Introduction to Symbolic Computation: a Maple/MATLAB Course

Jan Verschelde*

17 May 2005

Abstract

Virtually every student who chooses to learn how to do computations on a computer ends up taking a course in numerical analysis. Compared to numerical analysis, the field of computer algebra is relatively young and has fewer active researchers. The commercial success of systems like Maple combined with the obvious wider range of applications (again compared to numerical computations) show an enormous potential for the future. We dare to predict that, just as numerical analysis is routinely taught at the undergraduate level, introductory courses in symbolic computation will be just as common and essential.

The MSCS department at UIC offers "MCS 320", an undergraduate course aimed at introducing students to use mathematical software. Ideally, this course will be taken before the "MCS 471" (numerical analysis). Based on five recent successful offerings of this course, valuable experience was acquired on what should and what can be taught in an introductory course in symbolic computation. Compared to the canonical and mandatory list of topics expected to be covered in any first numerical analysis course, we can outline a typical list of topics, balancing the mathematical (im)maturity of the students, the needs of other math courses, and taking into account future trends and current research developments.

1 Introduction

The department of Mathematics, Statistics, and Computer Science at the University of Illinois at Chicago offers a program in Mathematical Computer Science (MCS). Once the students have passed the introductory programming courses, they have access to a wide variety of courses, organized into five clusters: (1) Coding, Cryptography, and Number Theory; (2) Combinatorics and Theory of Computation; (3) Programming; (4) Algorithms and Operations Research; and (5) Scientific Computation. The majority of the masters degrees awarded by the department are in MCS.

Regarding the five clusters above, the course "Introduction to Symbolic Computation" (MCS 320) is the starting point in the fifth cluster on Scientific Computation, to be followed by "Numerical Analysis" (MCS 471). In numerical analysis, Maple and MATLAB are frequently used to illustrate numerical algorithms and to facilitate computer projects. As MCS 320 introduces students to programming in Maple and MATLAB, the course also belongs to the Programming cluster. Lastly, but certainly not least, many courses in coding and cryptography have computational supplements using computer algebra. So students who have taken MCS 320 are better prepared when taking courses in the coding, cryptography and number theory cluster.

*Department of Mathematics, Statistics, and Computer Science, University of Illinois at Chicago, 851 South Morgan (M/C 249), Chicago, IL 60607-7045, USA. *Email:* jan@math.uic.edu or jan.verschelde@na-net.ornl.gov. *URL:* http://www.math.uic.edu/~jan. This material is based upon work supported by the National Science Foundation under Grant No. 0134611.

The author has offered the course four times (Spring and Fall 2001, Spring 2003 and Spring 2004). In Fall 2004, Anton Leykin taught the course, based on the lecture notes developed by the author. The course is popular with students. Often students were denied enrollment by the limited seating capacities of the computer labs where the course took place.

The textbook used for MCS 320 is [9]. While [9] is solid and unique in many regards, for MCS 320 it had three major deficiencies. Firstly, the book is at graduate level and too difficult for junior undergraduates whose mathematical education (like at any large scale university in the US) is far below the level of European universities. Secondly, the book does not cover MATLAB, which is after all the standard for scientific computing in almost all engineering disciplines. Thirdly, the second edition [9] went out of print and it took (too) long before the third edition [10] became available. Because of these deficiencies, the author started to develop a set of lecture notes. The structure of the lecture notes follows the table of contents of [9], but the notes are tailored more towards the specific needs and capabilities of the students. The author still recommends [10] as a reference, but students can take the course without purchasing [10].

This paper outlines the syllabus for the course, ideas and experience with computer projects, and ends with some perspectives for the future of courses like MCS 320. Such courses seem to be rare, although we found an interesting report on an educational experiment in [13]. We also recommend [5] and [18] for thoughful comments on using computer algebra to teach at the undergraduate level.

2 Goals of the Course

From a utilitarian point of view, the course introduces students to "canned" software. While a computer algebra system like Maple seems easy to use at first, we need a systematic and gradual introduction to the software to take advantage of its full power. The goals we try to achieve are the following:

High level of Scientific Programming: Programming at the low level is hard, actually very hard, if one wants to develop software which solves practical problems efficiently and reliably. We see a Maple worksheet as a tool to develop a high level mathematical program.

