FOLIA 290

Annales Universitatis Paedagogicae Cracoviensis

Studia ad Didacticam Mathematicae Pertinentia 11(2019)

ISSN 2080-9751
DOI 10.24917/20809751.11.11

Gregor Milicic, Simon Plangg

Experiments with iterative methods for linear systems using GeoGebra and TI-Nspire*

Abstract. Algorithms and algorithmic thinking are key topics in STEM Education. By using algorithms approximate solutions can be obtained for analytical unsolvable problems. Before new methods can be safely applied they have to be thoroughly tested in experiments.

In this article we present a series of exercise where students can experiment with algorithms and test them using GeoGebra or the TI-Nspire. Based on the results of such experiments the students can compare algorithms, showing them a heuristic and dynamical aspect of Mathematics.

1. Introduction

Real world problems provide a productive starting point to make contact with algorithms for analytical unsolvable problems in STEM education (Milicic, 2019b). As a first step in a modelling cycle, the real world problem has to be translated into the language of mathematics (Velten, 2009). In the context of STEM education, usually algorithms implemented on digital tools are used (Greefrath, 2011) to solve the mathematical problem. Before new methods can be safely applied for a variety of applications, they have to be thoroughly tested in experiments regarding their properties such as efficiency, accuracy and stability. Afterwards, results and predictions based on simulations (Mityushev, Nawalaniec, Rylko, 2018) can be concluded about the real world problem.

In this article we focus on the deformation process of a bridge under a constant load as a starting point and real world application. The main idea thereby is to let the students conduct experiments related to this model problem using GeoGebra and the TI-Nspire. Both software systems are common and worldwide used in the

*2010 Mathematics Subject Classification: Primary: 52B55; Secondary: 97R20
Keywords and phrases: Technology, Experiments, Algorithms, STEM Education, Computational Thinking

context of schools, in particular considering teaching and learning of mathematics. GeoGebra is furthermore free for non-commercial use and doesn't require any registration or installation when using the provided Applet (Milicic, 2019a), but only a browser with Javascript. Reasons for using the TI-Nspire are manifold. It provides elaborate pedagogical and didactical applications, also for programming. Although it is not free of charge, the software is affordable to schools in comparison to high developed packages such as MATLAB. Prepared files presented in this article ensure that the students can start with the corresponding experiments right away without having any prior knowledge about programming. From a mathematical perspective the exercises focus on using iterative methods to solve systems of linear equations such as the Jacobi, the Gauß-Seidel and the SOR (Successive Over-Relaxation) method (Schuppar, Humenberger, 2015; Meister, 2015).

The first section of this article explains the theoretical background for the experiments, i.e. the mathematization process of a bridge under a heavy load, yielding to a system of linear equations. In section two we present experiments using the iterative Jacobi, the Gauß-Seidel and the SOR method to solve the arising system of linear equations. The first experiments are related to the principle of modelling and functional thinking. How does the system change if we vary the force acting on the bridge are directive questions in this case for example. The following experiments focus on the convergence of those methods, it is then up to the students to determine how accurate those methods work and to compare them. Subsequently we present some exercises to analyse the influence of the relaxation parameter $\omega \in (0, 2)$ on the convergence of the SOR method. Section three provides insights how the implementation of those methods was done using the TI-Nspire and Javascript in GeoGebra.

1.1. Theoretical background

As a model problem we consider the following differential equation in one dimension:

$$-u''(x) = f \tag{1}$$

on $\Omega = [0, l]$ with $l \in \mathbb{R}^+$, $f \in \mathbb{R}^-$ and the Dirichlet boundary conditions $u(0) = 0 = u(l)$, or as used for the implementation with the TI-Nspire the symmetric Dirichlet boundary conditions $u(-\frac{l}{2}) = 0 = u(\frac{l}{2})$. The problem is therefore a simplified version of the Poisson equation in one dimension. A physical interpretation of equation (1) for example is the deformation process of a bridge or a string of length $l \in \mathbb{R}^+$ under a constant load $f \in \mathbb{R}^-$ (see Fig. 1). Usually the analytical solutions of differential equations are unknown and numerical methods like the Finite Element Method (FEM) are needed to discretize the problem and get an approximate solution u_h (Hughes, 2012; Logan, 2007). Using Finite Elements of order one, i.e. linear Ansatz functions, and equidistant points $x_i = i \cdot \frac{l}{n+1}$ with $n \in \mathbb{N}$ the total number of the so called *nodes*, solving the model problem (1) is being reduced to find a solution $y \in \mathbb{R}^n$ for the linear system

