"Using Maple to Grade Maple"
Assessment Software
from
North Carolina State University

Erich Kaltofen[1], Michael McLean[2], and Larry Norris[3]

Department of Mathematics, North Carolina State University, Raleigh, NC 27695-8205

with

Dmitriy Morozov[4], John May[5] and William Turner[6]

Department of Mathematics, North Carolina State University, Raleigh, NC 27695-8205

<u>Abstract</u>

We present an overview of the Maple Homework Assessment and Tutorial System developed in the Department of Mathematics at North Carolina State University. We have developed a suite of programs that manipulates worksheets, grades worksheets, provides interactive tutorials, and stores copies of graded materials and grades. The system "Uses Maple to grade Maple", having as its core a special "grader package" that exports various procedures that are used to analyze and grade Maple worksheets. We ask students to work problems written in preformatted Maple worksheets, and then to assign their answers to carefully chosen names. The values assigned to names such as ans1, my_plot and my_third_answer may be the output of any Maple computation. The eGrader is thus limited only by the limits of Maple itself. During the fall 2001 and spring 2002 semesters we stress-tested the system, grading over 130,000 problems for all sections of the engineering calculus courses.

---

[1]kaltofen@math.ncsu.edu

[2]mike@darkcanvas.com

[3]lkn@math.ncsu.edu

[4]foxcub@foxcub.com

[5]jpmay@unity.ncsu.edu

[6]wjturner@unity.ncsu.edu

1 Introduction

Each semester at North Carolina State University (NCSU) the Department of Mathematics offers approximately 50 sections of the engineering calculus sequence of courses MA 141, MA 241 and MA 242. Those 50 sections are populated by approximately 3000 students, mostly freshmen and sophomores, and since 1994 the curriculum of each of these courses has had a Maple [1] component. In the initial program there were no formal Maple labs, and students turned in printed copies of Maple worksheets that were graded by hand. In the fall of 1999 the Mathematics Department began a revised program that had (1) formal labs, (2) lessons for each lab written in a Maple worksheet, and (3) formal homeworks for each lesson in the form of preformatted problems in a Maple worksheet. The problems were all standard calculus textbook problems. The lessons and homework were obtained by the students via a web browser, and the homework was submitted via the internet and stored in a secure locker in the NCSU Unity network. The homework worksheets were then opened on 1/2 of the screen of a workstation, and each student's work was then compared to a key, open on the other 1/2 of the screen, by an undergraduate student hired to grade the homework. Although technologically more advanced than the grading of printed worksheets, the process of judging correct and incorrect answers had not changed.

This last step in the homework cycle, namely the human grading of the worksheets, was the least successful aspect of the entire program. Not only was the grading often inconsistent, the turn-around time from due date to return of the graded worksheets was often more than 10 days. In an effort to eliminate these unpleasant features of our Maple program, we began a search during the spring 2000 semester for a system that might provide us with a computer-based grading system to grade the Maple homework. We first considered the NCSU based WebAssign [2] facility developed in the Department of Physics at NCSU. We quickly realized that WebAssign would be inadequate for our purposes if we wished to retain the Maple worksheet as the medium for student homework assignments. The variety of answers that WebAssign could grade (in spring, 2000) was very limited, and certainly was not capable of grading a Maple worksheet. Hence, if we wanted to grade "real Maple" then WebAssign would not do. We also looked at Web-CT [3] but it also had only limited grading ability and was also not suitable for grading Maple worksheets. So we decided to build the system ourselves, and in spring 2000 we started the eGrader project to develop an automated grading system [4]. What has emerged from two years of work is a wonderful system that

- eliminates the human factor from the grading process and reduces the turn around time to a matter of hours, practically, and to a few seconds in principle,

- encourages students to learn and use proper Maple syntax

- includes the eTutor, which provides interactive practice Maple homework sessions for the students.

In the rest of this paper we will present an overview of the eGrader/eTutor project that has been developed over the last two years at NCSU.

2 Overview of the system

The Maple Program for the engineering calculus sequence or courses at NCSU is composed of the following parts:

A Lessons and Homework Archive Contains lessons written in Maple worksheets, that address various topics in calculus. The lessons are also available in html format. Associated with each lesson is a homework assignment in the form of a preformatted Maple worksheet, with each problem in a separate section in the worksheet. The lessons and homework worksheets are available from the Calculus with Maple Homepage at http://www.math.ncsu.edu/calculus. We have the ability to generate problems with random elements (numerical, expressions, functions, etc) at appropriate places in each problem, thereby giving us the ability to quickly generate a large data base of new problems.

A Homework Repository Students submit their homework via a browser to the NCSU Wolfware [5] repository where the files, the graded files, and grade records are all stored in a secure location. Students must login using their NCSU Unity ID.