Study of problems in Symbolic Computation: Just as numerical analysis is routinely taught, symbolic computation is an important subject area in its own right. While the research community is relatively small, the commercial success of packages like Maple and Mathematica stimulate a constant need for improvement. Faster computers always demand better software.

Integration of Symbolic and Numeric Tools: A practical application can seldomly be solved entirely by symbolic or numerical means. The symbolic approach usually requires too much of the available computer resources and a purely numerical solution is not always reliable enough.

These goals are at the basis of the discipline of scientific computing. Because scientific computing is so important and relevant, it serves the students well not to wait till they choose to enter a graduate program to introduce them in a systematic and structured way to the major software tools of scientific computing.

3 List of Topics

The course meets three times a week for 50 minutes, for 15 weeks during one semester. With two midterms, one review session before each midterm, and three review sessions at the end, this leaves 37 lectures, spread over five modules:

1. **First Steps with Maple:** We will spend nine lectures introducing computer algebra with Maple, extensively using the nice Maple worksheet interface. When used properly, worksheets provide an

important aid to mathematical modeling. Computer algebra emphasizes the exactness of calculations and allows us to compute beyond the hardware integers or machine floating-point numbers.

2. **Polynomials and Rational Expressions:** Computer algebra packages are essentially "equation crunchers" (in analogy with, or contrast to, the number crunching algorithms of numerical analysis). Therefore it is worthwhile to look into the internal data representations and basic expression manipulation facilities in Maple. One major problem in symbolic computation is to decide whether two expressions are mathematically equivalent. Symbolically, we define canonical and normal forms to represent expressions. Numerically, we compare the values of the expressions evaluated at sufficiently many random points.

3. **Calculus:** The operations of differentiation and integration take functions on input and return functions or numbers. In this module we explore Maple's capabilities of defining function which manipulate and return functions. While we see how to make recursive procedures efficient in Maple, this part is not really about programming (at least not at the same low level as programming in C for instance), but about the mathematical aspects of functions. Note that in this module we will not touch the Maple package "student" which implements the typical operations a student in a calculus course encounters. In this "Calculus" part we consider the more basic operations of calculus.

4. **Advanced Maple:** We start this last module by looking into the composite data structures of Maple. Then we discuss the ability to work with assumptions and investigate how simplifications are performed. Plotting is one of the main strengths of Maple, we will spend two lectures on plots, in two and three dimensions. The last three lectures in this part are devoted to solving polynomial, differential, and linear equations. There is a wide variety of interesting books on "solving mathematical problems using Maple", see for instance [15] for plotting curves and surfaces, [1] for ordinary differential equations and [3] for partial differential equations. Linear algebra with Maple is treated in [2].

5. **Introduction to MATLAB:** MATLAB is system that originally was designed to illustrate common linear algebra operations. As linear algebra is so useful, the original program has grown into a vast scientific software system. For many of the features of MATLAB we want the students to learn, GNU Octave (available via `http://www.octave.org/`) works just as well.

The outline of the course (as it was offered in Spring 2004) with a detailed schedule of the lectures in Appendix A. Appendix B lists the questions on the final exam.

4 Computer Projects

The course is organized in a computer lab. The actions of the instructor are projected onto a big screen and the students are expected to experiment with the software hands on in real class time. The last twenty minutes of every Friday ends with a quiz and also the exams are on line, open book, and in a computer lab. Students who pass up the opportunity of the in-class on-line, hands-on experience in the lab usually lack the agility to master the multitude of problems given to them during the exams.

The problems taught in class and questioned during exams are typically problems which can be solved in a couple of minutes by an expert user. Leaving the course at this would do grave injustice to the actual capabilities of the software tools, namely: the capability of building realistic computer models of problems in science and engineering. Especially the Maple worksheet environment invites to document the computations in such a systematic way that it blurs the lines between an actual paper (to be read) and a computer program (to be executed).