$$Ay = b, \tag{2}$$

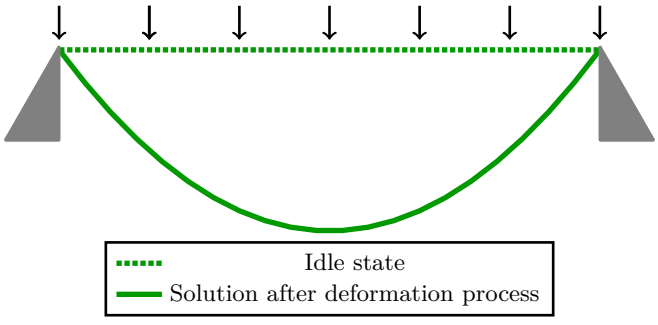


Fig. 1: The left and right fixated object (string or a bridge) is being deformed under a load.

with $A \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$. The entries of A and b can be given explicitly by

$$A_{ij} := \begin{cases} 2\frac{n+1}{l}, & i = j \\ -\frac{n+1}{l} & |i-j| = 1 \\ 0 & \text{otherwise} \end{cases}, \quad \text{and} \quad b_i = f \cdot \frac{l}{n+1}. \tag{3}$$

By construction y_i describes the displacement of the approximate solution u_h of the bridge at point x_i, $u_h(x_i) = y_i$, a direct visualization of the approximate solution is hence possible and can help the students to understand the connection between the solution of the linear system (2) and the approximate solution of the model problem (1).

The linear system (2) should be solved using iterative methods, as the condition of the so called *stiffness matrix* A gets bigger with a growing number of nodes $n \in \mathbb{N}$, resulting in big rounding errors when direct methods like Gaussian elimination are used. Often the exact solution is not even needed for applications, an approximate solution up to a certain degree of accuracy is sufficient.

In the following, we want to focus on experiments to solve the linear system (2) with A and b as given in (3) using the mentioned Jacobi, Gauß-Seidel and SOR method. The convergence of all three methods can be proven as A is a tridiagonal matrix (Kulkarni, Schmidt, Tsui, 1999), and the optimal parameter $\omega \in (0, 2)$ for the SOR method can also be given explicitly by

$$\omega_{opt} = \frac{2}{1 + \sin\frac{\pi}{n+1}}. \tag{4}$$

However, for this simple model problem (1) an exact solution with the Dirichlet conditions $u(0) = 0 = u(l)$ can be given by

$$u_{exact}(x) = -\frac{f}{2}x^2 + \frac{1}{2}fl \cdot x, \tag{5}$$

which can be used to evaluate the quality of the approximate solution u_h.

1.2. Educational aspects for the following teaching sequence

To solve the presented exercises in section 2 the students should be familiar with the technology, GeoGebra or the TI-Nspire. The theoretical background can also be briefly presented, in which case the concept of a *derivative* should be known to the students. The Finite Element Method as a well known and often used method in research and real world applications can be mentioned, but should be treated as a black box due to the necessary theoretical knowledge.

In accordance with the advice of Karl Fuchs (Fuchs, 1988) the algorithms should first be introduced using simple examples of linear systems, for example the arising matrix A and the load vector b for $n = 3$ nodes with $l = 2$ and a force of $f = -1$, resulting in the following system:

$$A = \begin{pmatrix} 4 & -2 & 0 \\ -2 & 4 & -2 \\ 0 & -2 & 4 \end{pmatrix} \quad \text{and} \quad b = \begin{pmatrix} -0.5 \\ -0.5 \\ -0.5 \end{pmatrix}. \tag{6}$$

The students can and should solve this system using the iterative methods without any technology to understand the instructions and necessary steps. Naturally the question arises how often the iteration should be conducted. The *residuum* for the approximate solution y^i after i iterations given by $res(y^i) = Ay^i - b$ can be introduced as a canonical choice as a termination criteria. The connection between the number of nodes and the number of equations should also be mentioned.

