Maple as a Tool for Selfstudy and Evaluation

N. Van den Bergh (norbert.vandenbergh@rug.ac.be)
T. Kolokolnikov (theodore.kolokolnikov@rug.ac.be)
Faculty of Applied Sciences
University of Ghent, Belgium

Abstract

We report on the results obtained so far within the framework of the ALICE-project at the University of Ghent. ALICE self-study modules in linear algebra, calculus and theoretical mechanics have been in use for over four years at our institute. Each module consists of a collection of Maple worksheets, containing theoretical materials, worked-out examples and exercises. For exercises which require symbolic answers or in which the verification of the answers involves complex mathematical operations, the use of a computer algebra package is an indispensable tool. A typical such question is to compute a basis for the vector space of all trace-free 2x2 matrices. Both $\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$ and $\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, (as well as infinitely many other choices) are correct answers. Grading this question requires therefore type conversion and some elementary linear algebra operations - things that Maple is good at. For automated delivery and evaluation of such exercises a fast and powerful web-server was developed, the Alice Interactive Mathematics (AIM) server. Its distinguishing features are the use of Maple as the engine and implementation language; versatility in question and quiz design, including the randomization of quizzes and questions; its visualization of mathematical formulae (without using GIF-images); the speed with which individual quizzes are generated; various feedback mechanisms; its facilities for giving partial credits; extensive grade reporting and monitoring capabilities; ability to collect surveys and the use of a web interface for both teacher and student.

1 Maple as a tool for self-study

In October 1996 the ALICE (Active Learning in a Computer Environment) project was launched at the University of Ghent. The target was a relatively small group of students (about 30 annually) entering the engineering faculty after one to four years of technical education, or after some years of experience in the industry. In contrast to the main body of about 300 engineering students annually entering the faculty, these students are technically well skilled, but have had little contact with abstract mathematics during the last years of their education and need some form of remedial teaching. To make matters worse, the students involved usually follow a compressed curriculum, enabling them in principle to complete their studies in two to four years, rather than the customary five years. The resulting very tight class schedule or (in some cases) also the students' ongoing professional occupations make it then impossible for them to attend the lectures and/or exercise classes.

For many years the main difficulty in the first term was the linear algebra course, which by the majority of these students was experienced as being highly 'abstract'. Therefore it was decided to develop a linear algebra package (a) which is suitable for selfstudy, (b) which contains material for self-evaluation and (c) which covers the same topics as the standard lectures, but offers these in a less abstract way, by putting more emphasis on examples and illustration of the main ideas, rather than on the replication of proofs.

It was decided to write the whole package as a collection of hyperlinked Maple worksheets. This choice was directly motivated by the fact that our institute was about to acquire a Maple (at that time Release 4) campus license, enabling us to distribute the main 'carrier' software to all students, for use on all platforms. In addition it was felt that a computer algebra system like Maple was best suited to fulfill all our needs at the same time: nicely formatted text and graphics output could be combined with 'hands-on' open exercises and self-evaluation 'tests', the possibilities of which went far beyond those of traditional educational packages, where usually only a literal comparison is made between a student's answer and a collection of standard answers.

No matter what the quality of a selfstudy package is, the main motivation of a student for using it is directly proportional to the marks that are to be gained with it. Therefore monthly evaluation sessions were organized in a small computer-lab, with each session taking one and a half hour per student, enabling the students to obtain a maximum of 8 out of 20 marks. During the evaluations 3 to 4 exercises were to be solved, whereby the students could use whatever tools they deemed necessary (full access to the Maple library, the course modules and the lecture notes was permitted). At the end of the term a traditional examination followed: two theoretical questions and one more 'abstract' exercise made up for the 12 remaining marks.

The pilot group of students had no previous knowledge of Maple and also did not receive any introduction to Maple. Although in the selfstudy package the use of the necessary Maple commands was built up very gradually, some students complained about the big overhead of having to learn all the necessary Maple commands for this single course. The results nevertheless were very encouraging and the students reacted very positively to this mode of teaching. They 'liked' to do linear algebra in this way (sic!) and a comparison with examination results of previous years indicated a definite improvement. Most remarkable was the fact that the main group of first year students (who were not using the package at that time) appeared to obtain lower marks for the 'abstract' exercises (which were identical for both groups)!

These results stimulated us to introduce the package also to the full group of engineering students. Financial support to carry out this program, in a collaboration with the Free University of Brussels (Belgium) and The University of Birmingham (U.K.) was obtained and at the moment we have developed Maple V Release 5 and Maple 6 modules for linear algebra, theoretical mechanics and calculus. Each module consists of hyperlinked worksheets, divided into sections which begin with some explanatory text and with the main body consisting of worked out Maple examples and directed exercises.