Three computer projects are assigned during the course. The first two projects are with Maple, the last one with MATLAB (or with Octave). Designing the right type of project (matching the skills and interests

of the students, but still realistic enough) has demanded a great deal of creativity of the instructor. A source of inspiration for the first project is coding and cryptography, which is an excellent application field to demonstrate the capabilities of extended arithmetic of Maple. Ideas for such projects were found in [14].

The project titles for the first project are
1. The RSA Cryptosystem with Maple (Spring 2001);
2. BCH Codes with Maple (Fall 2001);
3. Public Key Cryptography with Elliptic Curves (Spring 2003);
4. The Chinese Remainder Theorem in Cryptography (Spring 2004).

The second computer project demonstrates Maple's graphing capabilities and uses difference or differential equations to model popular topics of applied mathematics. Sources for inspiration are [1, 3, 6, 7, 8, 15]. Titles for the second project with Maple are
1. Heat-Seeking Particles with Maple (Spring 2001);
2. A Predator-Prey Model with Maple (Fall 2001);
3. Modeling the Outbreak of War (Spring 2003);
4. A Simple Model of Billiards (Spring 2004).

Compared to the two Maple projects, the scope of the third MATLAB project is limited. Most students are towards the end of the course quite comfortable with Maple's worksheet environment and do not quite like MATLAB or Octave prompting commands without the ability to clean up a messy session full of errors. Note that the majority of the students has not yet taken numerical analysis and most students not yet familiar enough with linear algebra. Titles for the third project with MATLAB are
1. Curve Fitting with MATLAB (Spring 2001);
2. Image Processing with MATLAB (Fall 2001);
3. Markov Chains with MATLAB (Spring 2003);
4. Wire-frame modeling with MATLAB (Spring 2004).

The setup of each computer project is typical. The students are presented with a computer model in the form of a Maple worksheet or a set of MATLAB .m files. The assignments fall in three categories. First, the students are required to make some small adjustments to the model, demonstrating they understand how to operate with the model. The second assignment requires them to solve a problem using the computer model, usually, this is the core of the problem. The third assignment consists of some slightly more advanced issue. The author feels it very important to give precise feedback to the students in the form of a worked solution, to demonstrate how the software tools are used by experts.

5 Textbook and Lecture Notes

The starting point to develop lecture notes was the introduction to MATLAB (not covered by what was then the current textbook [9]), in the last three weeks (or nine lectures) of the course. As a reference we recommend [11]. Inspiration was also found in [12, 16, 17, 22, 24].

With relative ease, the worksheets posted at the end of the each Maple lecture could be turned into documented command sequences which then in turn provided enough material for two or three page lecture notes to be distributed at the beginning of each lecture in the following semester when the course was offered again. The flexibility of adding information tailored and specific for our students made that the lecture notes have become more valuable than any other textbook available. Questions asked during quizzes and exams have become exercises and prepare students better for the skills they have to acquire.

In addition to [9] (and its third edition [10]), useful ideas and inspiration were found in [4] and [19, 20, 21] and [23].

At the time of this writing, the author is still unsure whether the lecture notes should be left in their dynamic, electronic format, or whether they should be archived in textbook format.

6 Discussion and Perspectives

That Gröbner bases can be introduced to students hardly familiar with matrices sounds rightfully suspicious. However, from a *technological* point of view, launching the Gröbner engine (a term coined by Carlo Traverso in the context of the European PoSSo Project) on a polynomial system in lexicographical order brings the system in a triangular form. In this restricted setting, a student is able to model a problem, e.g.: constrained optimization with Lagrange multipliers, bring in into the software and interpret the results of the computations.

As such, an introductory course in symbolic computation at the undergraduate level is more aimed at teaching students valuable skills and problem solving techniques rather than introducing them to higher mathematics. We haste to add that students will get a fair taste of mathematics which goes well beyond their level, but the main goals will appeal to a very broad audience of scientists and engineers. This note is an appeal to those – both inside and outside the scientific community of computer algebra – interested in renewing the way mathematics is taught, to consider introductory courses in symbolic computation.