Based on the boring and monotonous calculation process the possibility to transfer the instructions on the computer can be presented to the students. At this point the students can use the Applets provided for GeoGebra and the TI-Nspire to work on the exercises presented in section 2.

Further suggestions for this phase of teaching can be found in (Milicic, 2016). Especially the step from a specific matrix A as given in (6) to a general 3×3 matrix and afterwards to a matrix $A \in \mathbb{R}^{n \times n}$ can be challenging for the students and should be accompanied by examples and explanations.

2. Experiments

Four different exercises are given in section **2.1.** to introduce the students with the whole setting and the connection between the application of a bridge being deformed under a heavy load and the corresponding linear system. The three exercises in section **2.3.** focus on the SOR method presenting experiments to determine the optimal choice for the relaxation parameter $\omega \in (0, 2)$. For the exercises 2.2 to 2.4 a value of $\omega = 1.5$ was used for the relaxation parameter.

2.1. Comparison of the methods - tasks

With regard to the model problem (1) the following exercise addresses the structure of the given system of linear equations $Ay = b$ and also the solution process using the mentioned iterative methods.

Exercise 2.1
Examine the structure of the system of linear equations $Ay = b$ arising from the model problem having $n = 3, 4, 5, 8$ nodes. Therefor also vary the force f affecting the bridge as well as the length l of the bridge. How do the parameters f and l influence the structure of the system of linear equations? Describe.

Exercise 2.2
Solve the system of linear equations resulting from the model problem ($f = -1, l = 4$) for $n = 4, 8, 16, 32$ nodes using the following methods with a maximum of 100 iterations:

- *Jacobi method*
- *Gauß-Seidel method*
- *SOR method ($\omega = 1.5$)*

Compare the accuracy of the obtained approximative solutions for each value of $n = 4, 8, 16, 32$ towards the exact solution. How does the accuracy change for bigger values of $n \in \mathbb{N}$? Describe.

Exercise 2.3
Solve the system of linear equations resulting from the model problem ($f = -1, l = 4$) for $n = 4, 8, 16, 32$ nodes using the following methods with a tolerance of $tol = 10^{-6}$ for the residues:

- *Jacobi method*
- *Gauß-Seidel method*
- *SOR method ($\omega = 1.5$)*

Compare the obtained number of iterations for all three methods. How does the number of iterations change for bigger values of $n \in \mathbb{N}$? Describe.

Exercise 2.4
Based on the results of 2.3 give a presumption of how many iterations the named methods will take in case of $n = 64$ nodes. Check your assumption using the technology and document your findings.

2.2. Comparison of the methods - possible results

Exercise 2.1: Varying the acting force while keeping the length as well as the number of nodes constant only the right side b of the system changes while the matrix A remains unaffected (see Fig. 2). Changing the acting force with the factor c, also affects the values of b by the factor of c.

In contrast to that, varying the length of the bridge leads to a change in both sides of the system of linear equations (see Fig. 3). A change in length of the bridge with the factor c affects the values of the matrix A with the factor $\frac{1}{c}$ and the right side b with the factor c. A change in the number of nodes n also affects both sides of the system of linear equations. As mentioned before the number of

