui un nom.

répéter, pour J de 2 à N
 | si $M_1 > M_J$ alors ECHANGE(M_1, M_J)

Faites un essai pour $N=4$ et le contenu

M_1	M_2	M_3	M_4
4	2	0	1

② Utilisez l'algorithme du ① pour réaliser l'algorithme de classement de 4 nombres vu dans la fiche 4.

③ Donnez l'algorithme de classement de N nombres dans les N cases mémoires M_1, M_2, \dots, M_N .

```

0010 *CLASSEMENT N
0015 LIRE[/, 'DONNER N ' ] N
0020 TABLEAU CHAINE M[N], CHAINE R
0030 FAIRE 40 POUR J+1 JUSQUA N
0040 LIRE[/, 'DONNER UN ELEMENT: ' ] M[J]
0050 FAIRE 60 POUR J+1 JUSQUA N-1
0060           &MINIM(J, N)
0070 FAIRE 70 POUR J+1 JUSQUA N; AFFICHER M[J]
0080 TERMINER
0090 PROCEDURE &MINIM(A, B)
0100   FAIRE 110 POUR I=A+1 JUSQUA B
0110           SI M[A] > M[I] ALORS &ECHAN(M[A], M[I])
0120 RETOUR
0130 PROCEDURE   &ECHAN(C, D)
0140   R=M[C]
0150   M[C]=M[D]
0160   M[D]=R
0170 RETOUR
  
```

I Dérouler l'algorithme suivant:

- si $M_1 < M_2$ alors échange M_1, M_2
- si $M_1 < M_3$ alors échange M_1, M_3
- si $M_1 < M_4$ alors échange M_1, M_4
- si $M_2 < M_3$ alors échange M_2, M_3
- si $M_2 < M_4$ alors échange M_2, M_4
- si $M_3 < M_4$ alors échange M_3, M_4

M_1	M_2	M_3	M_4
2	4	3	5

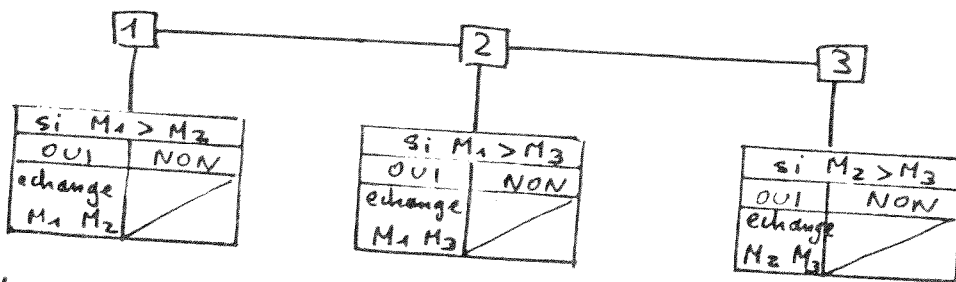
Que fait cet algorithme ?

II Ecrire un algorithme faisant le même travail sur 3 mémoires M_1, M_2, M_3

Vérifier sur

M_1	M_2	M_3
2	0	4

III Soit l'arbre suivant:



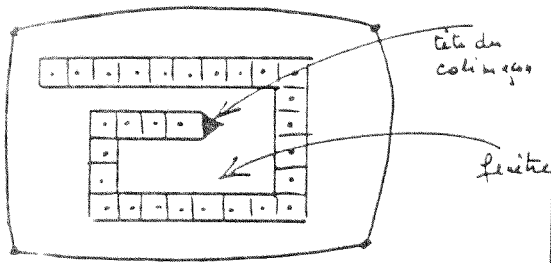
a) M_1, M_2, M_3 contiennent respectivement $-2, 4, -5$. Indiquez pour chaque test si l'on parcourt la branche OUI ou NON

1	2	3

b) Donner un exemple de contenu de M_1, M_2, M_3 tel que l'on ait:

1	2	3
OUI	NON	OUI

Fiche 7.B. Le jeu du colimaçon (jeu complet)



Essai de "colimaçon"



(vu en verticale !!)

Il s'agit d'écrire un algorithme permettant l'allumage automatique de cases consécutives dans une direction donnée, jusqu'à ce que le joueur frappe une touche quelconque du clavier, ce qui a pour effet de modifier la direction du colimaçon, selon l'ordre suivant :

\rightarrow \downarrow \leftarrow \uparrow (cf schéma ci-dessous). Le jeu s'arrête si

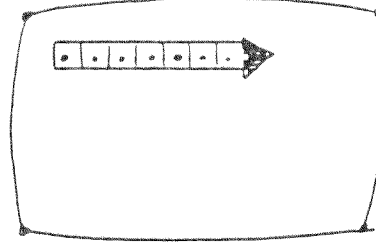
la "tête" du colimaçon entre en collision avec une case déjà allumée.

Le but du jeu consistera donc à deviner "en extrême" la trajectoire en frappant une touche avant la collision.

• Écrivez l'algorithme complet du jeu.

(on pourra désigner par x_{min} , x_{max} , y_{min} , y_{max} les limites de la "fenêtre" dans laquelle le colimaçon peut encore se développer à un instant donné.)

Fiche 7A. LE JEU DU COLIMAÇON (Preliminaires)



Vous disposez d'un écran et d'un clavier ainsi que de codes suivants :

WIT : effacer l'écran

COURSE(x,y) : allumer la case d'abscisse x et d'ordonnée y

CLAVIER : vérifie si une touche du clavier est enfoncée. Si oui, la case mémoire REPOSE contient le caractère fourni par cette touche. Sinon, la case mémoire REPOSE n'est pas modifiée.

1. Écrivez un algorithme permettant l'allumage automatique d'un segment horizontal, à partir de la case supérieure gauche de l'écran, et ce, jusqu'à ce que le joueur frappe une touche.
2. Prévoir, en outre, que l'allumage s'arrête si le bord droit de l'écran est atteint, même si le joueur n'intervient pas ($x = x_{max}$).
3. Faire en sorte que l'allumage se poursuive en descendant, à partir du moment où le joueur a frappé sa touche. L'allumage devant toujours s'arrêter, par contre, si le bord de l'écran est atteint sans que le joueur ne soit intervenu.

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 any, let alone every 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 school (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

189

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 ($\frac{1}{2}$ one year) and curriculum change ($\frac{1}{2}$ 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.

190
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 than 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's showed 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 at least 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 autonomous 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

191

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.

192
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 uncharted 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

143

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 people, 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.

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).

Hugh Burkhardt "The Microcomputer: Miracle or Menace in Mathematical Education" in Proceedings of ICME5, Ed Marjorie Carss (Birkhauser, 1985).

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 calcuations 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 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.

197

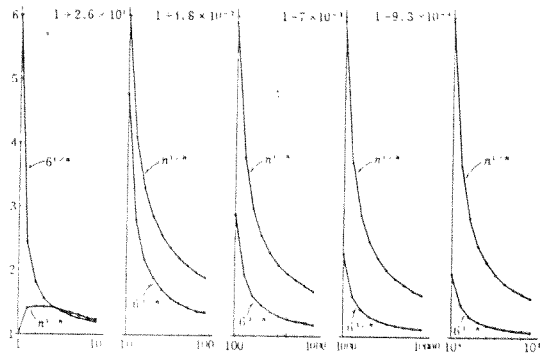
The main features of sequences and series to be discussed are:

- a. Order of growth, rapidity of convergence
- b. Behaviour of sequences. Monotonicity, 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, decreasingly (b_n for $n \geq 3$). 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

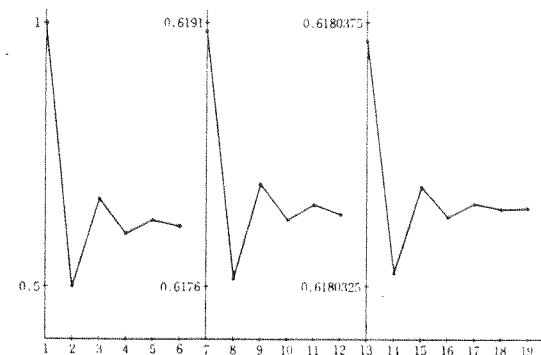
$$a_n = \frac{1}{\sqrt{5}} \left\{ \left(\frac{1 + \sqrt{5}}{2} \right)^n - \left(\frac{1 - \sqrt{5}}{2} \right)^n \right\}$$

So the order of growth is about $\frac{1 + \sqrt{5}}{2} \approx 1.6$. 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 derised).