References

[1] M.L. Abel and J.P. Braselton. *Differential Equations with Maple V.* Academic Press, second edition, 2000.

[2] W.C. Bauldry, B. Evans, and J. Johnson. *Linear Algebra with Maple.* John Wiley, 1995.

[3] D. Betounes. *Partial Differential Equations for Computational Science: with Maple and Vector Analysis.* Springer-Verlag, 1998.

[4] R.M. Corless. *Essential Maple 7. An introduction for Scientific Programmers.* Springer-Verlag, 2002.

[5] R.M. Corless and D.J. Jeffrey. Scientific computing: One part of the revolution. *J. Symbolic Computation*, 23(5-6):485–495, 1997.

[6] R.H. Enns and G.C. McGuire. *Computer Algebra Recipes: A Gourmet's Guide to the Mathematical Models of Science.* Springer-Verlag, 2002.

[7] W. Gander and J. Hřebíček. *Solving Problems in Scientific Computing Using Maple and MATLAB.* Springer-Verlag, third edition, 1997.

[8] R.L. Greene. *Classical Mechanics with Maple.* Springer-Verlag, 1995.

[9] A. Heck. *Introduction to Maple.* Springer-Verlag, second edition, 1996.

[10] A. Heck. *Introduction to Maple.* Springer-Verlag, third edition, 2003.

[11] D.J. Higham and N.J. Higham. *MATLAB Guide.* SIAM, 2000.

[12] B.R. Hunt, R.L. Lipsman, and J.M. Rosenberg. *A Guide to MATLAB, for beginners and experienced users.* Cambridge University Press, 2001.

[13] E. Kaltofen. Teaching computational abstract algebra. *J. Symbolic Computation*, 23(5-6):503–515, 1997.

[14] R.E. Klima, N. Sigmon, and E. Stitzinger. *Applications of Abstract Algebra with Maple.* CRC Press, 2000.

[15] G. Klimek and M. Klimek. *Discovering Curves and Surfaces with Maple.* Springer-Verlag, 1997.

[16] A. Knight. *Basics of MATLAB and beyond*. Chapman & Hall/CRC, 2000.

[17] C. Moler. *Numerical Computing with MATLAB*. SIAM, 2004. See http://www.mathworks.com/moler for electronic version.

[18] M.B. Monagan. Worksheets and notebooks: Can we teach mathematical algorithms with them? *J. Symbolic Computation*, 23(5-6):535–549, 1997.

[19] M.B. Monagan, K.O. Geddes, K.M. Heal, G. Labahn, and S.M. Vorkoetter. *Maple V Programming Guide*. Springer-Verlag, 1998.

[20] M.B. Monagan, K.O. Geddes, K.M. Heal, G. Labahn, S.M. Vorkoetter, J. McCarron, and P. DeMarco. *Maple 9 Advanced Programming Guide*. Maplesoft, 2003.

[21] M.B. Monagan, K.O. Geddes, K.M. Heal, G. Labahn, S.M. Vorkoetter, J. McCarron, and P. DeMarco. *Maple 9 Introductory Programming Guide*. Maplesoft, 2003.

[22] S. Nakamura. *Numerical Analysis and Graphic Visualization with MATLAB*. Prentice Hall, 1996.

[23] D.I. Schwartz. *Introduction to Maple 8*. Prentice Hall, 2003.

[24] S.D. Stearns and R.A. David. *Signal Processing Algorithms in MATLAB*. Prentice Hall, 1996.

