ADVANCED RANDOMIZATION AND GRADING IN THE E-LEARNING SYSTEM MAPLE T.A.

Andreas Zimmermann, Vilma Urbonaite, Andreas Körner, Stefanie Winkler, Stefan Krause, Maximilian Kleinert

Vienna University of Technology, Institute for Analysis and Scientific Computing
1040 Vienna, Wiedner Hauptstraße 8-10, Austria

andreas.zimmermann@tuwien.ac.at(Andreas Zimmermann)

Abstract

In this paper we introduce an enhanced use of the web-based e-learning system Maple T.A, that is distributed by the company MapleSoft. At the Vienna University of Technology the software is additionally attached to common mathematical lectures and provides some useful practice and assignment capabilities for students. Main advantages of the system are the abilities to create mathematical exercises with randomized specifications and to automatically verify the correctness of students' responses. For both tasks, the randomization and the grading, Maple T.A. provides certain functions. With increasing complexity of the exercises these common methods are not sufficient. Especially more complex mathematical objects like vectors, matrices, etc. showed to be difficult to handle, since there are no standard functions implemented.
For this purpose Maple T.A. allows to integrate Maple code, that assumes the randomization and grading of mathematical objects of these types. The generation of these Maple algorithms is not always an easy task, on the other hand it didn't turn out to be necessary to develop each algorithm over and over again because some of the problems equal each other to a certain degree. Within this paper we describe methods and algorithms that build the basis of additional moduls, that were created for simplification of above mentioned tasks and give some concrete practical examples, as they are used in mathematical lectures.

Keywords: Maple T.A., Maple, e-learning, blended learning

Presenting Author's Biography

Andreas Zimmermann is a student of Technical Mathematics at the Vienna University of Technology and will soon start his diploma thesis in the field of Soft Computing. During the last few years he also was a member of a group that developed and evaluated e-Learning systems for education in mathematics and modeling and simulation at the Vienna University of Technology.

1 Introduction

At the Vienna University of Technology the web-based e-learning Maple T.A. is used to assist the assessment of students' mathematical skills. It proved to be a useful addition to the common way of teaching. Former studies [1, 2] showed that with the possibility to practice their mathematical skills, students were able to deepen their knowledge in different mathematical disciplines outside of the lectures. Due to the randomization abilities of Maple T.A. students always get randomized specifications of questions and exercises, and so they can practice the same tasks over and over again, but with differing parameters. Further the software additonally provides an assignment and a gradebook module which are helpful tools to perform assignments for students' skills.

During the last two years of usage of Maple T.A. at the Vienna University of Technology a large amount of questions and exercises were created for different mathematical topics. But with increasing complexity of the topics the creation of questions became more and more difficult. Where for the beginner's lectures the common functions of Maple T.A. satisfied the demands adequately, difficulties arose in continuative lectures. In particular the randomization and grading of more complex mathematical objects (e.g. matrices, vectors,...) with or without special properties were challenges that had to be overcome. For this purpose two additional Maple libraries were developed and added to the Maple core of the Maple T.A. system. One of them is responsible for the randomization of these mathematical objects and the other one aims to simplify the grading process for them.

2 Randomization Library

The common Maple T.A. functions only support simple randomization of integers or real numbers. To get a greater variety of randomized objects, such as matrices, vectors, etc., various Maple routines can be used [3]. But the more complex the objects get, the more attributes the objects should have, the more complicated the creation of questions gets. To make this process less difficult an additional Maple randomization module was developed. It allows to generate various types of mathematical objects:

2.1 Function Definitions

Function FromSet.
Random elements from a set are selected. The data types of the set elements are not restricted. According to the parameters set the return value of the function is either a vector or a single value.

Parameters:

- *Set* - set with selectable values
- *Count* - number of elements to be selected
- *distinct* - with or without replacement
- *sorted* - determines whether the return vector is sorted or not

Function MakeInts/FromInts.
For given interval boundaries the function returns a sequence of either a specified number or all integers between these boundaries.