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. $a_n = \left(1 + \frac{1}{n}\right)^n$

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 hand-held calculators. Observe that different apparatus yield different values, and the difference is sometimes remarkable.

198

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 $\frac{1}{n}$ 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 expansion of

$$\left(1 + \frac{1}{n}\right)^n = \exp \left[n \log \left(1 + \frac{1}{n} \right) \right]$$

We can see from the form of this expansion that the acceleration

method can be applied to get a better answer.

Put

$$c_n = 2 a_{2n} - a_n$$

$$d_n = \frac{1}{3} (4 c_{2n} - c_n)$$

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

$$1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \dots$$

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 convergence is very slow. We here give a result of computation. The values are multiplied by 4 to get an easier comparison with π .

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
2.716923932	2.717602569	2.717828868	2.717942121	2.718010050	2.718054253	2.718086876	2.718111955	2.718130556
2.716923932	2.717602569	2.717828893	2.717942121	2.718010050	2.718055231	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	2.718254546
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	2.718173099	2.718281828	2.718281828	2.718173099	2.718336195	2.718281828	2.718010014
2.718280469	2.718281149	2.718003561	2.718281489	2.718281557	2.717194507	2.717194539	2.718281659	2.718009863
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 $\frac{1}{2n} e$.

199

1	4.0000000000				
2	2.6666666667	20	3.0916239067	200	3.1365926848
3	3.4666666667	30	3.1082695667	300	3.1392592295
4	2.8952380952	40	3.1165945568	400	3.1390924575
5	3.3396825397	50	3.1215946526	500	3.1395926556
6	2.9760461760	60	3.1249271439	600	3.1399259881
7	3.2837384837	70	3.1273076680	700	3.1401640829
8	3.0170718171	80	3.1290931418	800	3.1403426541
9	3.2523659347	90	3.1304818854	900	3.1404815428
10	3.0418396189	100	3.1315929036	1000	3.1405926538
				10000	3.1414926536

The remarkable phenomena seen from this table can be explained by

$$\left| a_n + \frac{1}{4n} - \frac{\pi}{4} \right| < \frac{1}{8n^3} \quad \text{for } n: \text{ even}$$

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

$$1 + \frac{1}{2} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots$$

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

$$b_n = 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n} - \log n,$$

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.

$$b_n = \gamma + \frac{1}{2n} - \frac{B_1}{2n^2} + \frac{B_2}{4n^4} - \frac{B_5}{6n^6} + \dots$$

200

But the way to derive this formula, and the meaning of asymptotic expansions 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

$$a_n - \gamma = b_n + b_{n+1} + \dots$$

Now,

$$b_n = \log \frac{n+1}{n} - \frac{1}{n+1}$$

$$= \int_0^1 \frac{1}{n+1-t} dt - \frac{1}{n+1}$$

$$= \int_0^1 \frac{t}{(n+1)(n+1-t)} dt$$

$$= \frac{1}{2} \frac{1}{n(n+1)} - \frac{1}{2} \int_0^1 \frac{t^2}{(n+1)(n+1-t)^2} dt$$

$$\int_0^1 \frac{t^2}{(n+1)(n+1-t)^2} dt$$

$$= \int_0^1 \frac{t^2}{(n+1-t)^2} dt - \int_0^1 \frac{t^3}{(n+1-t)^2} dt + \int_0^1 \frac{t^4}{(n+1)(n+1-t)^2} dt$$

$$= \left(\frac{1}{3} - \frac{1}{4} \right) \frac{1}{n^2} - \int_0^1 \frac{t^3}{(n+1-t)^2} dt + \int_0^1 \frac{t^4}{(n+1)(n+1-t)^2} dt + O\left(\frac{1}{n^5}\right)$$

$$= \frac{1}{12} \frac{1}{n^2} - \left(\frac{1}{4} - \frac{1}{5} \right) \frac{1}{n^3} + O\left(\frac{1}{n^4}\right)$$

$$= \frac{1}{12} \frac{1}{n^2} - \frac{1}{20} \frac{1}{n^3} + O\left(\frac{1}{n^4}\right)$$

So, finally we see

$$a_n - \gamma = \frac{1}{2n} - \frac{1}{12} \frac{1}{n^2} + O\left(\frac{1}{n^4}\right)$$

which may be applicable when $\frac{1}{n^4}$ is negligible.

One observes also from this expression that acceleration method

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	$\tan x$	$\tan^2 x$	$\tan^3 x$	$\tan^4 x$	$\tan^5 x$	$\tan^6 x$	$\tan^7 x$	$\tan^8 x$	$\tan^9 x$	
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 $\sin x$, $\cos x$ are taught, but not of $\tan x$. Then $\tan x$ cannot have these derivatives, nor it has not 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.

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.

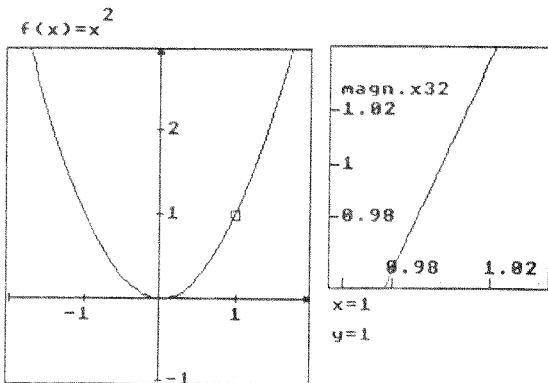
This paper reports an approach to the learning of the calculus [1] using programs to promote understanding of the fundamental concepts. The first four sections contain descriptions of the programs and the rationale underlying their design. Section 5 describes some of the reactions encountered in schools 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

$$\frac{f(x+h)-f(x)}{h} \quad (1)$$

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.

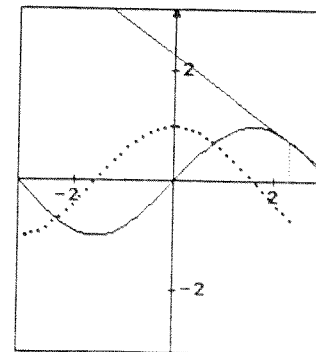


To represent the changing gradient of a graph, it is a simple matter to calculate the expression (1) for fixed h as x varies. The program

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 $\sin x$ being built up. It clearly approximates to $\cos x$; by superimposing the graph of $\cos x$ for comparison the gradient function may be investigated experimentally before any of the traditional formalities are introduced.

$f(x)=\sin x$
from $x=-\pi$ to π



gradient function
 $(f(x+c)-f(x))/c$
for
 $c=1/1800$

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 $x=\pm 1$ and $|\sin x|$ at multiples of π .

It is possible to draw a function everywhere continuous and nowhere

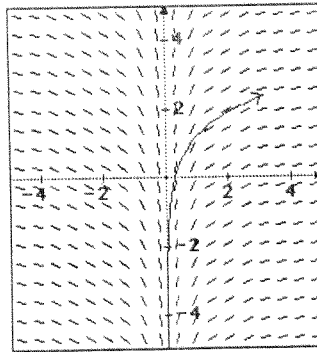
differentiable (a concept previously beyond the reach of elementary calculus): its graph is so wrinkled that it never magnifies to look straight. The program BLANCMANGE draws such a graph and allows investigation of its properties.