Appendix A: Course Outline of Spring 2004

Part I	L-1	Mon 12 Jan	Introduction to Computer Algebra	*First Steps with Maple*
	L-2	Wed 14 Jan	Getting Started and Getting Help	
	L-3	Fri 16 Jan	Exact and Floating-Point Numbers	
		Mon 19 Jan	**Martin Luther King Day — no class**	
	L-4	Wed 21 Jan	Algebraic and Complex Numbers	
	L-5	Fri 23 Jan	Assignment and Unassignment	
	L-6	Mon 26 Jan	Evaluation and Names of Variables	
	L-7	Wed 28 Jan	Types, Attributes, and Properties	
	L-8	Fri 30 Jan	Input/Output Formats and Files	
	L-9	Mon 2 Feb	I/O of Data and Code Generation	
Part II	L-10	Wed 4 Feb	Univariate and Multivariate Polynomials	*Polynomials and Rational Expressions*
	L-11	Fri 6 Feb	Rational Functions and Conversions	
	L-12	Mon 9 Feb	Representation of Expressions	
	L-13	Wed 11 Feb	Substitution, Expansion, and Factorization	
			Project One due Friday 13 February at 2PM	
	L-14	Fri 13 Feb	Normalizing, Collecting, and Sorting	
	R-1	Mon 16 Feb	Review of the first 14 lectures	
	E-1	Wed 18 Feb	**First Midterm covers lectures 1 to 14**	
Part III	L-15	Fri 20 Feb	Defining Mathematical Functions	*Calculus*
	L-16	Mon 23 Feb	Maple Procedures and Recursion	
	L-17	Wed 25 Feb	Working with Functions	
	L-18	Fri 27 Feb	Symbolic and Automatic Differentiation	
	L-19	Mon 1 Mar	Integration and Summation	
	L-20	Wed 3 Mar	Series, Approximations, and Limits	
Part IV	L-21	Fri 5 Mar	Sequence, Set, and List	*Advanced Maple*
	L-22	Mon 8 Mar	Array, Table, and Conversions	
	L-23	Wed 10 Mar	Assume and Simplification	
	L-24	Fri 12 Mar	Two-dimensional Plots	
	L-25	Mon 15 Mar	Three-dimensional Plots	
	L-26	Wed 17 Mar	Solving Equations	
	L-27	Fri 19 Mar	Differential Equations	
			Project Two due Monday 29 March at 2PM	
	L-28	Mon 29 Mar	Linear Algebra	
	R-2	Wed 31 Mar	Review of the lectures 15 to 28	
	E-2	Fri 2 Apr	**Second Midterm covers lectures 15 to 28**	
Part V	M-1	Mon 5 Apr	Introduction to MATLAB	*Introduction to MATLAB*
	M-2	Wed 7 Apr	Plotting with MATLAB	
	M-3	Fri 9 Apr	Polynomials and Fitting	
	M-4	Mon 12 Apr	Programming in MATLAB	
	M-5	Wed 14 Apr	MATLAB as Drawing Tool	
	M-6	Fri 16 Apr	Images and Movies in MATLAB	
	M-7	Mon 19 Apr	Signal Processing in MATLAB	
	M-8	Wed 21 Apr	Special Matrices in MATLAB	
	M-9	Fri 23 Apr	Linear Programming in MATLAB	
	R-3	Mon 26 Apr	Review of Maple, material covered in 1st Midterm	
			Project Three due Wednesday 28 April at 2PM	
	R-4	Wed 28 Apr	Review of Maple, material covered in 2nd Midterm	
	R-5	Fri 30 Apr	Review of MATLAB	

Appendix B: Questions on Final Exam of Spring 2004

Students were allowed two hours to solve the ten problems.

1. Explain the difference between a numerical and a symbolic factorization of a polynomial in one variable in Maple.

   (a) What are the relevant Maple command(s) to compute a symbolic factorization?

   (b) Describe how in Maple one computes a numerical factorization of a polynomial in one variable.

2. Consider the output of `dismantle(p)`:

```
SUM(5)
   PROD(5)
      NAME(4): x
      INTPOS(2): 1
      NAME(4): y
      INTPOS(2): 1
   INTPOS(2): 1
   NAME(4): x
   INTPOS(2): 1
```

   (a) Draw the directed acyclic graph to show the internal representation of `p`.

   (b) What does the expression `p` look like? Give the formula for `p`.

   (c) Suppose we do `q := subs(1=Pi,p);` how many times will we see π in the output? Justify your answer.