Parameters:

- *Min* - lower boundary of the interval
- *Max* - upper boundary of the interval
- *Count* - number of desired random values

Function MakeRats/FromRats.
The function is similar to MakeInts/FromInts. The return values are here rational numbers, that are restricted by some additional parameters for the denominators.

Paramters:

- *Min* - lower boundary of the interval
- *Max* - upper boundary of the interval
- *Count* - number of desired random values
- *MinDenom* - smallest possible denominator
- *MaxDenom* - largest possible denominator

Function Vec.
For a given dimension this function returns a random vector with user-defined entries.

Paramters:

- *Set* - set of numbers for random vector entries
- *Dim* - desired dimension of the vector
- *zerocount* - number of zeros in the vector

Function VecInts.
A random vector with integer entries is created and returned.

Paramters:

- *Dim* - dimension of the return vector
- *Max* - for each entry k: $|k| \leq Max$
- *zerocount* - number of zeros in the vector

Function Mat.
The Mat-function generates random matrices. It is possible to define the matrix dimension as well as a desired shape.

Paramters:

- *Set* - set of numbers for matrix entries
- *rows* - number of matrix rows
- *cols* - number of matrix columns
- *zerocount* - number of entries equal zero
- *shape* - determines the matrix shape (e.g. triangular, symmetric, etc.)

Function MatInts.
Just like the vector functions, MatInts creates matrices only with integer entries.

Paramters:

- *Max* - absolute values of entries are $\leq$ *Max*
- *rank* - desired rank of the matrix
- *rows* - number of matrix rows
- *cols* - number of matrix columns
- *zerocount* - number of matrix entries equal zero

Function MatIntsDef.
This function delivers a random symmetric matrix with integer entries and specifed numbers of positive (*pos*), negative (*neg*) and eigenvalues equal zero (*zero*). The dimension of the matrix is defined by: $dim = pos + neg + zero$, and the rank equals $pos + neg$.

Paramters:

- *Max* - absolute values of entries are ≤ *Max*
- *pos* - number of positive eigenvalues
- *neg* - number of negative eigenvalues
- *zero* - number of eigenvalues equal zero
- *zerocount* - number of entries equal zero

2.2 Algorithm Example

To demonstrate the functionality of one randomization algorithm, we give a short example for the creation of a random $m \times n$ matrix. Consider the following function call:
`MatInts(Max=4,rows=7,cols=5,rank=3)`
We start with creating a matrix that obviously has the desired rank r. Therefore we choose r random integer values within the given range and write them in the "diagonal" of the first r rows.

$$\begin{pmatrix} -4 & 0 & 0 & 0 & 0 \\ 0 & -4 & 0 & 0 & 0 \\ 0 & 0 & -4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Then we progressively fill the first r rows with random integer entries within the given range. When doing that, we have to make sure that we do not reduce the rank of the matrix. It can be shown, that for each selected entry there is at most one number $\alpha \in \mathbb{Q}$, that makes the matrix rows linear dependent. So if α lies in the given range, we have to select another value.

$$\begin{pmatrix} -4 & -4 & 4 & 1 & -3 \\ 0 & -4 & -2 & 0 & -4 \\ 2 & 4 & -4 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Now that we have a matrix with random entries and the desired rank we proceed with finding $m - r$ linear dependent vectors with integer entries within the given range and replace the last $m - r$ rows. For this purpose we first transpose the matrix and build a base of the span of the first r columns, which is easier to handle:

$$B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \frac{3}{2} & -\frac{7}{12} & \frac{7}{6} \\ 2 & \frac{1}{4} & \frac{3}{2} \end{pmatrix}$$