The linear algebra modules cover the topics sets, relations and functions, groups, rings and fields, complex numbers, polynomial rings, vector spaces, matrices (elementary matrices, blockmatrices, normal forms, LU-factorisation), dot product (Gram-Schmidt process, inner product spaces), linear transformations, change of basis, unitary and orthogonal matrices, systems of linear equations, determinants, eigenvalues and eigenvectors, minimal polynomial, Cayley-Hamilton theorem, diagonable matrices, Schur's lemma, normal matrices, symmetric bilinear forms and quadratic forms.

The theoretical mechanics modules cover the topics vector calculus, curvilinear coordinates, speed and acceleration, moving frames of reference, intrinsic coordinates, relative kinematics, force and momentum, equations of motion, kinetic and potential energy, Lagrange formalism and small oscillations.

The calculus modules cover the topics real numbers, sequences, Cauchy's criteria, Bolzano - Weierstrass theorem, functions, limits and continuity, differential equations, scalar and vector fields, vector calculus, line integrals, surface integrals, volume integrals and the theorems of Green, Stokes and Gauss.

Up to the academic year 1999-2000 the selfstudy packages were being used by the main group on a voluntary basis, with about one third of the students claiming that they had made extensive use of the package. Recently it was decided to go one step further in the integration of selfstudy and computer algebra into the curriculum. From this year onwards about half of the exercise classes for linear algebra and theoretical mechanics and all calculus exercise classes are being conducted in PC-labs: these classes take place under staff-supervision and participation is meaningful only after having gone through the corresponding sections with worked-out exercises in the selfstudy modules.

As the usage of the PC-labs (three labs of 25 computers each) by the different departments of our faculty is growing fast, from this year onwards the first year student's Maple classes have to be restricted to two half days per week. Dealing with a group of 300 students and allowing for each student about two hours of Maple exercises per week, this forces us to seat two students per computer. Aside from the obvious advantage of halving herewith the necessary amount of staff time, we hope that this measure will result in educational advantages too. In fact the previous years have shown us that —with one student per computer— a students's lack of progress in an exercise class is often caused by rather elementary forms of 'computer illiteracy' and not by his or her lack of understanding of the course material. As a consequence a large amount of staff time during the PC-labs has been spend so far on helping students with the 'wrong' kind of difficulties.

The weekly sessions are also meant to make the student familiar with the procedures which will prevail at the final exam (4 PC-labs will then be used). The marking scheme for the exam will be identical to that of the pilot-group, with the Maple exercises counting for 40% of the total marks.

2 Maple as a tool for evaluation

2.1 Why a computer algebra system?

An essential part of any selfstudy package is a system for giving *feedback* to the student, for example through the automatic administering of graded tests. For exercises which require symbolic answers or in which the verification of the answers involves complex mathematical operations, the use of a computer algebra package is an indispensable tool. A typical such question in a linear algebra course is to compute a basis for the vector space of all trace-free 2x2 matrices. Both $\begin{pmatrix}1 & 0\\0 & -1\end{pmatrix}$, $\begin{pmatrix}0 & 1\\0 & 0\end{pmatrix}$, $\begin{pmatrix}0 & 0\\1 & 0\end{pmatrix}$ and $\begin{pmatrix}1 & 1\\1 & -1\end{pmatrix}$, $\begin{pmatrix}0 & 1\\-1 & 0\end{pmatrix}$, $\begin{pmatrix}0 & 1\\1 & 0\end{pmatrix}$, (as well as infinitely many other choices) are correct answers for this question. Grading this question first requires converting the matrices appearing in the student's answer to vectors v_1, v_2 and v_3. Next one should check whether the span of v_1, v_2, v_3 equals the span of the vectors u_1, u_2, u_3 that correspond to the matrices in the model answer. This last check involves the calculation of the rank of the matrix $[u_1, u_2, u_3]$ and of the augmented matrix $[u_1, u_2, u_3, v_1, v_2, v_3]$. Tasks like this can be easily accomplished by a computer algebra package, but are almost impossible to be carried out with standard authoring software. This is one of the reasons why most of the traditional assessment software is restricted to the use of multiple choice or multiple response questions, which —no matter how well designed— are not able to deal with relatively 'open' questions such as the one above.

Using a powerful computer algebra system such as Maple for the purpose of evaluation, creates however its own difficulties. The most typical one is that it becomes quite difficult to distinguish a correct answer (one which is mathematically equivalent to the teacher's model answer) from an answer which is nothing but a syntactically correct rephrasing of the question itself: when a drill and practice test on integration asks for the calculation of $\int \sin^2(x)\mathrm{d}x$, the Maple input `int(sin(x)^2,x)` definitely should not be regarded as correct (whether such drill and practice exercises still are meaningful, once one decides that computer algebra systems are to be introduced in the curriculum, is an all together different issue)! Care therefore has to be taken that the answers do not make accidental (or intentional) use of 'forbidden' Maple commands which would make the exercise obsolete.