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.

$$f(x)=1/x$$

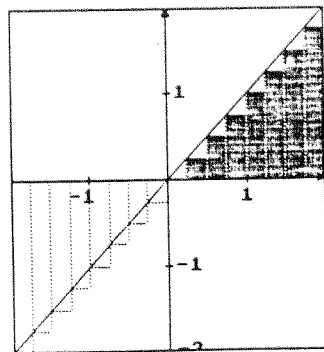


The program AREA calculates the area between the graph and the x -axis by a variety of methods. The numerical values are displayed and each part of the area is drawn using different colours for positive and negative results. Students can see that a positive step gives a positive result when the graph 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

$$f(x)=x$$

from $x=-2$ to 2



Area $A(x)$
from
 $a=-2$
to
 $b=2$
step
 $c=1/4$
First ordinate
 -8.5888

difficult yet clearly represented by moving graphics.

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

$$A(x+h)-A(x) \approx f(x)h \text{ for small } h,$$

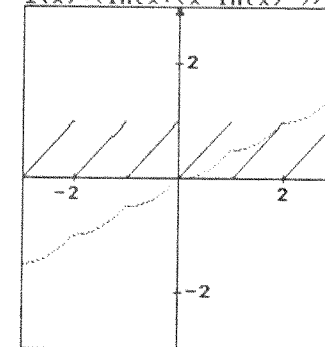
with the approximation getting better as h tends to zero.

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 $f(x)=x-\text{INT}x$ 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.

$$f(x)=x-\text{INT}x$$

$$I(x)=(\text{INT}x+(x-\text{INT}x)^2)/2$$



Area $A(x)$
from
 $a=0$
to
 $b=3$
step
 $c=1/29$
Mid ordinate

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 to 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

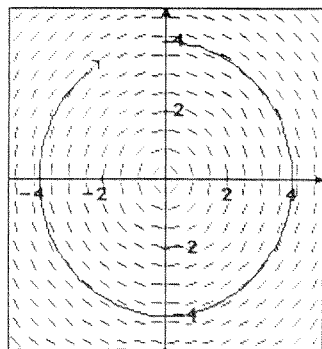
$$y \frac{dy}{dx} = -x$$

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.

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,

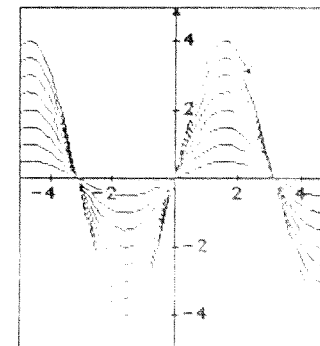
$$v = dy/dx$$

giving two linear equations:

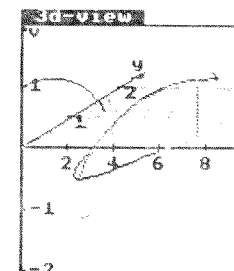
$$\begin{aligned} dy/dx &= v \\ dv/dx &= -x. \end{aligned}$$

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").

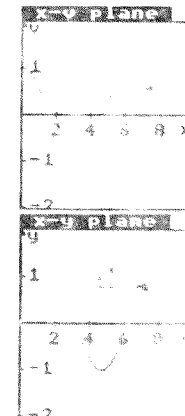
$$d^2y/dx^2 = -y$$



$$\begin{aligned} dy/dx &= v \\ dv/dx &= -y \end{aligned}$$



step=0.1
x=6.7
y=0.41535
v=0.91058



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 $n=-1,-2,1/2, \pi$ 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 $\sin x$ and $\cos x$ 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 $\ln(\text{abs}(x))$ and $\tan x$ 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 Cockcroft 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 be introduced 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 continuous 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.

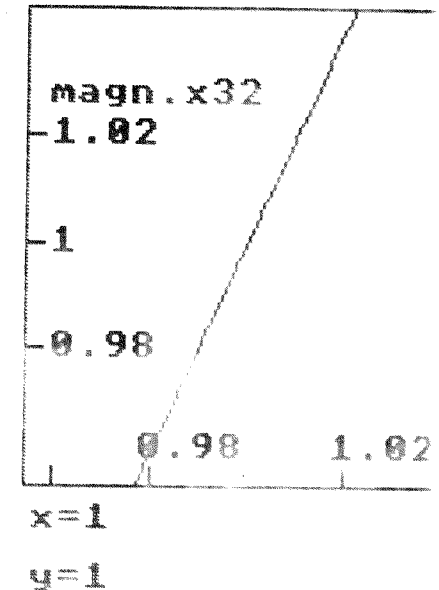
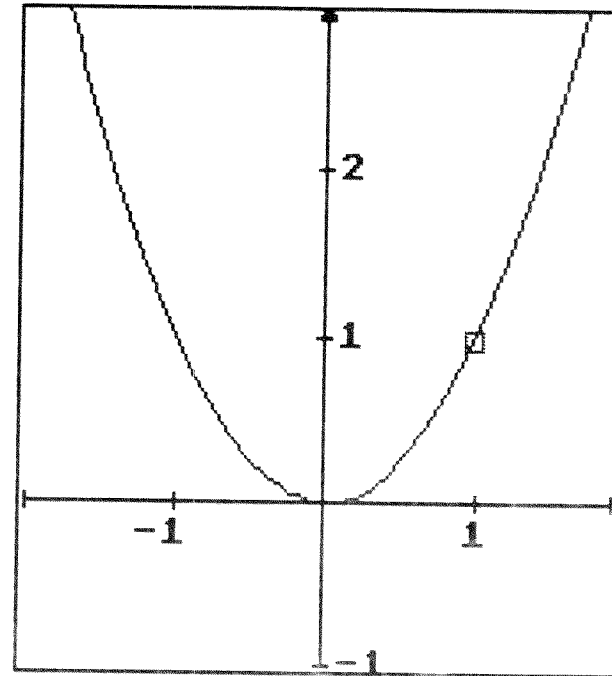
10. Teachers of mathematics do not need to study the whole of computing. They need to be trained to cope with those parts of computing relevant to mathematics teaching, including mathematical aspects of programming (particularly in mathematical algorithms) and the use of appropriate software in the development 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

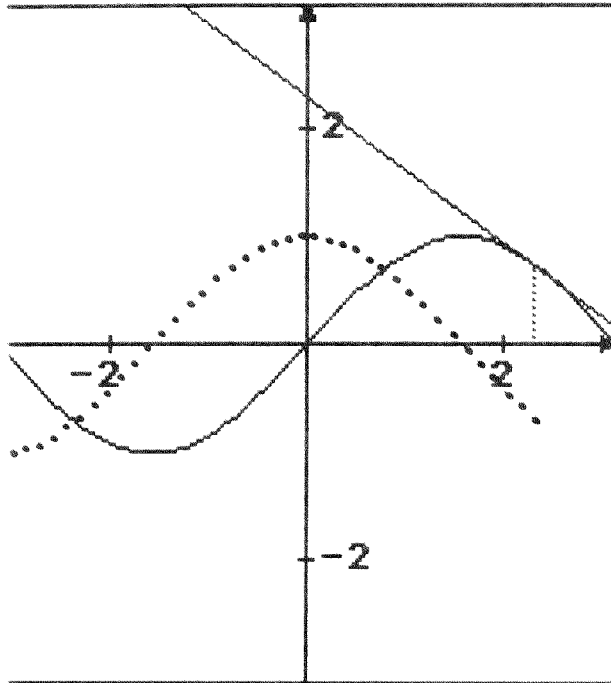
1. David Tall *Graphic Calculus for the BBC computer* Shiva Publications 1985
2. *The influence of Computers and Informatics on Mathematics and its Teaching*. An ICMI discussion document 1984
3. W. H. Cockcroft *Mathematics Counts* H.M.S.O. 1982