This transformation ensures that each linear combination of the columns with integer linear coefficients $|a_i| \leq \texttt{Max}$ has correct entries in the first r components. After that we calculate the common denominator (q) of all entries and multiply the last $m - r$ rows with q:

$$q = 12, P = \begin{pmatrix} 18 & -7 & 14 \\ 24 & 3 & 18 \end{pmatrix}$$

We apply Gauss elimination modulo q:

$$Q = \begin{pmatrix} 6 & 5 & 2 \\ 0 & 3 & 6 \end{pmatrix}$$

At last we randomly choose a vector
$v \in \{-\texttt{Max}, \dots, \texttt{Max}\}^r$ such that $Qv = 0$ and the absolute values of all entries of the vector $w = Bv$ are less or equal `Max`, for example:

$$v = \begin{pmatrix} 2 \\ -4 \\ -2 \end{pmatrix}, w = \begin{pmatrix} 2 \\ -4 \\ -2 \\ 3 \\ 0 \end{pmatrix}$$

We transpose w and insert it into the matrix:

$$\begin{pmatrix} -4 & -4 & 4 & 1 & -3 \\ 0 & -4 & -2 & 0 & -4 \\ 2 & 4 & -4 & -4 & 1 \\ 2 & -4 & -2 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

After repeating the last step for the remaining rows and a final permutation of rows and columns we get the resulting matrix of our algorithm:

$$\begin{pmatrix} -2 & -1 & 3 & 2 & 2 \\ -4 & -2 & 3 & 0 & 2 \\ 2 & 1 & -3 & -2 & -2 \\ -4 & -2 & 3 & 0 & 2 \\ -4 & 4 & 1 & -3 & -4 \\ -4 & -2 & 0 & -4 & 0 \\ 4 & -4 & -4 & -1 & -2 \end{pmatrix}$$

This function is also used when generating random definite matrices. Consider the function call:
`MatIntsDef(M,pos,neg,zero)`. So the aim of the algorithm is to get a matrix with `pos` positive eigenvalues, `neg` negative eigenvalues, and eigenvalue zero with mulitplicity `zero`.Therefore we calculate a random matrix $B \in \mathbb{Z}^{(\texttt{pos+neg}) \times (\texttt{pos+neg+zero})}$ with the rank `pos+neg` using the `MatInts` function. Let then the first `pos` rows of B be B^+ and the last `neg` rows of B be B^-.
We determine the matrix $A = (B^+)^T B^+ - (B^-)^T B^-$. Regarding to Sylvester's law of inertia [4] A has `pos` positive, `neg` negative eigenvalues and eigenvalue zero with multiplicity `zero`. This algorithm is repeated until the absolute value of all entries of A are less or equal `M`.

3 Grading Library

Automatic grading of answers is not always an easy task. It is necessary to have well defined rules when evaluating the correctness of entered sets, vectors or matrices. For this purpose Maple T.A. provides the Maple-Graded question type, that connects to the Maple engine when grading student's responses. The question designers have to write some Maple code that is responsible for the handling of the answers. According to the complexity of the question this task can be quite tricky and may need long development time. With the additional Maple grading library, that takes care of most of these problems, this time can be reduced. Following function are supported:

Function Expr.
This function is for common mathematical expressions. They are checked for mathematical equivalence to the given correct answer. For this object type no partial grading is possible. So the student's response is either completely correct or wrong.

Function ExprDiff.
ExprDiff is used when grading antiderivatives. The response is derivated and then compared to the original specification of the question.

Function Set.
For sets partial grading seems reasonable. Students may enter subsets or supersets of the correct ones. To handle this issue a return value is calculated that lies between 0 and 1, depending on how correct the student's answer is. Thus, sets are graded as follows:

$$\text{rv} = \begin{cases} |R \cap C|/|C| & \text{if } |R| \leq |C| > 0, \\ \max\{0, 1 - \frac{|R \setminus C|}{|C|}\} & \text{if } |R| > |C| > 0, \\ 1 & \text{if } |R| = |C| = 0, \\ 0 & \text{if } |R| > |C| = 0 \end{cases}$$

where R denotes the response set and C the correct set for the question.

Function Vec.
The Vector grading function returns 0, if the entered vector and the correct one have different lengths. Otherwise the return value is determined by dividing the correct entries by the total number of vector entries.

