Computer Algebra Systems and Undergraduate Mathematics Curriculum

Miroslaw Majewski
Zayed University
majewski@mupad.com

Abstract. In recent years, we faced a number of changes in undergraduate mathematics curriculum. This process was a bit different in each country but the general idea was to accommodate in undergraduate mathematics curriculum topics that are relevant to our times. In many countries, selected topics, which usually were taught in university courses, were moved to the high school curriculum and some classical mathematical topics disappeared completely from high school and even from university. Unfortunately, such changes very rarely addressed issues of using computers in teaching mathematics.

The main objective of this paper is to analyse teaching of mathematics from a computing point of view; to highlight changes or at least potential changes in the mathematics curriculum that follow from using computer applications and especially Computer Algebra Systems; to investigate some of the changes in how we teach mathematics; and to describe new skills that our students acquire from these changes.

Introduction

Teaching mathematics with a computer is becoming a popular trend and it has many different flavours. There are people who teach mathematics in the traditional way and use computers sporadically to illustrate certain topics. At the other end of this spectrum, there are teachers and university professors who use computers as much as possible and sometimes change the curriculum to include topics that earlier were never included in mathematics curriculum (see [6,7,9]). We all know of topics that could not be taught with the traditional paper-and-pencil methods, but which can now be presented and explored easily with the help of computers. This means it is worth to look on mathematics curriculum from computing perspective and think which of these topics it would be worth to accommodate in mathematics curriculum.

One may ask whether we are still dealing with the same mathematics or whether this is actually a new mathematics? Is this still mathematics, or is it something completely new? In this paper, we suggest to initiate a new area in undergraduate mathematics based on the old mathematics, but using computing methods and computers. A key question that arises is where this new subject fits into the curriculum. Should it be a part of mathematics curriculum, part of computer science or a completely new course of study?

Let us begin our analysis by referring to this hybrid subject as *compumatics*, where compumatics is a selection of mathematical topics taught with computers and computing methods that we use to solve mathematical problems.

Examples of such topics include numbers and number families, equations, fractals, dynamic geometry, etc. Methods include numerical methods for solving equations, algorithms and programs for generating numbers from various families, and procedures for producing new mathematical objects, graphical experiments, etc.

Compumatics, as a part of the undergraduate mathematics curriculum, would provide us with many new educational opportunities. In this paper, we explore some of them from the point of view of university courses taught by the author using the compumatics approach.

The author of this paper teaches Computing Foundations and Computational Methods courses to undergraduate students at Zayed University in UAE. He uses for his classes MuPAD for Windows, a Computer Algebra System with command line instruction and a very simple graphics interface, as well as MuPAD Computing Server (MCS), a tool that gives online access to the MuPAD computing engine. In this

last case, the user interface is a simple web form with JavaScript code to process user input through MCS. The MCS technology was described in details in two papers (see [1,2]).

As was pointed out at the beginning of the paper, we will concentrate on new skills that students acquire from compumatics, and discuss some of the changes in the undergraduate mathematics curriculum that are possible with the use of the compumatics approach. Let us start with skills.

New skills gained from compumatics

We all know that using computers is a new experience for students and teachers that isn't part of the traditional way of teaching and learning. There are new skills that we need to develop that allow us to formulate problems in the language of computers, skills that are involved in the use of specific software. Here are some of them.

An algorithmic approach and the top-down design methodology in problem solving

For centuries, algorithms have been an essential part of mathematics. Even at a time when no one dreamed of computers, famous mathematicians formulated algorithms to solve particular problems (see [3]). Computers have given us a good reason for taking a closer look at these algorithms. By writing them in the form of computer programs, we can finally see how they work and analyse their properties step-by-step. We can experiment with these algorithms and, by modifying them, produce new algorithms to solve new problems.

Formulating algorithms is a difficult task for our students. I first realized this when I asked my students to write down the algorithm for solving the quadratic equation. They were already familiar with the idea of an algorithm since they had used algorithms for simple tasks such as evaluating the result of a simple formula using a non-programmable calculator with two variables: display and memory.

It took me a lot of time to develop an understanding what an algorithm actually is and how to produce an algorithm to solve a particular problem. What came in handy at this point was a very important strategy, which in computer science is called a top-down design.