$$f(x) = x^2$$



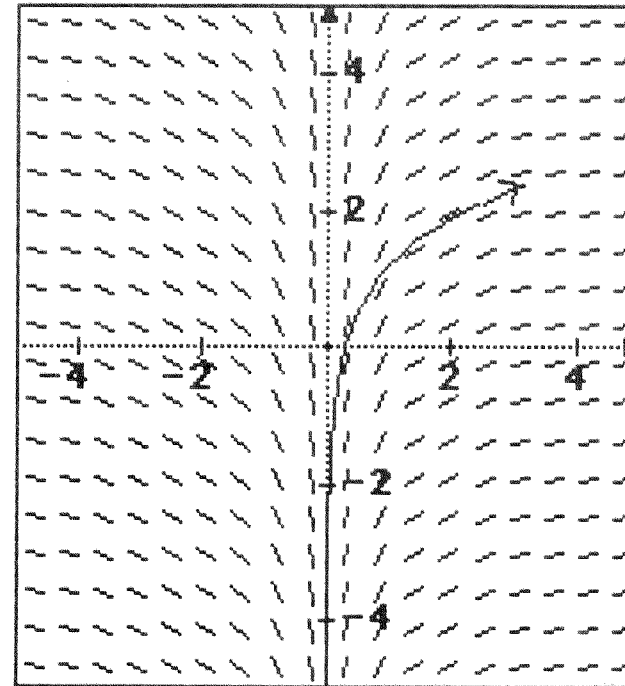
$$f(x) = \sin x$$

From $x = -\pi$ to π



gradient function
 $(f(x+c) - f(x))/c$
for
 $c = 1/1000$

$$f(x) = 1/x$$



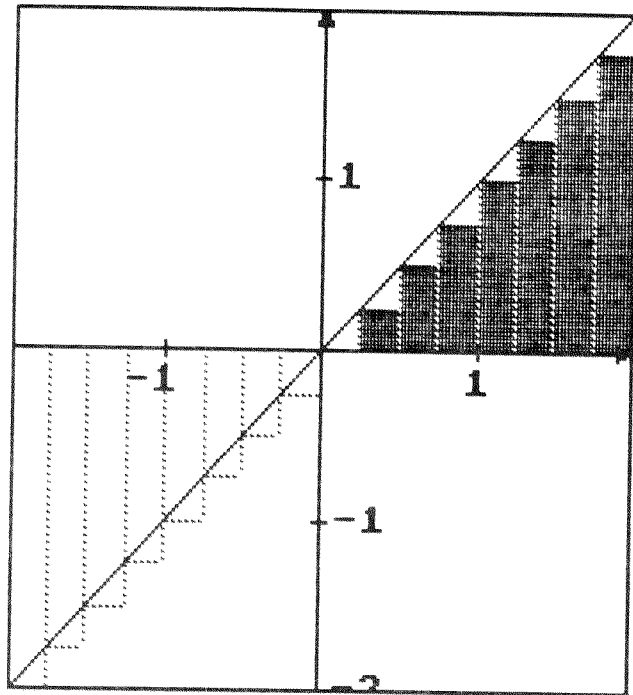
line segments
gradient $f(x)$

curve of
gradient $f(x)$

arrow at
 $x = 2.44613$
 $y = 2.10205$

$f(x)=x$

from $x=-2$ to 2

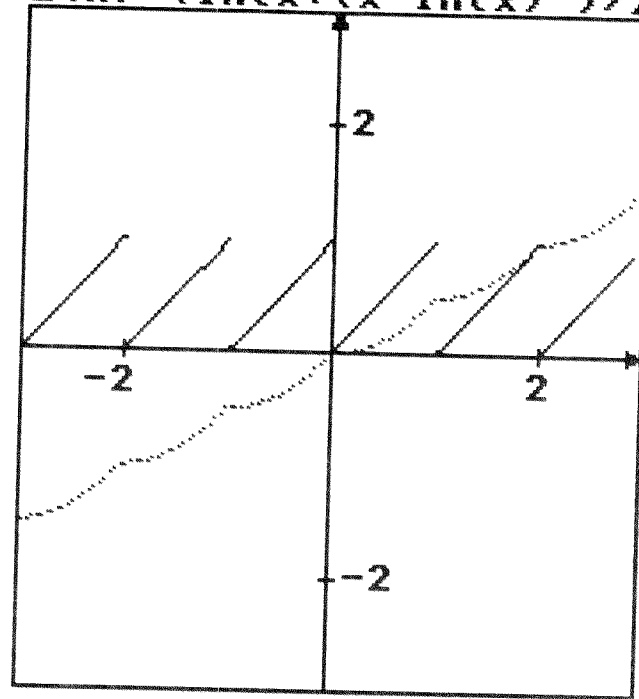


Area $R(x)$
 from
 $a=-2$
 to
 $b=2$
 step
 $c=1/4$

First ordinate
 -0.5000

$f(x)=x-intx$

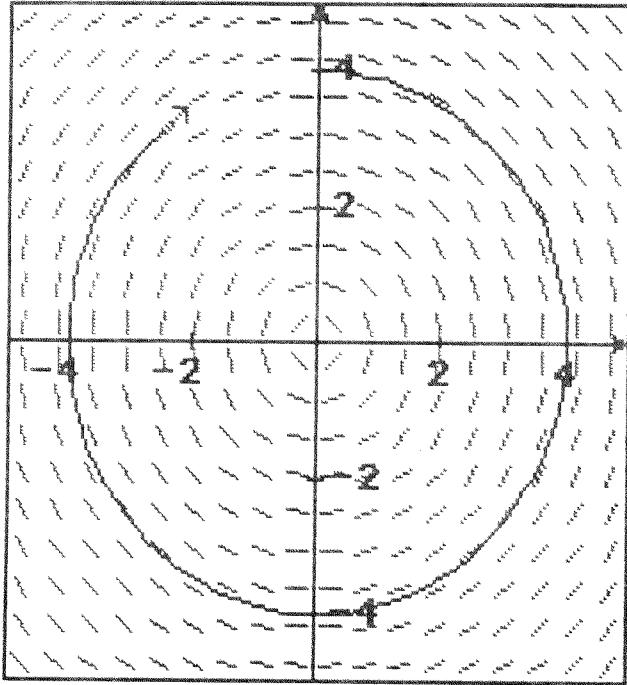
$I(x)=(intx+(x-intx)^2)/2$



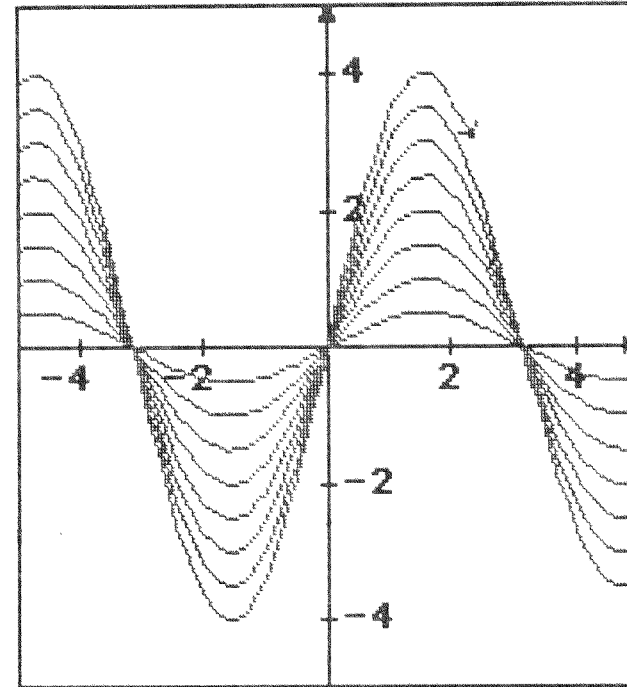
Area $R(x)$
 from
 $a=0$
 to
 $b=3$
 step
 $c=1/20$