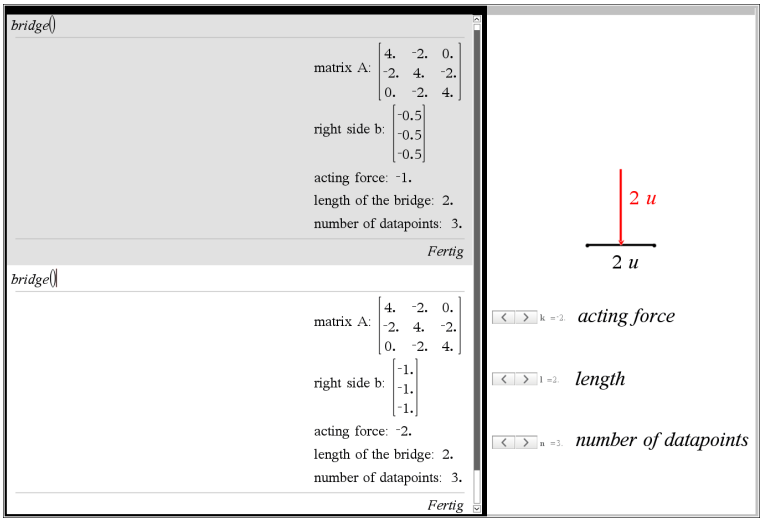


Fig. 2: Comparison of the structure of the system of linear equations $Ay = b$ arising from the model problem varying the acting force f

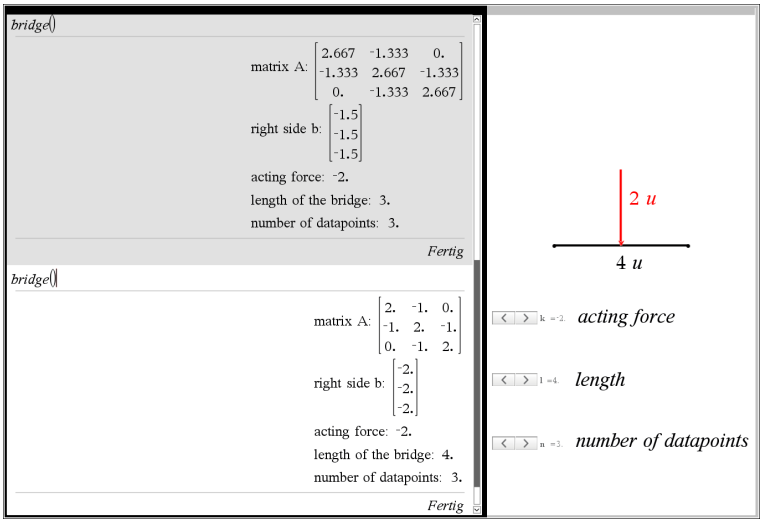


Fig. 3: Comparison of the structure of the system of linear equations $Ay = b$ arising from the model problem varying the length l of the bridge

nodes complies with the number of equations (or number of unknowns) of the system. The underlying structure of the matrix A stays the same though. It is always a symmetric tridiagonal matrix with identical values in each diagonal. In the end raising the number of nodes indicates an increase in computational costs.

The structure of the matrix A with many zero elements suggests to process the system of linear equations with iterative methods taking advantage of this structure.

Exercise 2.2: The results of the experiments indicate that the approximative solution of the SOR method gets the closest to the exact solution when limiting the number of iterations. The solution provided by the Jacobi method is the furthest from the exact solution (see Fig. 4). An increase in number of nodes also raises