Function Mat.
Evaluating the correctness of matrices follows the same way as for vectors. If the response matrix and the correct matrix have different sizes the entered matrix is graded with 0. In all other cases the return value is again the correct number of entries divided by the total number of entries.

Function FundSys.
This function is used for grading real fundamental systems, that define the sets of solutions of systems of homogeneous linear differential equations. The order of the differential equation determines the cardinality of the set. So the return value is calculated similar to the one of the set-grading function: The order of the differential equation is equal $|C|$, $|R \cap C|$ corresponds to the greatest number of linear independent real solutions in the response and $|R \setminus C|$ is equal to the smallest number of functions, that have to be removed from the response set, to get a set of linear independent real solutions.

4 Question Development Examples

4.1 Matrix invert

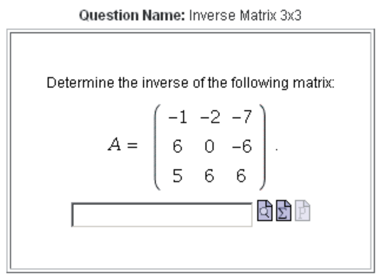

Fig. 1 *example specification for matrix inverse*

We consider an example of finding the inverse of a 3×3 matrix A. (fig. 1). In Maple T.A. there is no command for creating a random matrix. Maple itself provides the `RandomMatrix` function of the `LinearAlgebra` package. But here it is not possible to automatically create a regular matrix. So we have to write some kind of algorithm in Maple T.A. to get what we desire, e.g.:

```
$seed1=range(1,1000);
$seed2=range(1,1000);
$A = maple("
  det := 0:
  seed1 := ($seed1):
  seed2 := ($seed2):
  while det = 0 do
    randomize(seed1):
    A := LinearAlgebra[RandomMatrix]
      (3,3,generator=-9..9):
    det := LinearAlgebra[Determinant](A):
    seed1 := seed1+seed2:
  end do;
  A");
```

We calculate random matrices as long as we get a matrix determinant that is greater than zero which ensures that the matrix is invertible. With the new randomization library the algorithm remains to a single function call (create a 3×3 matrix with full rank):

```
$A=maple("
Random[MatInts](9,rows=3,column=3,rank=3)
");
```

The usual way of grading matrices with Maple T.A. is to use the `Equal` function of Maple's `LinearAlgebra` package. But with this function it is not possible to do partial grading of matrices since the return value is either true or false. The use of the grade library does this automatically (see Fig. 2).

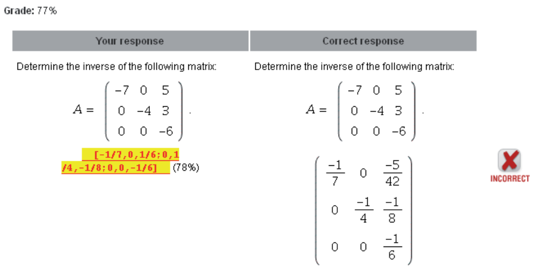

Fig. 2 *illustration of partial grading of matrices*

4.2 Multiple Choice Antiderivatives

For demonstrating the advantages of randomization from a set we employ an example of a multiple choice question, where students have to find the antiderivative of a given function (see Fig. 3).

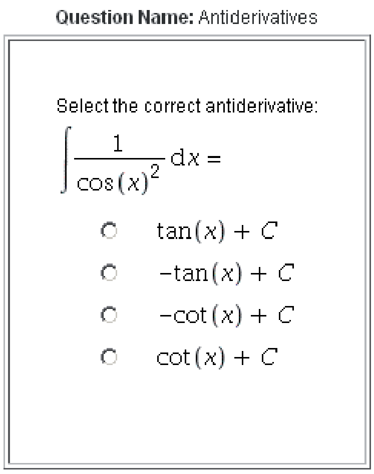

Fig. 3 *multiple choice antiderivatives example*

The Maple T.A. question algorithm remains to only two lines of code:

```
$f1 = maple("Random[FromSet]
  ({tan(x),-tan(x),cot(x),-cot(x)},4)
  ");
$int1ml = maple("
  Int(simplify(diff($f1[1], x)), x)
  ");
```

Within the first line, the set of functions is permuted and returned as vector, the second is required for the output of the question specification. The options needed for the multiple choice question can be taken from the vector obtained in line 1. The correct answer is located in `$f1[1]`.