Mid ordinate

$$dy/dx = -x/y$$

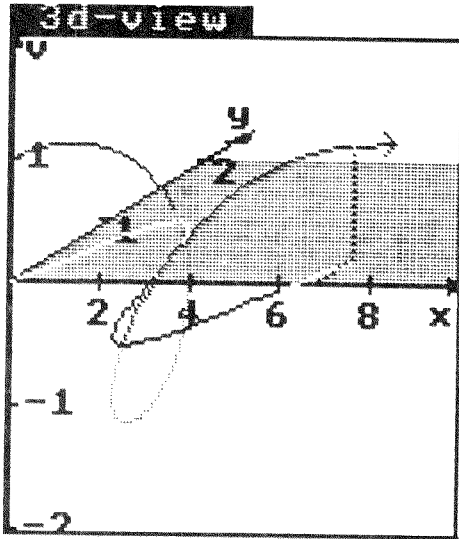


$$d^2y/dx^2 = -y$$

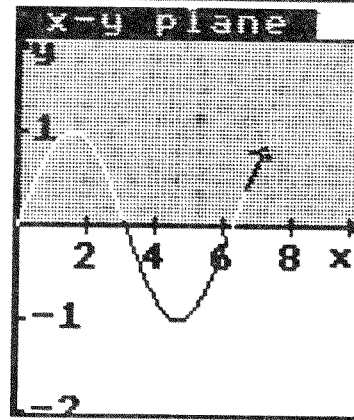
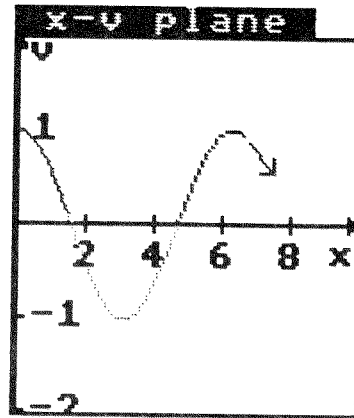


$$dy/dx=v$$

$$dv/dx=-y$$



step=0.1
x=6.7
y=0.41535
v=0.91858



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) On omitting certain calculational aspects from 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? We can give only a few examples in this presentation.

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 simple 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 notwithstanding.

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 an historical 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. We now briefly explore each of these proposals.

(i) An historical approach

Let us show them some of the power of mathematics and the power of the human mind. An historical approach in teaching the calculus can expose the problems that great minds wrestled with and finally solved. 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. In addition, the importance to the student of this approach 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.

(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 indispensable 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.

B.R. Hodgson
Département de mathématiques
Université Laval
Québec, G1K 7P4

E.R. Muller
Department of Mathematics
Brock University
St. Catharines, L2S 3A1

J.C. Poland
Department of Mathematics
and Statistics
Carleton University
Ottawa, Ontario, K1S 5B6

P.D. Taylor
Department of Mathematics
and Statistics
Queen's University
Kingston, K7L 3N6

Bibliography

- 1] D. Anderson, "Arithmetic in the computer/calculator age", in The Future of College Mathematics, ed. A. Ralston and G.S. Young, Springer-Verlag, 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. Rayner, "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.

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/

/2/ cf. /Winkelmann 1982/

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. Richtenhagen /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 Richtenhagen). 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 diminution 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 observations, 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

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 admissibility 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 discussed 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 possibilities 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 examples.

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: $X = X + V \cdot H$
 $V = V + G \cdot H$ ($G = 9.81$)
 $T = T + H$

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 assess 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 (Apple 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 subdivided into n steps ($T=n*H$). Simulating the computer run with general numbers for small values of n then leads to the following formulas:

$$\begin{aligned}
 V_{T=n*H} &= n * H * G \\
 X_{T=n*H} &= 0 + H*H*G + 2*H*H*G + \dots + (n-1)*H*H*G \\
 &= (1 + 2 + \dots + (n-1)) * (H*H*G) \\
 &= \frac{(n-1)*(n-2)}{2} * H*H*G \\
 &= \frac{n^2 - 3n + 2}{2} * H^2 * G \quad \leftarrow T = n * H \\
 &= \frac{G}{2} T^2 + \frac{T^2 (-3n + 2)}{2 n^2} G \\
 &= \frac{G}{2} T^2 + \frac{1}{n} (\dots)
 \end{aligned}$$

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 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 effect of reducing step width be swamped by more technical ballast. The nesting method sketched 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 $(b - a) * \text{function}(a)$

ELSE $h * \text{function}(b - h) + \text{area approximation}(a, b-h, h)$

END IF

END PROC area approximation;

In this case as well, error assessment is easily possible:

if $|f(x) - f(y)| \leq L * |x - y|$,

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 indispensable 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 computations 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

$$\int_a^1 f'g dx = f(1)g(1) - f(a)g(a) - \int_a^1 fg' dx$$

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:

$$A = 1 + 2 \int_0^1 \frac{\sqrt{x}}{(x+1)^2} dx .$$

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.

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,

/18/ This is based on the LOGO offered for Apple II by Terrapin, version 1.0; cf. /Abelson 1982/.

/19/ For a first general description cf. /Wilf 1982/.

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 expression 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 $A > 0$, 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.

- 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.

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 [4]. 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