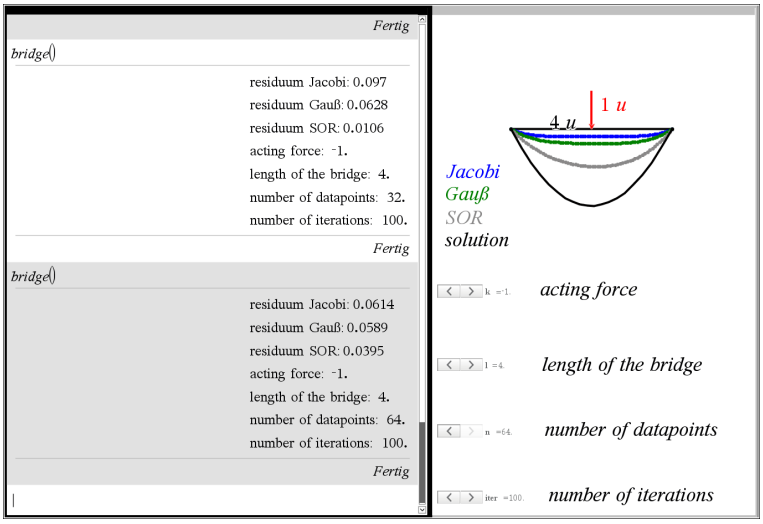
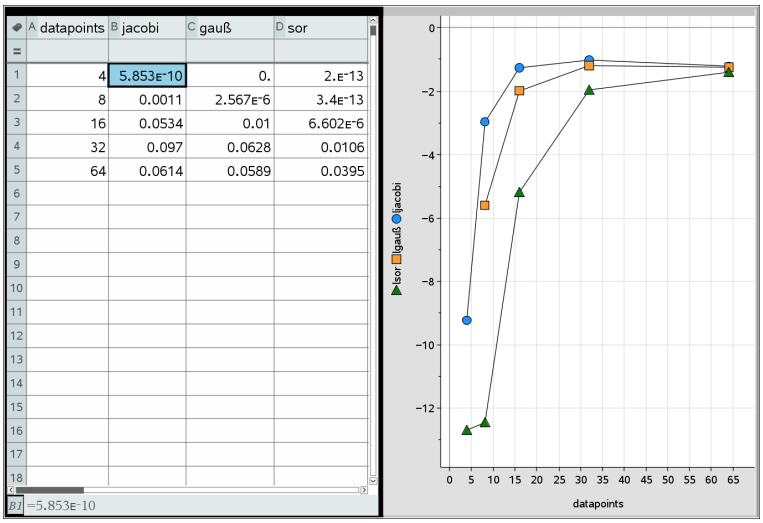


Fig. 4: Provided residues of all three methods varying the number of nodes from 32 to 64 (left), parameters, approximative and exact solution of the model problem (right)

the residues of all three methods when having a constant number of iterations indicating that the given number of iterations is not enough to provide a rather small residuum for the given number of nodes (see Fig. 5). Regardless the residues seem to stabilize at a magnitude of 10^{-2} which is available in all cases.

Exercise 2.3: The results indicate that an increasing number of nodes also raises the number iteration with a given tolerance for the residuum (see Fig. 6). Also, the Jacobi method takes approximately twice as many iterations the Gauß-Seidel method does, just as the theory predicts (Meister, 2015). Compared to those two methods the SOR method takes the least number of iterations. In detail, the log-log diagram in (6) indicates that the relation between the number of nodes and the number of iterations can be approximately described with a power function. In a log-log diagram the slope of the line indicates the exponent of the power function. As all three lines have an approximate slope of 1.8 this leads to a function rule similar to $a \cdot x^{1.8}$. Whereas the parameter a is different for all three lines indicating the vertical position of the line.

Exercise 2.4: Using the function rule $a \cdot x^{1.8}$ and $n = 64$ nodes from task 2.3 it can be assumed that the Jacobi method takes approximately $10^{3.42+(\log 64-1.51)\cdot 1.8} = 8977$, the Gauß-Seidel method approximately $10^{3.12+(\log 64-1.51)\cdot 1.8} = 4499$ and the



	A datapoints	B jacobi	C gauß	D sor
1	4	5.853E-10	0.	2.E-13
2	8	0.0011	2.567E-6	3.4E-13
3	16	0.0534	0.01	6.602E-6
4	32	0.097	0.0628	0.0106
5	64	0.0614	0.0589	0.0395

Fig. 5: Number of nodes with the corresponding residues of all three methods (left) and a graphical illustration of this relation using a logarithmic scale (base 10) for the residues (right)

SOR method approximately $10^{2.64+(\log 64-1.51)\cdot 1.8} = 1490$ iterations to hit the tolerance of 10^{-6} for the residuum. Checking with the TI-Nspire, the actual number of iterations amounts to 9644 for the Jacobi method, 4824 for the Gauß-Seidel method and 1604 for the SOR method.