4.3 Linear Homogeneous Differential Equation

Solutions of differential equations also need some preliminary work with Maple, that they can be graded accordingly. We examine an example of a real homogeneous linear differential equation with constant coefficients of second order: (see Fig. 4).

$$\frac{d^2x}{dt^2}(t) + a_1\frac{dx}{dt}(t) + a_2x(t) = 0.$$

A basis $\{y_1(t), y_2(t)\}$ of the solution is given by $\{e^{\lambda_1 t}, e^{\lambda_2 t}\}$, where λ_1, λ_2 denote the nulls of the characteristic polynomial of the differential equation. So the solutions are $x(t) = c_1e^{\lambda_1 t} + c_2e^{\lambda_2 t}$ [5].

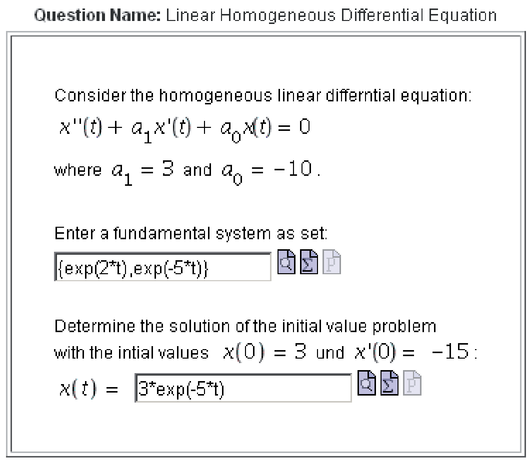

Fig. 4 *specifcation of differential equation example*

Since it is not advisable to have variables in the solution we want the student only to enter a basis of it. We cannot grade the student's response as a standard mathematical expression because we have to take into account that the correct answer does not depend on the order of the basis functions. For grading fundamental systems of linear differential equations we use the function `FundSys` of the grading library. Then the grading code for the Maple graded question remains to:

```
Grade[FundSys]("$RESPONSE",$ode,x(t),t,2)
```

`$RESPONSE` contains the student's response, `$ode` is the left side of differential equation to be solved, `x(t)` is the unknown function and `2` determines the order of the differential equation.

In figure 5 some grading results of the example above are shown. We see the results do not depend on the order of the entered function and the behaviour if either too many or too little functions are entered. If the number of entered functions differs only in one entry the answer is evaluated with 0.5 points.

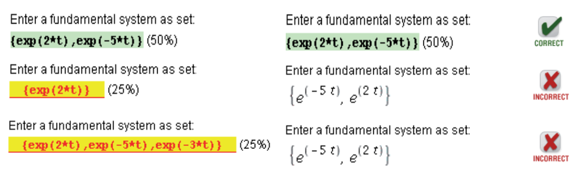

Fig. 5 *Grading of ODE example*

4.4 Definite Matrix and Eigenvalues Example

This example shows the usage of the `MatIntsDef` randomization function, that creates matrices with desired signatures. The signature, the number of positive, negative and eigenvalues equal zero, determines whether a matrix is positive definitie, negative definite or indefinite. So we call the `MatIntsDef` function with randomized parameters:

```
$neg = rint(4);
$pos = 3-$neg;
$A=maple("
   Random[MatIntsDef](4,$pos,$neg,0)");
```

Variable `$A` contains now a quadratic matrix with rank `$pos`+`$neg` and `$pos` positive eigenvalues and `$neg` negative eigenvalues. The matrix is positive definite if `$neg` equals zero, negative definite if `$pos` equals zero and indefinite in all other cases. The problems to be solved for the student are now to determine the eigenvalues of `$A` and then conclude whether the matrix is positive, negative or in-definite (see Fig. 6).