I have noticed that if we introduce the top-down design strategy while solving simple problems and follow it systematically throughout a course of study then students use effectively the same technique to plan their solutions, even while solving mathematical problems without computers. It is important to note that once they have been accustomed to this strategy, the teaching of programming to such students is much easier and more effective. However, in mathematics courses taught with computers we can go even further. We can use a computer algebra system, where a programming language is already built in. MuPAD, Maple or Mathematica are examples of such systems.

In my courses, I use MuPAD since its programming language has a syntax that resembles the way in which we write algorithms in mathematical textbooks (compare for example [4,5]):

When solving mathematical problems my students go through four stages. (1) The initial stage is to lay out the solution as a series of single tasks. At this stage, we concentrate on the overall structure of the solution rather than on the details. (2) Then we develop a formal algorithm for the solution using a pseudo-code language. (3) Next we use the algorithm to write a MuPAD program. (4) The final stage consists of testing the MuPAD program. This is the most critical activity. We analyse each error produced by the program. Very often, this is the starting point for major revisions of the global structure of the algorithm or its details.

From a mathematical point of view, such a program is very useful. Using it we can check the structure of the obtained mathematical object, for example check how quickly the terms of a sequence grow, we can plot the object or animate it to show how it changes depending on its parameters.

Using a procedural approach in problem-solving

When solving complex mathematical problems we often encounter situations where the same or similar algorithms are used to produce different parts of a solution. As soon as we realize this, we can start

producing solutions in a modular form. For example, while looking for roots of an equation $F(x)=0$, we need to realize that if the function $F(x)$ is continuous between two points a and b and $F(a)*F(b)<0$ then there must be a root between a and b. From here, by splitting the interval $[a,b]$ in two equal parts and checking which of them fulfils the above rule, we can easily develop an algorithm to obtain approximate values of the root. This is what, in computer science, we normally call a procedure. Building such procedures for solving mathematical problems with MuPAD makes a lot of sense since it allows us to simplify the top-down design of solutions and frees us from thinking about the details involved at the different levels.

Using an iterative way of thinking about sequential processes in mathematics

Many mathematical problems, even at the undergraduate level, contain iterative processes. For example, how do we produce a decimal value for a number that was written in binary, hexadecimal or any other number system with a non-decimal base? Most of our students do not realize how complex this task is until we ask them to produce an algorithm for it. Because of the different length of the numbers involved, general solutions of these problems use loops and iterations.

In undergraduate mathematics, we have a number of similar problems: finding an approximate sum of a number series, finding an approximate value for an irrational number, etc. In each case we need to determine when to stop the iteration. This means that we need to formulate a condition that will stop the loop. This involves several of other problems that students do not normally consider important. For example, loops may never stop or may not even start when the terminating condition is incorrect. We rarely pay enough attention to these questions in our traditional way of teaching mathematics. However, when using computers such problems are quite significant. When we reflect on these problems, we realize that iterations are actually the gateway to interesting problems and involve sophisticated mathematical objects.

Using a recursive approach in problem-solving

Do we like recursion? Certainly, we like some geometrical patterns obtained by recursion. Unfortunately, in undergraduate mathematics, recursion is not often used explicitly. Why not? Answer is simple: traditional methods are not good enough to explore recursion or even to explain recursive processes. With the advent of computers, however, we are finally able to show how mathematical induction and recursion are related and are able to produce recursive procedures and objects. A well-known Sierpinski triangle, a Menger sponge and many other mathematical patterns can be produced with as many recursive steps as we need.

Recursion was never easy for students. It is still not easy now. However, when we use good and interesting examples, students are able to understand it, use it and sometimes even enjoy it. One such example are L-systems. In MuPAD or in Mathematica, there are procedures to produce an L-system with a given initial figure and rules of creation. In my classes, I found that this example is a good turning point on the way to understanding recursion. In fact, my students never treat this topic as a part of mathematics. They always consider it as a kind of mathematical art. However, what is most important for me as an educator is that the students develop a good understanding of what recursion is and how to use it in simple situations. For example, my students are able to develop simple procedures that will produce Fibonacci numbers, Lucas numbers and many other types of numbers in a recursive way.