An often heard objection to computer aided assessment is that it is unable to distinguish between two incorrect answers, where one is the result of a simple error of calculation and where the other is a —more serious— 'method error': both answers are awarded the same marks by the system. This is a serious issue, which generally is tackled by splitting up questions such that 'method marks' in one form or another can be given, or —in a similar vein— by restricting the quizzes to large collections of relatively simple questions, such that the impact of calculational errors becomes negligible. However when students are allowed to do their home- or exam-work with the aid of a computer algebra system, this objection no longer holds, as in a certain sense the majority of the errors will be just the 'method errors'! Any remaining types of errors can be dealt with by taking appropriate actions (see the discussion in paragraph 2.4).

In order to include the mentioned feedback possibilities in our modules for self-evaluation, we developed the Alice Interactive Mathematics (AIM) server, which is a fast and Maple-powered webserver with extensive facilities for question and quiz design.

AIM is implemented using Maple (90%) and Java servlets (10%). Most of the work is done by Maple – the Java servlets are used only to display the web pages that Maple generates and to pass the student's answers back to Maple.

The web pages that Maple generates also contain forms for students to fill in their answer which are then sent back to the Maple program. Maple then takes an appropriate action (for instance grade the answers and generate a model solution) and sends another web page back to the student through Java servlets.

Only one copy of Maple is running on the server during the entire process. Thus there is no license problem if more than one person uses the AIM server.

2.2 Question design

Each AIM test consists of a series of questions. The teacher uses a web interface to type in a question definition, which is nothing but a text file that uses a flag-based syntax to describe question statement, grade the student's answers, generate a model solution etc. A (very) trivial example would be given by the following lines:

```
# A question on addition.
t> What is 2+2?
a> 2+2
end>
```

whereby the flags have the following meaning:

`#` (*comment*) Maple's standard comment line

`t>` (*type verbatim*) print the statement of the question verbatim using Maple's `printf` command

`a>` (*answer*) the expected answer can be any Maple expression and will be evaluated by Maple before being returned to the user

`end>` flag indicating the end of the file; usually a quiz file contains more than one question

Upon execution, Maple generates a web form presenting to the student the question "What is $2+2$?", next reads in the student's answer and compares it to its own answer, which is obtained by evaluating the line following the `a>` flag.

A more complicated calculus question on integration might for example be constructed as follows:

```
h> F :=`*`(op(combinat[randcomb]([exp(-x),sin(2*x),cos(5*x)],2)));
t> Evaluate the following integral:
p> Int(F,x)
s> [y->`aim/Testzero`(diff(y, x) - F), int(F, x)]
end>
```

whereby the `h>`, `p>` and `s>` flags have the following effects:

`h>` (*hidden command*) construct a random product of two factors out of the given list

`p>` (*prettyprint*) use Maple's prettyprinter to display math formulas in a nice two-dimensional format; this flag is especially useful for displaying matrices or integrals

`s>` (*special*) use the first element of the given list, namely a Maple procedure of the type $y \to f(y)$, to evaluate the student's answer (here: differentiate the answer, subtract it from the integrand (`F`) and check whether the result is zero, by applying to it the procedure ‘aim/Testzero‘); also return the model answer, which is given by the second element of the list.

After all the question definitions have been compiled by Maple they are added to a question database from which the actual tests will be built. A typical test of about 10 questions —provided it contains no on-the-fly generation of graphics— takes about 0.2 seconds to be generated (of course, depending on the load of the network, it takes a couple of extra seconds before the resulting page appears on the student's screen). Questions can include Greek symbols (these are first converted into escape sequences by Maple and then converted into HTML Greek letters by the Java servlet), subscripts, superscripts and two-dimensional Maple output (such as fractions, matrices or integrals). The full description of the syntax is available on the web (see `http://allserv.rug.ac.be/~fkolokol/aim/docs`).

2.3 Quiz design

To discourage cheating among students, AIM maintains a large database from which questions can be chosen at random using customizable selection criteria, based on topic and difficulty level. In addition Maple's random generator is used to provide additional variation to most questions, by randomly varying parameters. As a result, each student receives after logging in to the server a unique, personalized set of questions which is based on their student-ID. As long as the random generator receives the same student-ID as seed, the resulting question sets will always be identical. This gives students then the opportunity to inspect their questions, log off from the system, do their homework and come back a couple of days later to send in their answers. Of course, for pure self-evaluation purposes students can at any time log in anonymously to the system and receive for each topic of their choice a set of questions, which, in the absence of a seed, will be completely randomized.

