GENERATOR OF PROBLEMS WITH ITS SOLUTIONS FOR THE COURSE OF NUMERICAL METHODS IN ENGINEERING

H. Pablo-Leyva, R. B. Silva-López, I. I. Méndez-Gurrola

Metropolitan Autonomous University (MEXICO)

Abstract

Problem-based learning is used in engineering courses with good results. However, the bibliographic collection is limited and a considerable amount of problems are required. Particularly in the Numerical Methods course for engineering, problems need to be solved in class, included in task, self-assessments and exams. In addition, it is necessary to change the problems every trimester, so that student does not have the answers of them previously.

The objective was to develop software to generate the problems used in the course of Numerical Methods in Engineering, as well as its solutions. This allows a wide variety of problems avoiding repetition from one course to another.

The methodology considers for each topic of the course: a) the design of a problem that is solved with one of the Numerical Methods; b) the problem is parameterized to generate a problem other than the same degree of complexity is obtained; c) choose the numerical method that will serve as a basis for determining iterations number to solves the problem for a given precision, so that a student can solve it manually in a reasonable time; d) programming a module that generates and solves problems; e) create a program that generates a file in XML format necessary to build the battery of problems for the SAKAI LMS exams. The programs were developed in shell script, C language and FORTRAN.

Automation in the generation of problems with their solutions for each topic of the Numerical Methods course in Engineering allows the teacher to have a wide range of possibilities with problems of the same level of complexity. As well as the generation of test batteries, they minimize the capture time in the LMS. As future work is intended to add to the program the functionality to generate problems and solutions of other courses.

Keywords: High Education, problem generator with its solutions, assessment, problem-based learning.

1 INTRODUCTION

In three campus of the Metropolitan Autonomous University there are several Engineering degrees. One of the courses included in the study programs is Numerical Methods and should be offered to approximately 500 students every trimester. One activity that demands a lot of work for the teacher is the problems preparation for the class, problems that the student solves of task and problems that are included in the exams. These problems need to change every trimester to prevent students share solutions. Teacher must solve the problems that he uses every trimester in order to qualify them. Therefore, the generation of problems and solutions of this course requires considerable time by the teacher.

The course consists of 8 topics, which implies that the teacher must prepare and solve a minimum of 40 problems per trimester, preparing at least 5 problems per topic, one example for the class, another 2 for task, and another 2 for the exam. If this activity is done manually it is common that errors occur when solving them and therefore affect the grades of students. Or, if teacher is not careful, it is possible to generate problems whose solution requires a large number of iterations to arrive at an approximate acceptable value, which implies that the time required by the student to solve the problem, exceeds the time destined to solve it and submit the exam.

On the other hand, online education allows attending large groups of students (more than 100 students per group), so it is necessary to use an LMS (Learning Management System). This type of systems has the advantage of handling a battery of problems, so that many versions of a test can be applied without being repeated. This has helped to give the course of Numerical Methods. However, an additional problem was generated, generating as many problems as needed, to include them in battery for the exams.
This leads us to develop programs that generate as many problems as are required for the course, so that they are not repeated from one course to another, also ensuring that iteration number necessary to solve the problem is done in a finite time and student can solve it in the time destined.

The objective was to develop software to generate problems used in the Numerical Methods course in Engineering, as well as its solutions. This allows a wide variety of problems avoiding repetition from one course to another. The program considers parameterization of the solutions contemplating the maximum number of iterations and the number of significant digits. Finally, it considers the generation of problems that converge in few iterations.

2 RELATED WORKS

Related works are classified into two sections, the first is considered work associated with problem-based learning, the other focuses on works whose objective is the development of programs to generate some kind of numerical problem.

2.1 Problem-based learning

Problem-based learning is a pedagogical strategy effectively applied to train students in Mechanical Engineering, a particular case is the study of systems of nonlinear vibrations, in which a finite element technique is applied, which is part of the numerical methods [1].

On the other hand, LMS Moodle in recent versions has incorporated 2 modules that implement problem-based learning [2]. One of these modules allows to define questions with random numerical values, which could be useful for the teaching of numerical techniques.

There are authors who have worked in electromagnetism courses, applying problem-based learning, supporting on the software development that uses the FDTD (Finite Difference Time Domain Method) [3].

The works found are focused on applying problem-based learning to a specific type of subject for engineering students, however, no works was found that applied directly to a numerical methods course for engineering. On the other hand, works are presented in which modules are added to an LMS, to implement problem-based learning, with questions that generate numerical values randomly, or implement some software that supports problem-based learning. No works were found in which an application that generates problems of numerical methods and their solutions was developed.

2.2 Generators of numerical problems

There are works that present software development or tools for the generation of numerical problems. This is the case of e-status, which is a web tool capable of generating numerical problems, focuses on the generation of statistics problems, not on numerical methods [4], [5], [6], [7].