Applying experimental methods in solving mathematical problems

Mathematics has always been a theoretical discipline. However, many teachers used some form of experimentation in their classroom. For example, they solve a few similar problems and drew appropriate conclusions from the experiment. With computer and computer algebra, our ability to experiment freely has become a reality. Even a few lines of programming code for a mathematical problem can be turned into a computing machine into which we can feed arbitrary data and see what is the output might be and what possible problems could arise in the process. We can easily check hundreds of input data without worrying about their complexity, and can produce results in a matter of minutes or even seconds. What we can do with computers in a short period of time would either have required many hours, days or even years of hard work

before, or would even have been impossible to do with pencil and paper, now may take but a few second with computer algebra.

Other types of experimentation are available in dynamic geometry packages such as Cinderella and Cabri. Here students can move geometry objects, rotate, scale, and perform hundreds of other experiments similar to the ruler-and-compass activities in traditional geometry classes. This type of experimentation is a completely new phenomenon in teaching mathematics.

Mathematical discovery through computing methods

From experiments with computers to mathematical discovery is often only a further step involving a bit of imagination. It is therefore not surprising to learn how often teachers using computer algebra and other mathematical packages in their classes report new mathematical facts discovered by their students. It is almost certain that the opportunity to use attractive mathematical software in the classroom can create a lot of excitement and appeal to many students. By using these innovative tools they develop a completely new way of thinking about mathematical objects. Very often they easily understand constructions that are difficult to imagine for a person unfamiliar with mathematical software.

Developing 2D and 3D imagination

As we all well know, drawing 3D curves and surfaces on paper or on the blackboard is rarely successful. Only, a very few students and teachers are able to produce accurate images. With traditional methods, we were usually able to visualize only certain simple 3D objects such as planes, spheres, and selected quadrics. Perhaps some of us were even able to produce models of polyhedrons. For an average student, most of other 3D objects were almost impossible to draw. With a technology like MuPAD or some other computer algebra system, however, we can now show 3D objects from all sides, make them transparent, and explore many other geometric properties. MuPAD, for example, offers a very sophisticated environment for working with 3D objects. Using this feature of MuPAD, mathematics students are able to develop their understanding of 3D space, perspective and shadowing. All of these skills are invaluable for future architects, computer game developers and graphics designers. In addition to exploring the geometric properties of 3D objects, we can use computer algebra to enhance our algebraic understanding of space by working with 3D coordinate systems, quaternions (see [9]) and formulae for many 3D surfaces. According to my observations, 3D features of computer algebra are especially exciting for all kinds of students when they are exploring geometry on their own, or are doing independent projects.