The eGrader A collection of Maple library functions and auxiliary applications running in a unix/Linux environment that processes the grading of the homework worksheets. A typical call to the eGrader is

grade ma ma141 005 Homework2 "*"

which starts the grading of homework assignment #2 for section 005 of MA 141. The last option "*" indicates that all papers in the section should be graded. When eGrader grades an individual worksheet, it stores the original file, writes the graded worksheet to a new graded directory, and records the grade in a grade file. For each problem, eGrader returns three subsections in the graded worksheet:

- **Your work:** This section contains a verbatim copy of the student's work on the problem.
- **Grader Comments:** The total score is reported here, plus individual scores for each answer in the problem, and comments related to the partial credit assignments.
- **Sample Solution:** This subsection has a worked out solution to the problem.

The eTutor (Under Construction) A collection of interactive sessions designed to provide the student with practice on problems of the sort that will be encountered in the homework. The eTutor is based on the concept that each partial credit assignment made in the grading code of a problem addresses a specific anticipated mistake, and so a variant of the grading code should be able to supply the student "advice" on how to correct that mistake when it is made when working on a problem in a practice worksheet. We have a number of sample interactive sessions. When the student enters his/her name at the top of the worksheet, the worksheet will automatically launch a second Maple application that contains the eTutorial grading code. These two individual Maple sessions then communicate via the sockets facility available starting with Maple 7.

The ePublisher (Under Construction) A collection of web based tools designed to produce either (1) a new problem for a future homework assignment, (2) a homework assignment, or (3) an interactive eTutorial. After writing the grading code by hand for the first data base of homework problems, we surveyed the grading code to find high-level grading procedures, and are now constructing GUIs to assist the user to automatically generate most, if not all, of the grading code for a large class of problems. Each interface will allow for partial credit assignments and "comments" that will be issued either by the eGrader in the graded worksheet, or by the eTutor in a practice homework problem. At the moment we have prototype GUIs for problems with (a) numerical answers, (b) expression answers and (c) answers that involve the plot command. We anticipate the construction of GUIs for many other types of answers.

The Grades Handler We keep track of the grades and distribute grade files to the instructors in the courses.

3 A Survey of the Grading Package

There were two major challenges in setting up the system. The first was the problem of dealing with the fact that in Maple 6 the worksheet was not a data structure. Mike McLean spent two weeks in July, 2000, working at WMI figuring out how to work with Maple worksheets. The end result was a Maple worksheet package to manipulate worksheets. The second major challenge in constructing the grading package was to build a stable system that would not shut down when it encounters one of the many Maple errors generated, for example, when improper syntax is used by a student in the worksheet. The procedures that are exported by the grader package and that are used in the grading code all are formulated to trap exceptions using Maple's exception handling mechanism using the try..catch statements. Below we list and give a brief description of some of the procedures used in the grading code. Keep in mind that many but not all of these procedures were motivated by the subject, namely calculus, for which the program was initially designed. However, new procedures can be added to the grader package as new problem types arise when the system is used in conjunction with other mathematics courses, and indeed, other NCSU courses in the future.

1. **expcheck** A procedure to determine, without error, whether or not two expressions are equivalent. For example,

$$\text{expcheck(sin(2x),2*cos(x)*sin(x))}$$

returns "true".

2. **is2dplot** Check to determine if a name is assigned to the data structure of a 2 dimensional plot.

3. **numcheck** A procedure to determine, without error, whether or not two expressions of type numeric are equivalent. A tolerance is allowed as an option. For example,

$$\text{numcheck(ans1,976.2257,tolerance=1e-5)}$$

will return "true" if ans1 differs from 976.2257 by less than 10^{-5}.

4. **plotdist** A procedure that determines the "distance" between 2 two dimensional plots. This function and several others follow the "grade the output" approach, which we are going to change for plot structures.

5. **rangecheck** A procedure to determine the range(s) of a data structure of type plot.

6. **scorekeeper** The module that keeps track of partial credit assignments, totals, and comments to be included for the student with the graded worksheet.

7. **searchplot** A procedure designed to determine if a data structure contains the plot of a given function.

These procedures were included in the original set of procedures in our Maple eGrader package, but they have since been replaced with higher-level functions that are easier to use and more versatile. For example we now use **numsetgrade** and **expsetgrade**, two procedures that compare a set of student numbers or expressions, respectively, to a set of anticipated answers. Although the commands listed above all grade the output of Maple, we are changing the analysis of problems involving plots and are now grading the Maple input on such problems.

3.1 A sample from our data base of problems

Here is a sample Maple worksheet with the grading code for a problem from the 3rd MA 141 homework problem set. This worksheet is used to both include this problem as one of the problems in a homework worksheet or eTutorial, and to supply the grading code at grading time. In this problem the student is asked to assign to the name ans1 the plot of the functions $10 * x^2$ and $x^3/10$, and to later do some comparisons of the functions based on this plot. In the code _score is the shorthand name for the scorekeeper module, and the exported procedure _score:-part( ) is the partial credit assignment procedure. The worksheet is divided into three sections:

- **The problem:** Used to insert the text of the problem into a Homework worksheet.