$$f(x)=pf(2x), \quad 0 < x < 1/2$$

$$f(x)=p+qf(2x-1), \quad 1/2 \leq x < 1$$

$$f(0)=0, \quad f(1)=1$$

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 :X :P
  FUNCTION F(X,P),
  IF :X=0 OP 0      WHEN X=0, 0 EXIT,
  IF :X=1 OP 1      WHEN X=1, 1 EXIT,
  IF :X <.5 OP :P*F 2*:X :P  WHEN X < 1/2, P*F(2*X,P) EXIT,
  OP :P+(1-:P)*F 2*:X-1 :P  P+(1-P)*F(2*X-1,P),
END
ENDFUN;
```

Fig.1

Fig.2

p	0.4	0.3	0.25	0.1	$5 \cdot 10^{-2}$	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	0.504433

Table 1

Table 1 shows a few examples. For $p \leq 1/2$ bold gamble is an optimal strategy. For $p \geq 1/2$ 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

$$0.5 \uparrow 127 = 5.87747176E-39 \text{ (correct to 9 significant digits).}$$

But we get abruptly $0.5 \uparrow 128 = 0$.

To prevent underflow we use logarithms. The following program BIN is extremely reliable and gives for input n,p,c,d the output

$$s=b(c)+b(c+1)+\dots+b(d).$$

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
40 L=L+R+LOG((N-X+1)/X) ← computes ln b(c)
50 NEXT X
60 S=EXP(L):IF D=0 THEN 110 ← computes b(c)
70 FOR X=C+1 TO D
80 L=L+R+LOG((N-X+1)/X) ← computes s=b(c)+b(c+1)+...+b(d)
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.

Vitamin C	common cold		sum
	yes	no	
yes	302	105	407
no	335	76	411
sum	637	181	818

x	407-x	407
637-x	x-226	411
637	181	818

226+x	181-x	407
411-x	x	411
637	181	818

Table 2. Source: Canad. Med. Ass. J., Sept. 1972, 503-508.

Table 3.

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

$$P = P(x \leq 302 | H) = \sum_{x=216}^{302} \frac{\binom{407}{x} \binom{411}{637-x}}{\binom{818}{637}}$$

If we use Table 4 instead of Table 3, we get

$$P = P(x \leq 76 | H) = \sum_{x=0}^{76} h(x) = \sum_{x=0}^{76} \frac{\binom{411}{x} \binom{407}{181-x}}{\binom{818}{181}}$$

234

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
30 L=L+LOG((N-R-I+1)/(N-I+1)) ← computes ln h(0)
40 NEXT I
50 P=EXP(L):IF X=0 THEN 95 ← computes h(0)
60 FOR I=1 TO X
70 L=L+LOG(((R-I+1)*(S-I+1))/(I*(N-R-S+I))) ← computes
80 P=P*EXP(L)                                     p=h(0)+h(1)+...+h(x)
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.

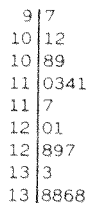


Fig.6

We get for P an estimate

$$P(RS \leq 42) \approx 5.9675\%$$

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 numbers 0 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 $0,1,\dots,u$ 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 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

$$w(0,m,n)=w(u,0,n)=w(u,m,0)=1, w(u,m,n)=0 \text{ for } u < 0.$$

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 computation of the same number $w(14,8,7)$ takes somewhat less time.

235

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 than 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,O,Y)=1
50   NEXT Y
60 NEXT Z
70 FOR Z=0 TO U
80   FOR X=0 TO M
90     W(Z,X,O)=1
100  NEXT X
110 NEXT Z
120 FOR X=0 TO M
130   FOR Y=0 TO N
140     W(O,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. Wilcoxon's signed rank test

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 6 shows the result.

Pair	1	2	3	4	5	6	7	8	9	10	11	12	13	14
IQ of heavier twin: X	97	79	100	100	100	124	95	80	91	108	91	90	104	119
IQ of lighter twin: Y	97	70	100	106	85	123	84	70	84	106	97	90	92	104
D=X-Y	0	9	-1	-6	15	1	11	10	7	2	-6	0	12	15

Table 6. Source:Churchill and Willerman: Intelligence and birth weight in identical twins, Child Development, vol.38, No.3, pp623-629.

a) Let us now proceed with a simulation. We strip down the data by subtracting from each column the same number so that the minimum becomes zero.

heavier twin	0	9	0	0	15	1	11	10	7	2	0	0	12	15
lighter twin	0	0	1	6	0	0	0	0	0	0	6	0	0	0
$ D =d_k$	0	9	1	6	15	1	11	10	7	2	6	0	12	15

Table 7.

We use as test statistic the linear combination

$$S = d_1 I_1 + d_2 I_2 + \dots + d_{14} I_{14}$$

236

We compare the value of S with the sum 82 of the first row or the sum 13 of the second row of Table 7. In the simulation program in Fig.12 we do both to get twice as many simulations for roughly the same price.

```

10 INPUT N,L,H:DIM D(N):DEF FNB(X)=INT(RND(1)+0.5)
20 FOR I=1 TO N:READ D(I):NEXT
30 FOR I=1 TO 1000:S=0
40   FOR J=1 TO N:S=S+FNB(1)*D(J):NEXT
50   IF S>=H OR S<=L THEN T=T+1
60 NEXT I
70 PRINT T
80 DATA 0,9,1,6,15,1,11,10,7,2,6,0,12,15

```

Fig.12.

20 simulation runs with n=14, high h=82, and low l=13 gave the T-values:

23,27,27,21,30,30,18,29,26,24,29,22,31,28,18,31,27,22,29,23

with the P-value

$$P \approx 1.2875\%$$

In this case, and even with slightly more extensive data, we could go through all 2^{14} subsets of differences d_k and count those subsets giving a sum $S \geq 82$ or $S \leq 13$. With the program in Fig.13 we get

$$P(S \leq 13) = P(S \geq 82) = \frac{240}{16384} = 1.4648\%$$

Our simulation results converge versus this exact value. All later P-values should be measured against this exact value.

Unfortunately, the computational effort for this so called permutation test grows exponentially with the number of data. So we have to resort to simulation in general.

```

10 INPUT N:DIM C(N),D(N):T=1:MAX=2↑N-1
20 FOR I=1 TO N:READ D(I):NEXT
30 FOR M=1 TO MAX:I=1:B=M
40   B=B/2:IF B=INT(B) THEN J=J+1:GOTO 40
50   C(J)=1-C(J):D=2*C(J)-1:S=S+D(J)*D:IF S <=13 THEN T=T+1
60 NEXT M
70 PRINT T
80 DATA 0,9,1,6,15,1,11,10,7,2,6,0,12,15

```

Fig.13.

In the program in Fig.13 the Cray code $C(1), \dots, C(14)$ is used for subset generation. Thus one subset element changes its state each time, so that the subset sum S is easy to update. The only nontrivial line 40 computes the position J of the Cray code digit $C(J)$ to be changed in going from one subset to the next.

b) We have already found the exact P-value, but only because we had just 14 differences d_k . By going from the d_k 's to their ranks we can beat the curse of exponential growth of the computation time. In addition we have to make two more minor compromises. We drop the two d_k which are equal to zero. There are several ties among the d_k . It is customary and reasonable to give them average ranks. Table 8 gives for our example D, |D| and $\text{rank}(|D|)$.

D	-1	1	2	-6	-6	7	9	10	11	12	15	15
D	1	1	2	6	6	7	9	10	11	12	15	15
$\text{rank}(D)$	1.5	1.5	3	4.5	4.5	6	7	8	9	10	11.5	11.5

Table 8.

The sum of the negative ranks is

$$T^- = 1.5 + 4.5 + 4.5 = 10.5$$

We want to find

$$P(T^- \leq 10.5) = P(T^+ \geq 67.5)$$

Under H both $X-Y$ and $Y-X$ have the same distribution. That is, by coin tossing we must select a random subset of $\{1, 2, \dots, 12\}$ and find its sum T^- . Favorable cases are those with $T^- \leq 10.5$. Possible cases are all 2^{12} subsets. In other words the number of favorable cases are the number of representations of 0, 1, 2, ..., 10.5 in the form

$$1 \cdot I_1 + 2I_2 + \dots + 12 \cdot I_{12}$$

with I_k equal to 0 or 1. Now

$$\begin{aligned}
10 &= 9+1=8+2=7+3=6+4=7+2+1=6+3+1=5+4+1=5+3+2=4+3+2+1, \\
9 &= 8+1=7+2=6+3=5+4=6+2+1=5+3+1=4+3+2, \quad 8=7+1=6+2=5+3=5+2+1=4+3+1, \quad 7=6+1=5+2=4+3=4+2+1, \\
6 &= 5+1=4+2=3+2+1, \quad 5=4+1=3+2, \quad 4=3+1, \quad 3=2+1, \quad 2, \quad 1, \quad 0.
\end{aligned}$$

These are 10+8+6+5+4+3+2+2+1+1+1=43 subsets with $T^- \leq 10$. But we have $T^- \leq 10.5$. So we make another compromise and add 6 of the 12 subsets with $T^- = 11$:

$$11 = 10+1 = 9+2 = 8+3 = 7+4 = 6+5 = 8+2+1 = 7+3+1 = 6+4+1 = 6+3+2 = 5+4+2 = 5+3+2+1.$$

Thus we get 49 favorable cases altogether. So

$$P(T^- \leq 10.5) = \frac{49}{4096} = 1.2\%$$

Finally we want to find a computer algorithm which finds

$$q(t, n) = \text{number of favorable cases for the event } T^- \leq t = \# \text{ of representations of the numbers } 0, 1, \dots, t \text{ in the form } 1 \cdot I_1 + 2 \cdot I_2 + \dots + n \cdot I_n \text{ with } I_k = 0 \text{ or } 1.$$

We can easily derive a recursion for q. A representation either contains n or it does not. If it contains n, we must partition the remaining number $t-n$ into at most $n-1$ distinct parts. This can be done in $q(t-n, n-1)$ ways. If it does not contain n, we must partition t into at most $n-1$ distinct parts. This is possible in $q(t, n-1)$ ways. Thus

$$q(t, n) = q(t, n-1) + q(t-n, n-1)$$

with the obvious boundary conditions

$$q(t, n) = 0 \text{ for } t < 0 \text{ and } q(0, n) = q(t, 0) = 1 \text{ for } n \geq 0 \text{ and } t \geq 0.$$

The translation of this recursion into the LOGO language is immediate:

237


```

TO Q :T :N
IF :T<0 OP 0
IF :T=0 OP 1
IF :N=0 OP 1
OP (Q :T-N :N-1)+(Q :T :N-1)
END

```

Fig.14

The inputs Q 10 12 and Q 11 12 yield indeed the results 43 and 55. This LOGO program is quite fast even for moderate values of n and t. For large n,t the BASIC program in Fig.15 can be used. Its space and time complexity is O(t.n).