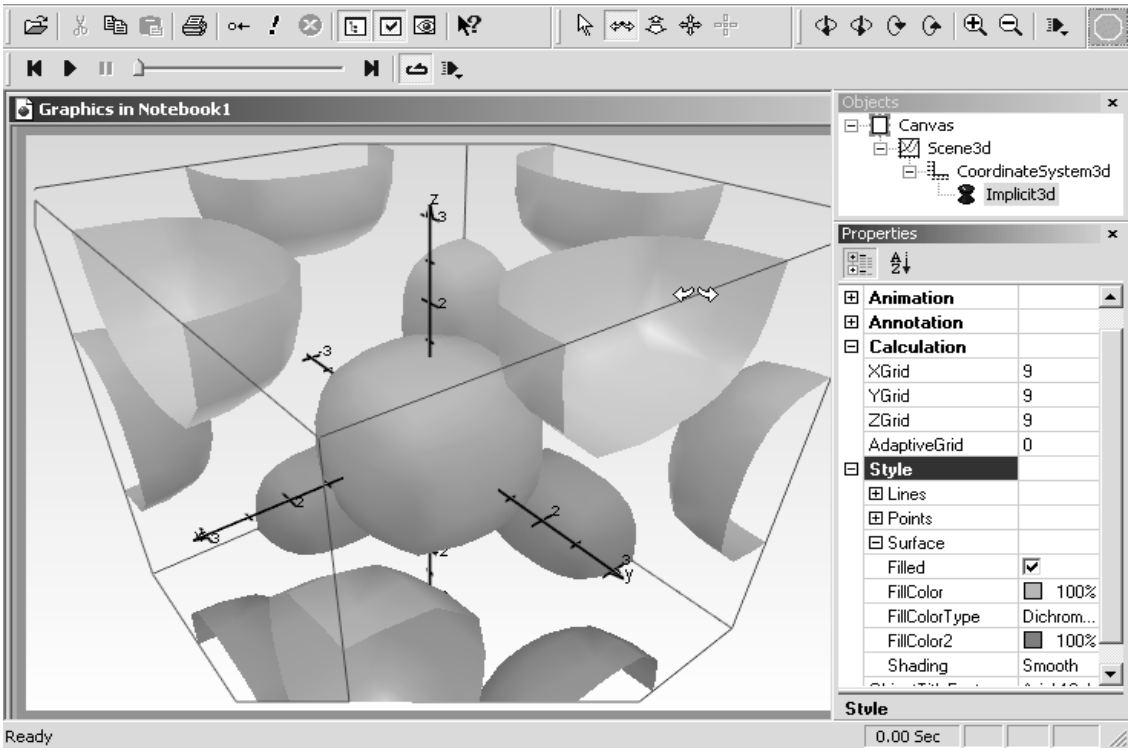

Figure 1 MuPAD offers a very sophisticated virtual reality environment for working with 3D objects.

New topics in mathematics curriculum

In the previous chapter, we concentrated on skills that follow from compumatics, now let us examine possible changes in undergraduate mathematics curriculum. As we have mentioned earlier, there are many topics that were difficult or impossible to teach without computers. After all, some topics that have been in the mathematics curriculum for centuries can now be looked at from a computing perspective. Let us consider some examples where compumatics can make a difference. Undoubtedly, many readers will be able to add more examples to this mosaic.

Numbers, number systems, number families and number properties

In my CIS110 Computing Essentials course, I teach a selection of topics related to numbers in different bases. I teach how to convert numbers from one base to another, how to obtain special families of numbers, and I explore properties of some numbers. There is no doubt that most of these activities in my class can be done by hand. However, this would slow down the tutorials and because time is limited, I would not be able to cover many interesting examples. With MuPAD, my students gain much more than knowledge of numbers. They also gain an opportunity to apply and practice certain algorithms required to work with numbers. For example, they are able to find out that Horner scheme can be used in many mathematical situations, even in the case of numbers in different bases and the study of polynomials. They are also able to discover that Horner scheme can be used to calculate numbers or manipulate polynomials with fewer operations than with any other method performing the same task. This represents a significant insight gained by writing algorithms in the form of small programs. This is the type of knowledge that is completely missing from our traditional way of teaching mathematics.

The world of fractals

Benoit Mandelbrot has introduced into mathematics completely new objects, as well as a new ways of generating them. Now his fractals have become part of our daily lives. We can see them in newspapers, artistic creations, in films and in music. We now use fractal methods in very sophisticated technologies.

However, there are very few textbooks of undergraduate mathematics where fractals are even mentioned. Why? We know for sure that we can use paper and pencil to explain how a fractal is created, but that is about all. Most fractals cannot be created without a computer. This is the response to the question why most mathematics courses taught without a computer cannot introduce fractals in an effective way. But we can do so using compumatics.

Let us ask the question – can we afford not to teach our students such important mathematical concepts like fractals and their applications?

Recursive objects in mathematics

We were already talking about recursive approach in solving mathematical problems. Now let us to concentrate a moment on recursive objects. In undergraduate mathematics we teach mathematical induction and how to prove theorems using mathematical induction. Recursion is a constructive version of mathematical induction. Instead of proving a property we use induction to construct new objects. Each new object is created from a previous one by the same operations. If we know the starting object and the recursive rules, we are able to produce new objects with each step. Thus creating recursive objects can give our students more hands-on experience. Such well known mathematical objects as Fibonacci and Lucas numbers, Sierpinski triangles, Koch snowflakes or L-systems in general and many others rarely appear in undergraduate mathematics textbooks. However, we can find out more about them in some books on mathematical recreations, for example in the famous books by Clifford Pickover (see [8]). Again the question is: why we do not teach these topics? Are they not interesting or perhaps even useless? They are certainly interesting for thousands of readers of Pickover's books. In modern technology and science there also are very deep applications of such recreational objects. Let me to mention L-systems and modelling plants in biology or modern music.