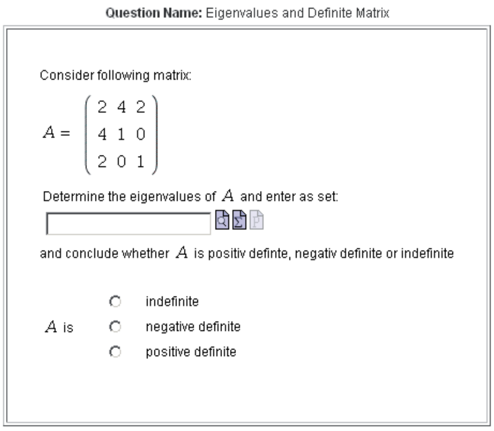

Fig. 6 *Definite Matrix and Eigenvalues Example*

The eigenvalues are graded as set, similar to the fundamental system, and the second question part is evaluated with the common Maple T.A. multiple choice grading routine. This example seems quite normal, but with a closer look it reveals one of Maple T.A.'s weaknesses. If the student makes a mistake during the calculation of the eigenvalues and gets for instance three positive eigenvalues instead of two positive and one negative, he is intended to identify the matrix as positive definite, which is a correct conclusion because of his calculations. Anyway the answer is graded as wrong because it is not possible to connect different answer fields to each other (Fig. 7)

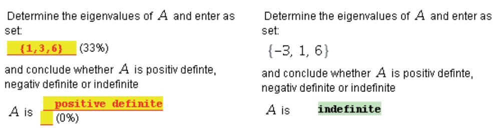

Fig. 7 *Incorrect grading of eigenvalues*

5 Usage experiences

The usage of the new libraries simplified the creation of Maple T.A. questions significantly. Due to the fact, that most of the algorithms used in questions were assumed by the new libraries and the Maple engine, the exercises became less error-prone and better readable for non-editors. On the other hand the increased use of Maple entailed some other unexpected problems. With many requests at the same time a certain slowdown of the system could be noticed. Also the creation of algorithms within questions has to be done with slight more attention. Infinite loops or other computationally intensive operations can cause very long computing times for the system. To avoid these problems question development within a local Maple installation is advisable.

Nevertheless the Maple T.A. system became more reliable with the use of the new libraries. The needed time for the development of exercises decreased in a noticeable way. So a larger amount of high-quality questions could be placed at disposal for the students.

6 Summary and Outlook

The recent work with the new developed library gained acceptance of the system among both students and question designers. But as everywhere there is still plenty of room for improvement and further development of new algorithms. According to the above mentioned system speed concerns the existing algorithms should also go under closer investigations. Further we are looking forward to future releases of new Maple T.A. versions, since some helpful features for question development have not been integrated into the system yet. Maybe the most wanted one is the conjunction of students' responses. With it the creation of high quality questions would be easier.

7 References

[1] Judex F., Breitenecker F., Schneckenreither G., and Zauner G. Cas-based e-learning for the improvement of refresher courses in mathematics. In Bretenecker F. Troch I., editor, *Argesim Report No. 35*, pages 2106–2111, Vienna, February 11-13 2009. Vienna University of Technology.

[2] Zimmermann A., Körner A., and Breitenecker F. Blended learning in maple t.a. in der lehre für mathematik und modellbildung. In Luther B. Gnauck A., editor, *Cottbus, September 2009*, pages 95–102, Cottbus, September 23-25 2009. Brandenburg University of Technology.

[3] Maplesoft. *Maple T.A. User Guide*. Maplesoft, a division of Waterloo Maple inc., Waterloo, ON, Canada, 2009.

[4] Sylvester J.J. A demonstration of the theorem that every homogeneous quadratic polynomial is reducible by real orthogonal substitutions to the form of a sum of positive and negative squares. *Philosophical Magazine*, IV.:138–142, 1852.

[5] Tenenbaum M. and Pollard H. *Ordinary Differential Equations*. Dover Publications, 1985.