```

10 INPUT T,N:U=T+1:DIM Q(U,N)
20 FOR I=0 TO N:Q(O,I)=1:NEXT
30 FOR I=0 TO U:Q(I,O)=1:NEXT
40 FOR J=1 TO U
50   FOR I=1 TO N
60     IF J<I THEN Q(J,I)=Q(J,I-1):GOTO 80
70     Q(J,I)=Q(J,I-1)+Q(J-I,I-1)
80   NEXT I
90 NEXT J
95 PRINT Q(T,N)/2↑N, Q(U,N)/2↑N

```

Fig.15

Comment: The permutation test in a) has exponential complexity. By some ingenuity one can often reduce the computation time considerably. In Fig.13 we can easily reduce the computation time by a factor 4 by observing that a subset with sum $S \leq 13$ cannot contain any of the two differences 15. So we can reduce MAX from $2^{14}-1$ to $2^{12}-1$ merely by dropping the two numbers 15 in line 80 and using input 12 instead of 14. We gain another speed-up by the factor 4 by dropping the two zeros.

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	365	386	108	022	330	249	093	111	225	359	019	129
2	159	144	029	272	298	228	150	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	365	013	348	184	283	037	105
9	194	338	317	239	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	271	068	100	346	133	272	015	142	242	072	066	314
13	318	157	259	174	295	069	042	307	175	138	126	163
14	238	004	354	231	178	356	027	291	119	171	131	320
15	017	089	169	273	130	180	322	102	113	204	243	156
16	171	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	325	100	336	075	104	227	311	177	241	203	240
20	102	107	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	376	247	153	339	160	117	009	053
23	118	057	256	252	319	109	172	116	119	201	182	162
24	359	236	258	002	051	358	023	036	195	196	230	095
25	352	179	143	351	361	137	067	286	149	176	132	084
26	092	165	170	340	357	022	303	245	018	007	309	173
27	155	205	264	074	236	084	289	152	233	264	047	078
28	377	299	223	262	108	222	088	167	257	094	281	123
29	149	285	362	191	226	353	270	061	151	229	099	016
30	164		217	208	103	209	787	333	315	038	174	003
31	211		330		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

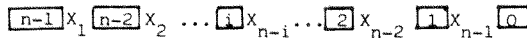
The T-values of 10 simulation runs are 0, 1, 2, 2, 1, 2, 1, 2, 3, 2. This gives for $P(INV \leq 11) = P(INV \geq 55)$ the estimate

$P \approx 0.0008$

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 $p(n,k)$ be the number of n-permutations with at most k inversions. Our aim is to derive a computer friendly recursion formula for $p(n,k)$.

Suppose I have an (n-1)-permutation $X_1 X_2 \dots X_{n-1}$ of $\{1, 2, \dots, n-1\}$. 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



If I place element n into place i, it contributes i inversions. To get at most k inversions altogether the (n-1)-permutation $X_1 X_2 \dots X_{n-1}$ 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

$$p(n,k) = \sum_{i=0}^{k-1} p(n-1,k-i)$$

This recursion is not yet computer friendly. But let us replace k by $k-1$:

$$p(n,k-1) = \sum_{i=0}^{k-1} p(n-1,k-1-i)$$

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

$$p(n,0)=p(1,k)=1 \text{ for } n \geq 1, k \geq 0 \text{ and } p(n,j)=0 \text{ for } j < 0$$

then we get

$$k < n \implies p(n,k)=p(n,k-1)+p(n-1,k)$$

$$k \geq n \implies p(n,k)=p(n,k-1)+p(n-1,k)-p(n-1,k-n)$$

$$p(n,0)=p(1,k)=1$$

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

$$P(\text{INV} \leq 11) = P(\text{INV} \geq 55) = 431886/12! = 0.00090$$

```
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 $\{1,2,\dots,n\}$. 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

$$E(T) = N \left(\frac{1}{N} + \frac{1}{N-1} + \dots + \frac{1}{N-S+1} \right) \approx N \cdot \ln \frac{N}{N-S+1}$$

For $S \ll N/2$ 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 $(\text{Time,Space})=(CS,CS)$, 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 $(\text{Cln } S, CS)$. 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

Report on ISI Round Table Conference on the Impact of Calculators and
Computers on Teaching Statistics in Canberra in August 1984

Lennart Råde
Chalmers University of Technology
Gothenburg, Sweden

ISI has in the past sponsored a number of round table conferences on teaching statistics. These conferences have been organised with a small number of invited participants and observers and each time dealing with a specific topic. The idea has always been to publish the proceedings of the conferences to make the presentations and the discussions available on a world wide basis.

The round table conference in Canberra in Australia took place from August 20-23 1984 just before the Fifth International Conference on Mathematics Education in Adelaide, Australia. The theme of the conference was The impact of calculators and computers on teaching statistics and it was organised by Lennart Råde, Chalmers University of Technology, Gothenburg, Sweden and Terry P. Speed, CSIRO Division of Mathematics and Statistics, Canberra, Australia. The conference was held at the University House of the Australian National University in Canberra.

The following were the invited participants of the conference.

Kazuyoshi Aoki	Chuo University, Tokyo, Japan
Barry Bastow	Education Department of Western Australia, Perth, Australia
Claire Dupuis	Université Louis Pasteur, Strasbourg, France
Arthur Engel	University of Frankfurt, Frankfurt, W.Germany
Joe Gani	University of Kentucky, Lexington, USA
Peter Holmes	Centre for Statistical Education, Sheffield England

James M. Landwehr	AT&Bell Laboratories, Murray Hill, USA
Daniel Lunn	The Open University, Milton Keynes, England
Don McNeil	Macquarie University, North Ryde, Australia
Luke Moortgat	De La Salle University, Manila, Phillipines
Koos Oosthuizen	University of the Orange Free State, Blomfontein, South Africa
Lennart Råde	Chalmers University of Technology, Gothenburg, Sweden
Ken Sharpe	University of Melbourne, Parkville, Australia
Terry P. Speed	CSIRO, Canberra, Australia
Jim Swift	Nanaimo Senior Secondary School, Nanaimo, Canada
Bevan Werry	Department of Education, Wellington, New Zealand

The following observers were present at the conference.

Bob Hall	School of Mathematics and Computer Studies, South Australian Institute of Technology, Adelaide, Australia
Chris Heyde	Australian Mathematical Society, University of Melbourne, Parkville, Australia
Joan Robson	Canberra Mathematical Association, St Clare's College, Manuka, Australia
Chris Stevenson	Australian Bureau of Statistics, Belconnen, Australia
John Taffe	Canberra Mathematical Association, Australian National University, Canberra, Australia

The proceedings of the conference will be published by Chartwell-Bratt Ltd, England, and will appear in 1985. The proceedings will contain the recommendations of the conference, a list of participants and observers, a bibliography

and the following papers with discussions.