2.4 Grading

After the answers have been submitted to the server, they are evaluated by Maple. For each wrong answer the student is given the opportunity to redo the question, but is given a penalty (of 10% by default). When the student answers with a syntactically incorrect Maple statement, a simple warning is issued and no penalisation issues. This method of automatically giving partial marks has several advantages and appears to be well appreciated by the students — despite the penalisation feature: it forces them to be careful, it encourages them to verify their answers before submitting them and students know immediately after completion of the test if they have made a mistake and they don't have to wait for days, as is often the case with traditional homework.

AIM allows other possibilities for giving partial marks too: custom grading procedures and multiple response questions. A custom grading procedure is a Maple procedure written by the teacher. It accepts the student's answer as an argument and returns either true (meaning correct answer) or false (meaning incorrect answer) or else a number between 0 and 1 which represents the student's grade on that question. Grading procedures can actually warn the student and give hints (or give an extra penalization), depending on the form of the answer.

To facilitate question creation, AIM supports five pre-defined question types: algebraic, matrix, multiple choice, multi-choice, and constant. In addition, any of the existing Maple types can be used as well. For example, a question that asks to compute the normal at some point to some surface will have a Maple type [constant, constant, constant]. The five predefined types come with a different grading procedure as described below.

By default a question has type 'algebraic'. In this case the grading procedure consists of checking the mathematical equivalence between the student's answer and the model answer, by subtracting one from the other and applying Maple's extensive machinery to test if the result is 0. When the question is of type 'matrix', the same procedure is applied component-wise. The student receives full marks or zero, depending on whether the results all equal zero or not.

For multiple choice questions, the students are presented with a menu from which they must choose a single answer. The standard procedure for multiple choice is the natural one - the student receiving full marks or zero, depending on whether the student's selection corresponds to the teacher's selection. As with the other types, the teacher can also design a custom grading procedure which can assign partial marks to some of the choices.

For a multiple response question, the students must click on all of the correct answers from a given list of correct and incorrect answers.

For an answer of type 'constant', the grading procedure consists of verifying numerically if the student's and teacher's answers are within 5% of each-other.

In addition to the pre-defined AIM types, any Maple type can also be used when designing a question. In this case the student's answer is checked for compatibility with the type specified. By default any mismatch results in a warning, without a penalty for the student. In order to deal with those circumstances where a specific mismatch hints at a basic misunderstanding on the student's part, the teacher can also opt for a custom grading procedure. In that case the Maple procedure for evaluation of the answer will take a penalizing action when one or more specific type mismatches occur (as for example could be the case in an exercise where a student mixes up vectorial and scalar quantities incorrectly).

2.5 Administration details

For the teacher AIM provides extensive facilities for administering questions and quizzes. All of these facilities are available through a web-interface and are password-protected.

AIM provides four different tools to monitor the student's progress: grade reports, aggregate statistics, log files and the 'spy' ability to inspect and modify a student's answer. In addition students may communicate with the teacher by email, through a link conveniently placed on every quiz.

The log and grade reports are updated every time the student modifies an answer. The teacher has complete control over how the grades for each quiz are assigned, as well as how the overall grade is computed. The 'spy' feature allowing the teacher to inspect and modify a student's questions and/or answers is useful to react quickly to any reported bugs in the questions. It's presence is also indispensable, because, even though the use of a computer algebra system virtually rules out the possibility that a student receives zero marks as a consequence of an elementary error of calculation, there still remains the possibility that a somewhat distracted student solves an altogether different exercise correctly and receives zero marks. There is no way in which a computer programme can deal with this kind of error (which of course frequently occurs also with paper and pencil work) and this case should be dealt with by the teacher on an individual basis.

For administering homeworks a due date can be set up by the teacher. Up to this due date the students can enter their answers and modify them, but they are not able to obtain the correct results from the computer. After the due date their answers cannot any longer be modified and the model answers become available.

AIM is currently being used at two different universities and one college in Belgium. The AIM server can be installed on any Windows computer that has Maple and a net connection. It is freely available by writing to the authors.

3 References

1. Alice Interactive Mathematics (AIM) http://allserv.rug.ac.be:8081/aim/docs
2. Calculus on the Web (COW) http://fourier.math.temple.edu
3. Heriot-Watt WebTest http://webtest.ltc.hw.ac.uk
4. WebWorks http://webwork.math.rochester.edu/docs/
5. Web Hosted Assessment of Mathematics (WHAM) http://l62.csm.port.ac.uk/