On the other hand, there are repositories of problems already elaborated, for example, there is a repository of problems for chemical engineering, in which a collection of 10 numerical problems is presented [8]. These are typical problems in the practice of chemical engineering, which require numerical solution. The authors evaluate the solution with several software packages.

Another approach is the use of software that solves a specific type of problems [9], such as the use of software called OptiA, which solves mathematical programming problems related to optimization problems. In this work, a problem generator is not developed either.

Finally, a work was found in which problems of partial differential equations are solved [10], using various software packages.

Therefore, works done have several approaches: a) tendency to use existing software packages to solve a specific type of problems; b) use a repository of problems already elaborated and solve them also with existing software; and c) develop software that generates numerical problems. For the latter case the work that was located addresses the generation of problems for statistics courses, not for numerical methods in engineering.

3 METHODOLOGY

The methodology considers for each topic of the course is:
a) The design of a problem that is solved with one of the Numerical Methods;
b) The problem is parameterized to generate a problem other than the same degree of complexity;
c) Choose the numerical method that will serve as a basis for determining iterations number to solve the problem for a given precision, so that a student can solve it manually in a reasonable time;
d) Programming a module that generates and solves problems denominated PGP-NM (Program Generator of Problems for Numerical Methods);
e) Integrating problems and testing;
f) Create a program that generates a file in XML format necessary to build the battery of problems for the SAKAI LMS exams; and
g) Test the problem battery templates in an LMS.

The programs were developed in shell script, C language and FORTRAN.

4 SPECIFIC CONSIDERATIONS OF PGP-NM

This section describes the types of problems that are generated with the software developed PGP-NM, under certain conditions.

4.1 Theory of errors

The first topic that is contemplated in the course of numerical methods is theory of errors. All the contemplated operations integrate examples of cases that are problematic when performing arithmetic operations.

First case. Quadratic equations of the form:

\[ a_2 \times x^2 + a_1 \times x + a_0 = 0 \]

are generated.

Where:

- \( a_2, a_1, a_0 \): Are the coefficients randomly generated.

The coefficients are generated in such a way, to have the problem when applying the general formula to subtract 2 almost equal numbers. It is also tried to generate the problem of dividing between numbers near zero, when using a modification of the general formula.

Second case. It is considered to generate vectors of the form:

\[ x_1, x_2, x_3, \ldots, x_i, \ldots, x_n \]

\[ y_1, y_2, y_3, \ldots, y_i, \ldots, y_n \]

Where:

- \( x_i, y_i \): Are components of the vectors. These are generated randomly, the index i ranging from 1 to n.
- \( n \): Vector size, also randomly generated.

Vectors components are generated in such a way that when calculating their point product this is almost zero. The above allows to have a sum with values of different magnitude order for the calculation of the point product.

Third case. Consider a vector of the form:

\[ x_1, x_2, x_3, \ldots, x_i, \ldots, x_n \]

Where:

- \( x_i \): Vector components. They are generated randomly with the index i ranging from 1 to n.
- \( n \): Vector size is generated randomly.
Components are generated in such a way that the values variance leads to a difference of almost equal numbers.

4.2 Roots of Nonlinear Equations

For topic of roots of nonlinear equations, the following case is contemplated:

**First Case.** Equations of the type:

\[ a_2 \cdot f(x) + a_1 \cdot x^n + a_0 = 0 \]

Where:

- \(a_2, a_1, a_0\) : Coefficients randomly generated.
- \(f(x)\): \(e^x, e^{-x}, \cos(x), \sin(x), \cosh(x), \sinh(x)\). It is chosen randomly.
- \(n\): Exponent of 1 to 4, randomly generated.

**Second Case.** They are considered polynomials of the form:

\[ a_n x^n + a_{n-1} x^{n-1} + \cdots + a_2 x^2 + a_1 x + a_0 = 0 \]

Where:

- \(a_n, a_{n-1}, a_2, a_1, a_0\) : Coefficients randomly generated.
- Polynomial degree is randomly generated from 1 to 4.

Coefficients and exponent are generated in such a way that the equations have at least one real root, which converges to the solution in less than a given number of iterations with a given precision.

4.3 Systems of Linear Equations

In topic of systems of linear equations, systems of the type:

\[
\begin{align*}
    a_{11}x_1 & + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n = b_1 \\
    a_{21}x_1 & + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n = b_2 \\
    \vdots & \\
    a_{n1}x_1 & + a_{n2}x_2 + a_{n3}x_3 + \cdots + a_{nn}x_n = b_n
\end{align*}
\]

are generated.