Computer arithmetic

It is remarkable how often people blame non-existing bugs in computer software for their own errors. This happens so often. However, with a little knowledge of computer arithmetic we can easily point out where certain errors come from and how to avoid them. For example, in undergraduate mathematics we do not mention how numbers are stored in computer memory and what are the implications of the fact that we can use only limited memory space for representing numbers. We often do not realize that a simple instruction like the one below in a computer program may not work the way we expect:

```
if 1.0=1 then print("Got it")
```

Do our students know that irrational numbers in a computer are not really irrational? Do they know that the set of all real numbers in the computer is a finite set? There are many other topics like the mentioned ones related to mathematics and computers. Shall we teach such topics in undergraduate mathematics? Well, if we wish our students to have a good understanding of what they can expect from computers and pass successfully through courses where computers are essential tools, then we should teach things that are relevant for our students' further study and their profession life.

Graphs and paths on graphs

Here is another topic that is neglected in undergraduate mathematics. Yet it is so easy to introduce it using matrices and can be used to illustrate many interesting applications of graphs. With compumatics, graphs can be easily investigated. For example, MuPAD has a quite extensive library of procedures for experimenting with and analysing graphs. By using simple commands, we can explore properties of graphs and visualize them.

Other mathematical topics

By looking carefully thorough the covered topics in the undergraduate mathematics curriculum, we can find several topics that could be taught successfully with computers and provide a new perspective on them. Among them are probability, statistics, and geometry. Perhaps we should analyse our current curriculum and identify other areas where the compumatics approach could lead to innovative teaching and learning. This would certainly make the mathematical knowledge of our students more relevant to their future professions and give them insight into issues that have now become relevant and important.

Summary

Compumatics can as either be taught as a new undergraduate course or can be incorporated in the regular undergraduate mathematics curriculum. Either way, it will make a significant difference to our undergraduate mathematics education. Here are some of the benefits it offers.

- We can teach and learn topics, which we could not really teach or learn before because of their computational and structural complexity.
- We can teach and learn mathematics in a more structured way by representing intuitively understood algorithms in different programming languages.
- Students can learn new strategies of problem solving.
- Students can acquire new enthusiasm to mathematics through an active experimentation.
- Students can learn by discovery.
- The teaching process becomes more dynamic, especially in problem solving. This may lead to completely different ways of thinking.
- Students will be still able to learn old mathematical facts, but in different ways and quite often from a different points of view.
- Students will get into new interesting and visual areas of mathematics.
- Some of the old, important but less exciting and more difficult topics can be learned faster. This will free up time for new more attractive topics.
- Many old and difficult topics taught with computer may get a new flavor and may even become interesting for many students (for example, probability).
- Students will be still learning how to prove theorems, but perhaps some of the proofs will be different.
- High school students will be better prepared to study computer science and university mathematics.

References

1. A. Bhatti, M. Majewski, *Using MuPAD and MCS to Develop an Online Course for Quantitative Sciences* – to be published in the international journal Innovations in Teaching and Learning in Information and Computer Sciences, 2003.
2. A. Bhatti, M.Majewski, *MuPAD Computing Server Demystified*, mathPAD Online, 2003.
3. Jean-Luc Chabert et Al., *A History of Algorithms – from the Pebble to the Microchip*, Springer Verlag, 1994.
4. R.G. Dromey, *How to Solve it by Computer*, Prentice-Hall, Series in Computer Science, 1982.
5. M. Majewski, *MuPAD Pro Computing Essentials, Second Edition Revised for MuPAD 3.0*, Springer Verlag, 2004.
6. M. Majewski, F. Szabo, *Integrating MuPAD into the Teaching of Mathematics*, Proceedings of The Fifth International Conference on Technology in Mathematics Teaching, University of Klagenfurt, Austria, August 6 - 9, 2001
7. M. Majewski, F. Szabo, *Developing Open and Flexible Computing Environments for Teaching Mathematics and Science*, Proceedings of the 2nd International Conference on the Teaching of Mathematics by John Willey and Sons, 2002
8. C. Pickover, *Keys to infinity*, John Wiley & Sons, 1996.
9. F. Szabo, *Linear Algebra – An Introduction Using Maple*, Harcourt Science, Academic Press, 2002