- **The Solution:** Used to supply the "Sample Solution" in the graded worksheet that is returned to the student.

- **Grading:** Supplies the grading code when eGrader is called to grade this problem in a Homework worksheet.

    -   **The Problem**

Use graphs to determine which of the functions $f(x) = 10\, x^2$ or $g(x) = \frac{x^3}{10}$ is eventually larger (that is, larger when x is positive and very large). To do this, graph both functions on one set of axes using an appropriate domain. Make each function a different color so that you know which is which.

Assign your **plot showing both functions**, to the name `ans1`.

Assign to `ans2` the function, as either a function or an expression in x, that is eventually larger.

```
> ans1 := your_plot_command_here:  %;
```

```
> ans2 := your_answer_here  ;
```

- **The Solution**
```
> f := 10*x^2;
```
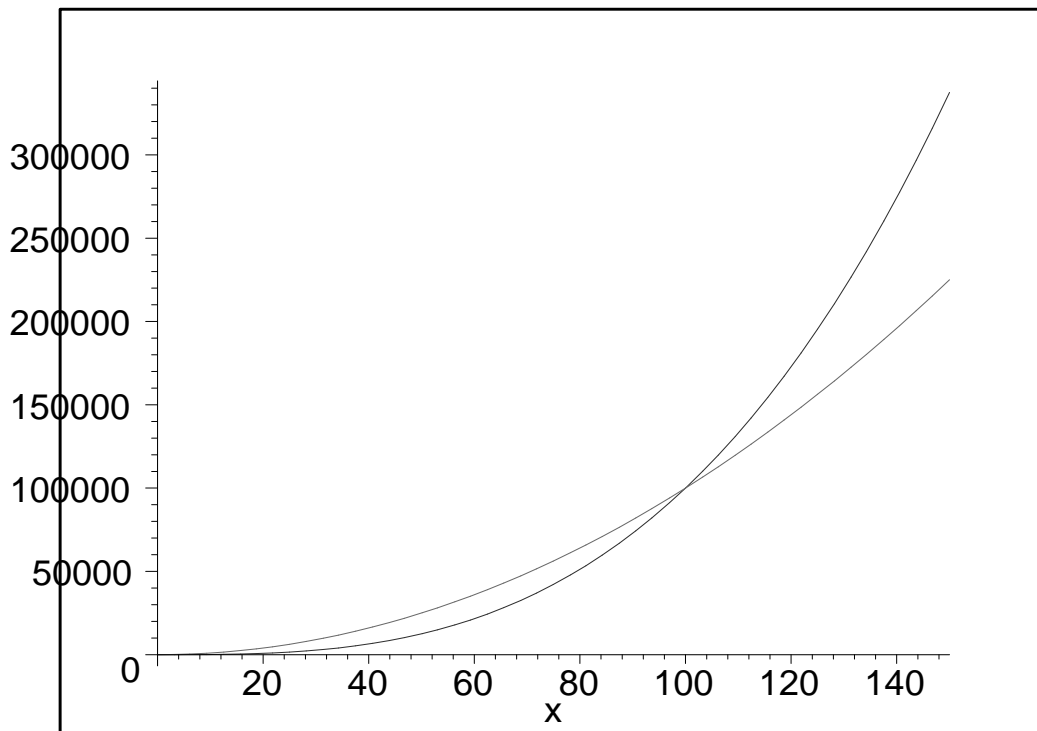```
> g := x^3/10;
```

$$f := 10\,x^2$$

$$g := \frac{1}{10}\,x^3$$

```
> solve(f=g,x);
```

$$0,\ 0,\ 100$$

x=100 is the last point of intersection, so whichever function is greater after that point will remain greater.

```
> ans1 := plot([f,g],x=0..150,color=[magenta,navy]):  %;
```


And we see that g is eventually larger.

```
> ans2 := g;
```

$$ans2 := \frac{1}{10}\,x^3$$

5

- Grading

```
>   with(grader):
>   _f := x -> 10*x^2;
>   _g := x -> x^3/10;
>   if not is_2dplot(ans1) then
>       _score:-part(0,3, "did not assign two dimensional plot to ans1");
>   else
>       _f_comp := searchplot(ans1, function=_f);
>       _g_comp := searchplot(ans1, function=_g);
>       _x1 := -infinity;
>       _x2 := infinity;
>       if nops(_f_comp) = 0 then
>           _score:-part(0,1,"did not plot f");
>       else
>           _score:-part(1,1);
>           _f_range := plotrange(ans1, [op(_f_comp)], 'total');
>           _x1 := lhs(op(1, _f_range));
>           _x2 := rhs(op(1, _f_range));
>       end if;
>       if nops(_g_comp) = 0 then
>           _score:-part(0,1,"did not plot g");
>       else
>           _score:-part(1,1);
>           _g_range := plotrange(ans1, [op(_g_comp)], 'total');
>           _x1 := max(_x1, lhs(op(1, _g_range)));
>           _x2 := min(_x2, rhs(op(1, _g_range)));
>       end if;
>       if _x1 < 100 and _x2 > 100 then
>           _score:-part(1,1);
>       else
>           _score:-part(0,1, "domain does not include point of
>   intersection");
>       end if;
>   end if;