Where:

- \(a_{ij}\) : Coefficients of the matrix that are generated randomly.
- \(b_i\) : Independent terms that are calculated based on the system solution.
- \(x_i\) : Solution of the system, which are generated randomly.
- \(i,j\): Indexes of coefficients, independent terms, or solutions, ranging from 1 to \(n\).
- \(n\): System size.

**First case.** The coefficients and solutions are generated in such a way, that systems of linear equations converge to the solution in less than a determined number of iterations, with a certain precision. This is by an iterative method: Gauss Jacobi or Gauss Seidel.

**Second case,** it seeks to generate the coefficients and solutions, in such a way to have equations systems that are badly conditioned, to be solved by a direct other method.
4.4 Systems of Nonlinear Equations
For topic of systems of nonlinear equations.

**First case.** Systems of the type:

\[
(x - h_1)^2 + (y - k_1)^2 = radio_1^2 \\
m \cdot x + b = y
\]

are generated.

**Second case.** Systems of the type:

\[
(x - h_1)^2 + (y - k_1)^2 = radio_1^2 \\
(x - h_2)^2 + (y - k_2)^2 = radio_2^2
\]

are generated.

Where:
- \(h_1, k_1, h_2, k_2\): Circle center coordinates are generated randomly.
- \(m, b\): Slope and ordered to the origin of the line, are generated randomly.
- \(radio_1, radio_2\): Circles radios, are randomly generated.

Circles coordinates, radios, slope and ordered to the origin are generated in such a way, that systems of nonlinear equations have one or two solutions.

4.5 Interpolation
In the interpolation topic, the necessary tables are generated with nonlinear equations of the type:

**First case.** Equations of the type:

\[
a_2 \cdot f(x) + a_1 \cdot x^n + a_0 = 0
\]

are generated.

Where:
- \(f(x) = e^x, e^{-x}, e^{-x^2}, \cos(x), \sinh(x), \cosh(x), \sin(x), \cos(x), \sqrt{x}, x^{\alpha}\). It is chosen randomly.
- \(a_2, a_1, a_0, \alpha\): Coefficients randomly generated.
- \(n\): Exponent from 1 to 4, generated randomly.

**Second case.** They are considered polynomials of the form:

\[
a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 = 0
\]

Where:
- \(a_4, a_3, a_2, a_1, a_0\): Coefficients randomly generated.

Polynomial degree is randomly generated from 1 to 4.

Generated tables are as of the form of Table 1:

<table>
<thead>
<tr>
<th>Table 1. Format of generated tables</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Y</td>
</tr>
<tr>
<td>X_1</td>
<td>Y_1</td>
</tr>
<tr>
<td>X_2</td>
<td>Y_2</td>
</tr>
<tr>
<td>X_m</td>
<td>Y_m</td>
</tr>
</tbody>
</table>
Where:

$X_i, Y_i$: Points in the table generated with one of the equations above, where the index goes from 1 to $m$.

$m$: Number of points in the table, which is randomly generated.

Coefficients and exponent are generated in such a way, that the equations have values, when evaluated at a randomly determined interval, nor very large to cause problems to handle them, nor very small to be considered zero. Repeated values are avoided and it is taken into account that some functions such as $\sqrt{x}$

which have problems with negative values, are not included. Finally, random variable length intervals are determined to consider unequally spaced tables. Likewise, equally spaced tables are also generated. Values are randomly generated to be interpolated or extrapolated.

4.6 Curve Adjustment

For topic of curve adjustment, the necessary tables are generated in the same way as for interpolation, with special consideration: the tables are equally spaced.

In a random manner, it is determined that

First case: the best available curve,

Second case: a curve of the form:

$$a_2 \cdot f(x) + a_1 \cdot x^n + a_0 = 0$$

In this case, the value of coefficient $a_0$, exponent $n$, and the function $f(x)$ are provided. Leaving to the student the task of finding the other coefficients, adjusting the curve.

Finally they will be generated as in the case of random interpolation, values where it will be interpolated or extrapolated with the adjusted curve. This in order to review the root of nonlinear equations topic.

4.7 Numerical Integration

On topic of numerical integration.

First case, randomly select one of the following integrals:

1. $\int_0^{\alpha} \sin^{2\alpha+1}(x) \cos^{2\beta+1}(x) dx, \alpha > 0$, $\beta > 0$
2. $2\int_0^{\alpha} \sin^{2\alpha-1}(x) \cos^{2\beta-1}(x) dx, \alpha > 0$, $\beta > 0$
3. $\int_0^{\alpha} \frac{\sin(x)}{\sqrt{1-\alpha^2 \sin^2(x)}} dx, |\alpha| < 1$
4. $\int_0^{\alpha} \frac{\cos(x)}{\sqrt{1-\alpha^2 \sin^2(x)}} dx, |\alpha| < 1$
5. $\int_0^{\alpha} \frac{\sin^2(x)}{\sqrt{1-\alpha^2 \sin^2(x)}} dx, |\alpha| < 1$
6. $\int_0^{\alpha} \frac{\cos^2(x)}{\sqrt{1-\alpha^2 \sin^2(x)}} dx, |\alpha| < 1$
7. $\int_0^{\alpha} \frac{\cos(\alpha)}{1-2\beta \cos(\alpha) + \beta^2} dx, \alpha \geq 0$, $|\beta| < 1$
8. $\int_0^{\alpha} \ln(\alpha + \beta \cos(x)) dx, \alpha \geq \beta$
9. $\int_0^{\alpha} \ln(\alpha - \beta \cos(x)) dx, \alpha \geq \beta$
10. $\int_0^{\alpha} \ln(\alpha^2 - 2\alpha \beta \cos(x) + \beta^2) dx, \alpha \geq \beta > 0$, $\beta \geq \alpha > 0$
11. $\int_0^1 x^\alpha (1-x)^\beta \, dx, \alpha > 0, \beta > 0$

12. $2\int_0^1 x^{2\alpha+1}(1-x^2)^\beta (x) \, dx, \alpha > 0, \beta > 0$

13. $\int_0^1 \frac{x^{\alpha-1} x^{\beta-1}}{(1+x)^{\alpha+\beta}} \, dx, \alpha > 0, \beta > 0$

14. $\int_0^1 \frac{1}{1-2\cos(\alpha)x^\alpha} \, dx, 0 < \alpha \leq \frac{\pi}{2}$

Where:

$\alpha, \beta$: Integral coefficients are generated randomly. Coefficients are generated in such a way that the constraints for each integral are met. They are also generated in such a way that converges to the value of the integral in less than a determined number of iterations, with a certain precision.

### 4.8 Numerical Differentiation

Finally, for the case of numerical differentiation, the necessary tables are generated in the same way as for interpolation, with the proviso that they are equally spaced.

In a random way the order of the derivatives of 1 to 4 is generated. They are only considered derivatives in the extremes and center of the table.

## 5 RESULTS

The problem-generating program for the numerical methods course (PGP-NM) has been used since the end of year 2011 to date. A total of 31 numerical methods courses have been given, using this software as shown in Table 2.

<table>
<thead>
<tr>
<th>Year</th>
<th>Groups attended</th>
</tr>
</thead>
<tbody>
<tr>
<td>2011</td>
<td>3</td>
</tr>
<tr>
<td>2012</td>
<td>4</td>
</tr>
<tr>
<td>2013</td>
<td>4</td>
</tr>
<tr>
<td>2014</td>
<td>7</td>
</tr>
<tr>
<td>2015</td>
<td>6</td>
</tr>
<tr>
<td>2016</td>
<td>6</td>
</tr>
<tr>
<td>2017</td>
<td>1</td>
</tr>
</tbody>
</table>

The PGP-NM allows to generate more than 1000 problems. The time required to generate problems and their solutions depends on the type of numerical method applied, that is, it can go from about 3 minutes for problems of root to nonlinear equations, until 4 hours for problems involving tables. This involves considerable time savings, since before having this system, the teacher took weeks to generate and solve a few problems.

Technologies used for the development of PGP-NM are:

1. Linux CentOS for the server operating system.
2. Shell scripts in Korn Shell for generating tables or equations or problem batteries.
3. FORTRAN and C language for solving problems.
4. XML for question batteries for the LMS.

The LMS used was Sakai [11], given the advantages it offers [12], [13], [14], [15].
6 CONCLUSIONS

Since PGP-NM was developed, it is easy, reliable and fast to elaborate the problems used for exercises, tasks and exams in numerical methods courses for engineering. Avoiding the complex work of manually elaborating problems and solving them.

The PGP-NM allows the generation of batteries with hundreds of problems, to elaborate exams in an LMS, in such a way that practically avoiding to present the same examination to 2 or more students.

PGP-NM enables the efficient use of problem-based learning, since the teacher counts with the solutions of all the problems generated, and is easier to clarify doubts of students. At the same time, it prevents students share the solutions of problems from one course to another.

A possible copyright problem is also avoided when using book problems.

7 FUTURE WORK

Future work is to replace the FORTRAN code with C, to increase the type of equations generated, to include the generation of problems solving ordinary differential equations, to minimize the execution time as much as possible for cases that take hours, use a database to store problems generated in an open repository.

Other improvements considered for PGP-NM are to add stored problem search functionality, add the export to formats such as PDF and spreadsheet, as well as generate problems from other engineering courses in which problem-based learning is applied.

ACKNOWLEDGMENT

This work was carried out with the support of PRODEP (Program for the Professional Development of Higher Education Teaching) through the Program of CODAES (Networks for Digital Communities for Learning in Higher Education).

REFERENCES