2.3. Optimal parameter for the SOR method - tasks

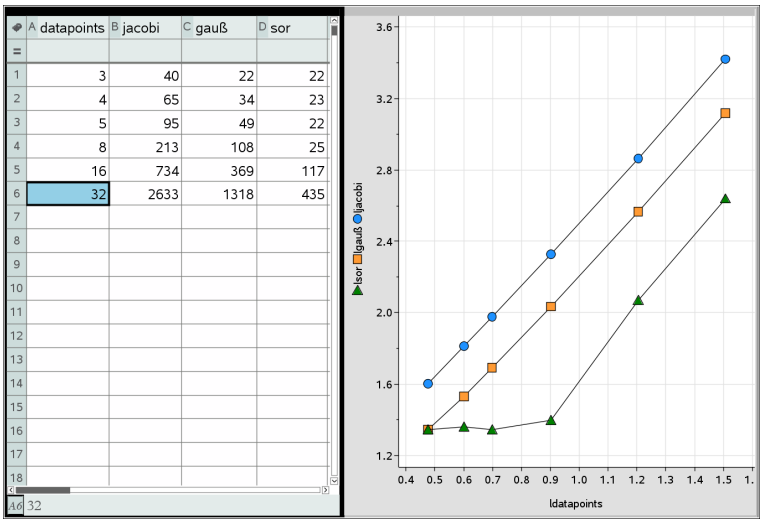
The following experiments focus on the SOR method and the optimal parameter $\omega \in (0, 2)$ to solve the linear system $Ay = b$.

Exercise 2.5
Solve the system of linear equations resulting from the model problem ($f = -1, l = 2$) for $n = 4, 8, 16, 32$ nodes using the SOR method with a tolerance of $tol = 10^{-9}$. Use different values for the relaxation parameter $\omega \in (0, 2)$. For which value of ω does the SOR method require the smallest number of iterations?

Exercise 2.6
Theoretically the optimal choice for the relaxation parameter ω_{opt} for the SOR method applied to the model problem ($f = -1, l = 2$) is

$$\omega_{opt} = \frac{2}{1 + \sin \frac{\pi}{n+1}}. \tag{7}$$

Can you verify this choice for $n = 3, 4, 5, 8, 16, 32$? Compare your findings with the results from exercise 2.5!



	A datapoints	B jacobi	C gauß	D sor
1	3	40	22	22
2	4	65	34	23
3	5	95	49	22
4	8	213	108	25
5	16	734	369	117
6	32	2633	1318	435

Fig. 6: Number of iterations for all three methods while increasing the number of nodes with a given tolerance of 10^{-6} for the residues (left) and a log-log diagram(base 10) of the given relation between the number of nodes and the number of iterations for all three methods (right)

EXERCISE 2.7
How do your results of the exercises 2.2, 2.3 and 2.4 change if you use the optimal value for the relaxation parameter ω?

2.4. Optimal parameter for the SOR method - possible results

Exercise 2.5 and 2.6: Parameter studies, i.e. experimental analysis of the influence of one particular parameter, are a commonly used approach as a first step when working with and beginning to understand new numerical methods.

Using the given GeoGebra Applet (Milicic, 2019a) the results from Table 1 can be obtained (see Fig. 7). The optimal choice for the relaxation parameter using the formula (7) with the corresponding number if iterations is given as well as experimentally obtained better values for ω. The students can see that theoretical results about approximate methods always have to be tested in numerical experiments, sometimes getting even better (or worse or unexpected) results due to rounding errors.

Exercise 2.7: The results with the optimal parameter for the SOR method using formula (7) for the exercises 2.2, 2.3 and 2.4 are listed in Table 2. As can be expected, the results get better using the optimal value for the parameter ω. Initially a value of $\omega = 1.5$ was used for the relaxation parameter in the exercises 2.2 to 2.4.

	SOR		SOR	
n	ω	$Iter_{min}$	ω_{opt}	$Iter_{min}$
3	1.175	13	1.17157	15
4	1.3	19	1.25962	19
5	1.335	22	1.33333	23
8	1.5	34	1.49029	35
16	1.7	64	1.68955	67
32	1.826	131	1.82639	129
64	1.908	246	1.90783	249

Table 1: Number of iterations and the corresponding choice of ω for different number of nodes n for the model problem (1) ($f = -1, l = 2$) as a possible result for exercises 2.5 and 2.6

		Exercise 2.2	Exercise 2.3
n	ω	$residuum = Ax^i - b$	number of iterations
4	1.3	$1 \cdot 10^{-10}$	14
8	1.5	$1 \cdot 10^{-10}$	25
16	1.7	$1 \cdot 10^{-10}$	48
32	1.826	$6 \cdot 10^{-7}$	96
64	1.908	$1.34 \cdot 10^{-3}$	180