3. Divided differences for a function $f(x)$ are defined as follows:

$$f[x_1, x_2, \ldots, x_{n-1}, x_n] = \frac{f[x_2, \ldots, x_{n-1}, x_n] - f[x_1, x_2, \ldots, x_{n-1}]}{x_1 - x_n}, \quad \text{for } n > 1$$

   and $f[x_k] = f(x_k)$, for all k.

   Write a Maple procedure `dvd` which computes divided differences for any f, and is called like `dvd[a,b,c,d](f)`. The output of `dvd[a,b,c,d](f)` shows

$$\frac{\frac{\frac{f(d)-f(c)}{c-d} - \frac{f(c)-f(b)}{b-c}}{b-d} - \frac{\frac{f(c)-f(b)}{b-c} - \frac{f(b)-f(a)}{a-b}}{a-c}}{a-d}.$$

   For simplicity, assume the user always makes the correct call to `dvd`, i.e.: include no error handling features.

4. Consider the following Maple session:

```
[> expand(p);
   2  2       2         2         2                       2
5 x  a  + 61 x  a + 66 x  + 10 x a  + 121 x a + 121 x + a  + 15 a + 44

[> s := solve(p,x);
                                  2
           -10 a - 11 + sqrt(80 a  + 116 a + 25)
  s := 1/2 -------------------------------------,
                          5 a + 6
                                  2
            -10 a - 11 - sqrt(80 a  + 116 a + 25)
       1/2 -------------------------------------
                          5 a + 6
```

   (a) For which values of the parameter `a` is the answer valid?

(b) Give the Maple command(s) to treat the special case(s).

(c) As you can see the polynomial `p` is shown in expanded form. Give the Maple command to "un-expand", i.e.: what is the command which reveals it all?

5. (a) Explain why the normalization of expressions is so important in symbolic computation.

(b) Describe a numerical test to solve a problem for which the symbolic method needs a normal form.

(c) Compare the numerical with the symbolic approach to this problem.
What are the advantages and disadvantages of the two approaches?

6. Consider the following Maple session:

```
[> p1; p2; p3;
                          p1 := 4 + 2 x + 2 y + z + x y
                                                       2
                       p2 := 4 + 2 x + 2 y + 2 z + 4 y
                                             2
                              p3 := x + y + x

[> gb := grobner[gbasis]([p1,p2,p3],[x,y,z],plex);
                           3         2
  gb := [11710 x + 792 z  + 5437 z  + 21200 z + 30068,
                                 2               3
        11710 y - 5150 z - 2953 z  + 5168 - 568 z ,
                      2            3      4
        652 z + 346 z  + 324 + 79 z  + 8 z ]
```

(a) From the output of `gbasis`, how many complex solutions does the system defined by the polynomials `p1`, `p2`, `p3` have? Justify your answer.

(b) Describe how you can find all complex solutions of the system.

(c) Suppose we were only interested in the rational or real solutions to the system. Describe how you would modify your answer to the previous question to limit Maple to compute only the real or rational solutions.

7. Sometimes Maple displays symbols like _C1, _C2, etc... in its output.

(a) What does this mean?

(b) Give a good example of a problem which would show symbols like _C1, _C2, etc...

8. Explain the difference between the commands **polyfit** and **spline** in MATLAB.

(a) Describe a problem for which you should use polyfit rather than spline.

(b) Describe a problem for which you should use spline rather than polyfit.

9. To find a fixed point $x = f(x)$ for some function f, we may apply the iteration

$$x_{k+1} = f(x_k), \quad k = 0, 1, \ldots$$

(a) Write the MATLAB function, using prototype

```
function x = fixpti(f,x0,n )
%
%  Applies the fixed-point iteration x(k+1) = f(x(k))
%  n times, starting at the point x0.
%  On return is the final value for x of this iteration.
%
```

(b) Give the function call to compute a fixed point of $x = \sin(x)$, using 30 iterations, and starting at 1.

10. Give **all** MATLAB commands to solve the following problem:

$$\max_{x,y} 23x + 44y$$
$$\text{subject to} \begin{cases} x + 12y & \leq & 123 \\ 24x + 30y & \leq & 912 \\ -2x + 83y & \leq & 134 \\ x \geq 0, y \geq 0 \end{cases}$$