>   if expcheck(ans2, _g(x)) or expcheck(ans2(x), _g(x)) then
>       _score:-part(1,1);
>   else
>       _score:-part(0,1, "did not find correct function");
>   end if;
```

This grading code was written by hand two years ago, but today we would write most of this grading code automatically using the ePublisher.

4   The eTutor, or "What we have learned about the mistakes students make."

The eGrader package was tested on two sections of third semester calculus classes during the second summer session, 2001 at NCSU, and then stress-tested during the fall 2001 and spring 2002 semesters. Upon examining graded worksheets in fall 2001 we immediately began to put together a set of common errors for particular classes of problems. For example, instructions may tell the student "...to assign to A the number 234", and a small group of students will enter

```
>   a:= 234;
```

Since the initial codes allowed for very few partial credit assignments, students were loosing full credit for this type of mistake. We added a few lines in the code to anticipate this mistake, and subtracted 25% credit. If students complained about loosing points for this mistake, we replied that "this is not just calculus, it is calculus with Maple, and you must use proper syntax to get full credit." It is my impression that students have accepted this explanation. When extreme persuasion is required (as it is with some aggressive undergraduates who write their complaints to Maple_Grader@math.ncsu.edu) we point out that "you must use upper case "I" and not lower case "i" to represent the imaginary unit in Maple. Moreover Maple will not evaluate pi to the number you expect it to, while Maple is quite happy to evaluate Pi to 3.14159.... . You loose points if you don't use proper syntax." A positive side effect of using the eGrader is that it forces students to use proper syntax in their work.

When the partial credit is assigned for this problem, one might write, using the partial credit procedure _score:-part( ),

_score:-part(.75,1,"You were asked to assign the numer 234 to capital A, not lower case a");

and this comment would appear in the "Grader Comments" section of the graded worksheet. But it should be clear that if this same response could be reported to the student immediately after the student presses the enter key in the worksheet, the student could correct the mistake and then move on. This is the idea behind the eTutor - *to use the written comments about partial credit to alert students to their errors when they make them.* We have constructed a number of sample interactive practive worksheets using this idea, and we plan to make such eTutorials a central feature of the program in the future. The basic reason, beyond the obvious benefits just described, is that students seem unwilling to read lessons written in Maple worksheet format, no matter how well-written. The new generation of students seem to need "instant feedback", and the eTutor will in fact supply such instant feedback about mistakes. Our second generation of problems are now formatted to serve as templates for either (1) homework problems, or (2) eTutorials.

4.1   How do we anticipate mistakes?

How should we decide what partial credit assignments we should allow in order that the eTutor be focused on what students actually do? We need a "focus group", and in fact we have the data from a very large focus group, namely the complete set of homework problems submitted during the fall 2001 and spring 2002 semesters by all students in the three calculus courses. We have added a few lines to the grading code for each problem that writes any new mistakes to a record file. Here a "new mistake" is any mistake in a problem for which there is no partial credit assignment. During summer 2002 we are analyzing these "new mistakes" in order to incorporate them back into the grading code. The eTutor will thus be an ever-improving expert system. It is intriguing to wonder how many semesters it will take to capture, say 90% of the "typical calculus errors". Can we reach 95% or more?

5   Conclusions

The eGrader/eTutor system from the Mathematics Department at NCSU is a system designed to grade Maple worksheets. It should be emphasized that the system is very flexible and in principle can grade anything that Maple can compute. Thus, although the system was originally designed to be used in calculus courses, the system can be used in any course in which Maple can be used for homework assignments. We are presently designing an application for use in a junior level Electrical and Computer Engineering course

at NCSU. The system has also been used in an elementary differential equations class at NCSU, and we plan to eventually offer the system to the entire NCSU community.

References

[1] Waterloo Maple, Inc., http://www.maplesoft.com

[2] WebAssign, http://www.WebAssign.com

[3] WebCT, http://www.webct.com

[4] See also Alice Interactive Mathematics (AIM) http://calculus.rug.ac.be:8080 for another system that uses Maple to grade. AIM uses Maple's symbolic capabilities to grade vector calculus tests.

[5] Wolfware, http://wolfware.ncsu.edu