Table 2: Number of iterations and the corresponding choice of ω for different number of nodes n for the model problem (1) ($f = -1, l = 2$) as a possible result for exercise 2.7

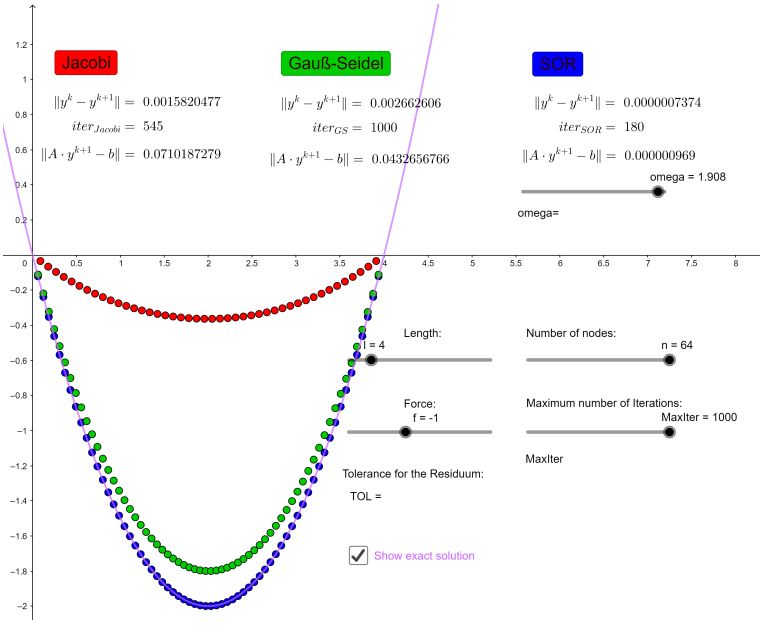


Fig. 7: The GeoGebra Applet (Milicic, 2019a) can be used to solve the model problem (1) and to obtain the results given in Table 1 and Table 2

3. Implementation

The implementation of the presented iterative methods in GeoGebra was rather difficult, as GeoGebra is a great tool for geometry, algebra and even the use of spreadsheets, but it does lack a programming interface or loops, making it thus very difficult to program the iterative methods. We nevertheless decided to use GeoGebra, as it is commonly used and accepted, free to use for educational purposes, a great tool for visualization and doesn't require any registration or installation.

The closest thing to actual loops in GeoGebra are sequences, but a `do ... while` loop is not supported, which is required for iterative methods where the number of iterations is not know at the beginning but dependent on the termination criteria.

GeoGebra however does support scripting, we hence implemented the iterative methods in GeoGebra using Javascript, which has several control structures and arrays to store the data.

The parameters of the Applet can be accessed in Javascript by defining first a variable for the Applet by `var g=ggbApplet;`, afterwards the parameter for the length l of the bridge for example can be accessed in Javacript with `var l = g.getValue("l");`.

After the iteration stops and the approximate solution is obtained, all the data (approximate solution, residuum, number of iterations) has to be transferred back from Javascript into the GeoGebra interface. We mainly used the following three commands for this task.

```
// set the Value of a Variable in GeoGebra
g.setValue("VarNameGeoGebra",value);
// evaluate directly a GeoGebra command
g.evalCommand("GEOGEBRACOMMAND");
// set the value of different indixes in a list
g.setListValue("NameList", index, value);
```

After termination of the method the components of the approximate solutions were first written into a list and in a second step a GeoGebra command was called in Javascript to copy the list values into a sequence. The sequence is immediately shown in the Algebra View, so that the approximate solution is directly visualized after the calculation is done.