K. Aoki, Mastering elementary probability and calculator programming in a class

B. Bastow, The place of computers in the teaching of statistics

D. Dupuis, How calculators and computers change the field of problems in teaching statistics

A. Engel, Statistics and computer science: An integrated high school course

J. Gani, The impact of calculators and computers on teaching statistics

P. Holmes, Using microcomputers to extend and supplement existing material for teaching statistics

J. Landwehr, Using microcomputers for data analysis and simulation experiments in junior and senior high school

D. Lunn, Computer animation: A powerful way of teaching concepts of probability and statistics

D. McNeil, Using microcomputers for teaching introductory statistics - experimental results and implications

L. Moortgat, New and improved skills in computer era

P. Nuesch, Are statistical tables obsolete?

K. Oosthuizen, The microcomputer as an aid to instruction - problems and pitfalls

L. Råde, The pocket computer in the classroom

K. Sharpe, Use of the computer in statistics courses at the University of Melbourne

T. Speed, Teaching statistics at university level: How computers can help us find realistic models.

J. Swift, The Nightingale data library and protocol: A proposal for a library of interesting data

B. Werry, The teaching of statistics in New Zealand Secondary Schools.

The round table conference made a number of recommendations. These are mainly meant to be of help for the ISI Education Committee but should also be of general interest. The recommendations are given in an appendix to this report. The round table also strongly recommend to the ISI Education Committee that the sequence of round table conferences should be continued. As possible themes for future round table conferences the following topics were recommended.

1. Teaching of statistics in developing countries
2. Training teachers to teach statistics in schools
3. Further impacts of calculators and computers on teaching statistics
4. Nature of school statistics and probability: principles, practice and training
5. Effect of assessment methods on teaching of statistics.

Recommendations to the ISI Education Committee
by the Round Table Conference
on the Impact of Calculators and Computers on Teaching Statistics

Canberra, 20-23 August 1984

Introduction

During the past decade, electronic calculators and computers have become readily available in developed countries; they are now gradually coming into use in the developing countries. Their role in every day life is visible in supermarkets, banks, business and airline offices; in the developed countries, there is an appreciable number of private users of micro-computers, and with the increasing cheapness of electronic components, it seems likely that programmable calculators and computers will gradually spread to the developing countries.

The discipline of statistics, dealing as it does with the collection, storage and evaluation of data, and sometimes relying on the probability models which underlie statistical computations, is increasingly dependent on the use of calculators and computers. Furthermore, for certain aspects of statistics (e.g. simulation, the graphical display of data, the communication and analysis of large sets of real data), the computer is an invaluable tool which opens up new and improved methods of education. These uses have affected statistical practice considerably and it is clear that the teaching of statistics must now change to accommodate the impact of calculators and computers.

The ISI Round Table Conference on the Impact of Calculators and Computers on Teaching Statistics has been concerned with this problem. Following the reading of some 16 papers on various aspects of the topic, together with lively discussions on them, the Conference has made the following unanimous recommendations to the ISI Education Committee.

Recommendations

1. Calculators and computers as statistical tools

- (a) Calculators and computers must be recognized as tools of basic importance in the teaching of statistics, as well as its practice. They should, wherever possible, be made accessible to all teachers and students of statistics; teaching methods and syllabuses in statistics courses must take full account of their availability.
- (b) Educational authorities should be encouraged to include statistical courses making use of calculators and computers in school syllabuses. In all countries, it would be desirable to teach statistics in schools as early as possible; real life statistical examples should be experienced by children by the ages of 11-12 years. In developed countries where calculators and computers are readily available, they should be used as companion tools in this school instruction; in developing countries, it would be helpful for educational authorities to provide a minimal supply of solar-powered calculators in

schools and programmable calculators or computers in universities. Such supplies could perhaps form part of educational aid programmes from the developed world (see also 6(b)).

2. Teacher training and retraining

- (a) To respond constructively to the impact of calculators and computers on statistics, the training of new teachers and the retraining of existing teachers is essential. A start could be made by including statistics in courses for potential teachers; these should be practically oriented, with an emphasis on the use of calculators and computers (computational statistics).
- (b) In such training and retraining, the role of workshops in computational statistics, such as those run by various national statistical societies and educational institutions is important. These workshops should be greatly encouraged.
- (c) At a more general level, projects like the ASA/NCTM's Quantitative Literacy Project in the USA, and the Indian Statistical Institute's ISEC in Calcutta, which respectively train teachers and civil servants in statistics, should be strongly supported. The Sheffield Centre for Statistical Education, which acts as an international resources centre in statistical education, should receive strong backing to explore the integration of calculators and computers in statistics teaching. Its results should be disseminated internationally. Other countries should be encouraged to set up similar research and training centres for statistical education.

3. Educational research

- (a) Teaching Methods: Continuing research is needed to determine
 - (i) at what age and through what methods statistical concepts can be effectively learned by children.
 - (ii) the stage at which calculators and computers can best be introduced in the teaching of statistics.
 - (iii) for what statistical purposes calculators and computers are best suited.
 - (iv) how developments in computers can affect statistical courses and syllabuses; this will require regular monitoring.
- (b) Programming Methods: Research is also needed into programming and its educational value. To what extent does program writing develop the logical and quantitative skills of children and students? How can statistical packages be developed, adapted and improved for school use?

4. Text and software development

- (a) The development of new texts making use of calculators and computers for statistics courses should be strongly encouraged; these should include good statistical data, examples and suggestions for projects. Publishers should be made aware of the importance of providing software to accompany their statistical texts. Such texts, as well as new statistical software will be produced in increasing quantities in future, and some mechanism for their evaluation will become necessary.
- (b) The review of computationally oriented statistics texts and statistical software should remain the responsibility of statistical journals such as *Teaching Statistics* and the *American Statistician*. The ISI should be encouraged to include similar reviews in its publications, and to assist in coordinating information on texts and software, possibly through the Sheffield Centre for Statistical Education.

5. International cooperation and communication

- (a) International cooperation and communication in the teaching of computational statistics is essential, particularly if new methods and procedures are to spread to developing countries. In such an endeavour, the role of the Sheffield Centre for Statistical Education may prove crucial.
- (b) Computer networks for the dissemination of statistical data and information should be encouraged to provide free services to educational institutions. An example of one such network is the Nightingale Network in North America.
- (c) In the context of dissemination of information, an international statistical magazine distributed to all members of ISI could be helpful in promoting international cooperation for a more realistic approach to statistics teaching in the computer age, both in the developed and the developing countries.

6. Manufacturers of calculators and computers

It is clearly in the interest of calculator and computer manufacturers to support the use of their equipment in the practice and teaching of statistics.

- (a) They should therefore make efforts to produce relatively cheap basic calculators and computers as universal teaching aids, bearing in mind the needs and resources of both developed and developing countries. Computers used for teaching classes should have large screens, and teachers should be able to rely on manufacturers for a good range of statistical packages. Some thought should also be given to reducing the costs of network software, and the communication potential of computers.

- (b) Computer manufacturers should be approached by the ISI Education Committee and asked to support the Committee's activities on computational statistics and its teaching. They might be persuaded to offer calculators and computers to key institutions in the developing countries, with the eventual prospect of opening up new markets (see also 1(b)).

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

246
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

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.

COMPUTER ANIMATIONS IN MATHEMATICS TEACHING
AT THE OPEN UNIVERSITY

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 himself '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 prepared 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 lecturer 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, and students. With a screen to watch and then work on, students 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 way of 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 richly 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 else's 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 can 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 else's.

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 from 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.

Summary

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.