The implementation with the TI-Nspire is rather simple compared to GeoGebra. For this purpose, we used the provided program editor to implement the programs for the discussed iterative methods. The supported programming language is TI-Basic using a comfortable menu to process algorithms, e.g. define variables, insert control structures or check syntax (see TI-Basic Reference[1]). Within this editor the students can use the syntax they know from working with all the other TI-Nspire applications. For the analysis of the methods the programs (e.g. `bridge()`, Fig. 4) can be executed using a calculator storing the results in certain variables. Illustrating those results is now truly simple using provided data & statistics or

[1] `https://education.ti.com/en-gb/guidebook/search`

spreadsheet windows reading the values of those variables. The prepared tns-files for the discussed exercises can be easily transferred to the students' hand helds or computers in order to let them focus on the content immediately. They are also provided on the webpage (Plangg, 2019) and are free to use for educational purposes.

4. Conclusion

Experiments with (numerical) algorithms can be conducted in school using existing tools like GeoGebra or TI-Nspire. We presented a series of exercises showing how students can make contact with numerical mathematics without having any knowledge in a programming language using the Applets provided in this article. It is about making important aspects of (numerical) mathematics such as algorithmic, dynamical, constructive and theoretical aspects (Pfahl, 1990) available to students exhausting the potential of the technologies mentioned (Plangg, Milicic, 2018).

References

Fuchs, K.: 1988, Erfahrungen und Gedanken zu Computern im Unterricht, *Journal falr Mathematik-Didaktik* **2/3**, 247–256.

Greefrath, G.: 2011, Using technologies: New possibilities of teaching and learning modelling – overview, w: G. Kaiser, W. Blum, R. Borromeo Ferri, G. Stillman (red.), *Trends in Teaching and Learning of Mathematical Modelling*, Springer Netherlands, Dordrecht, 301–304.

Hughes, T.: 2012, *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*, Dover Civil and Mechanical Engineering, Dover Publications. Accessible at: `https://books.google.at/books?id=cHH2n_qBK0IC`.

Kulkarni, D., Schmidt, D., Tsui, S.-K.: 1999, Eigenvalues of tridiagonal pseudo-toeplitz matrices, *Linear Algebra and its Applications* **297**(1), 63 – 80. Accessible at: `http://www.sciencedirect.com/science/article/pii/S0024379599001147`.

Logan, D.: 2007, *First Course in the Finite Element Method*, Thomson. Accessible at:`https://books.google.at/books?id=wjr3ArdvAc4C`.

Meister, A.: 2015, *Numerik linearer Gleichungssysteme*, Springer Fachmedien Wiesbaden, Springer.

Milicic, G.: 2016, Iterative Laser falr lineare Gleichungssysteme, *Mathematik im Unterricht* 13–20.

Milicic, G.: 2019a, *GeoGebra Applet for Iterative methods*, Accessible at: `https://www.geogebra.org/classic/qjvyt6w4`.

Milicic, G.: 2019b, Innermathematisches Experimentieren im Kontext der Modellierung mit Algorithmen, *Beiträge zum Mathematikunterricht* .

Mityushev, V., Nawalaniec, W., Rylko, N.: 2018, *Introduction to Mathematical Modeling and Computer Simulations*, CRC Press. Accessible at: `https://books.google.de/books?id=EtNMDwAAQBAJ`.

Pfahl, M.: 1990, *Numerische Mathematik in der gymnasialen Oberstufe*, Lehrbücher und Monographien zur Didaktik der Mathematik, BI-Wiss.-Verlag. Accessible at: `https://books.google.at/books?id=6YIIMQAACAAJ`.

Plangg, S..: 2019, TI-Nspire applet for iterative methods. Accessible at: `https://resources.t3europe.eu/t3europe-home/?resource_id=2708`.

Plangg, S., Milicic, G.: 2018, Die Numerische Mathematik im Kontext von Schule und Bildung, *BeitrĂge zum Mathematikunterricht* .

Schuppar, B., Humenberger, H.: 2015, *Elementare Numerik für die Sekundarstufe*, Springer Spektrum.

Velten, K.: 2009, *Mathematical Modeling and Simulation: Introduction for Scientists and Engineers*, Wiley. Accessible at: `https://books.google.de/books?id=_B1fbVnM5uQC`.

Goethe-University Frankfurt
e-mail: milicic@math.uni-frankfurt.de
Salzburg University of Education Stefan Zweig
e-mail: simon.plangg@phsalzburg.at