On-line interactive educational programs for students of engineering and natural sciences

Ivan Slapničar and Damir Krstinić
Faculty of Electrical Engineering, Mechanical Engineering
and Naval Architecture
University of Split
R. Boškovića b.b., 21000 Split, Croatia
E-mail: ivan.slapnicar@fesb.hr, damir.krstinic@fesb.hr

Abstract— **We present interactive programs for solving and visualizing the solutions of scientific and engineering problems. The programs are primarily intended for students, but can also be used at a high-school level. The programs plot functions (NetPlot), show animated convergence (FAni), perform Fourier analysis (FRed), and perform numerical computation and visualization (Octave On-line). Programs have an intuitive easy-to-use graphical user interface, and are fully documented with the on-line help. The programs are accessible at `http://www.fesb.hr/mat1` and `http://bolek.fesb.hr/czr/en/on-line.php3`. The programs were developed within several projects financed by the Croatian Ministry of science and technology.**

I. Introduction

WHILE traditional teaching methods can provide students with theoretical base, engineering ability, which requires a lot of intuition, can be developed only by solving number of examples. In traditional classroom it is hard to give every student the attention she or he needs. Beside that, only limited number of examples can be solved, and it is hard to visualize solution and solving techniques.

To meet these requirements, we have developed four programs for solving and visualization the solutions of scientific and engineering problems. These programs are primarily intended for students, but can also be used at high-school level. All programs have an intuitive and easy-to-use graphical user interface and are accessible over Internet. By using this tools, teacher can visualize examples to students, and students can explore different examples and immediately see visual and numerical solutions. By playing with examples, students can clarify and adopt knowledge in parts of theory they did not understand so good.

The paper is organized as follows: in Section II we describe basic technical requirements which are necessary for the realization of our programs, in Section III we give detailed description of programs and their usage, in Section IV we give more details about how the programs perform their tasks, and in Section V we give some concluding remarks.

II. Technical requirements

IN creating our programs we have used several tools, all of which are freely available on the Internet. On our servers we are using SuSE Linux 6.3 (see *http://www.suse.de*) operating system and Apache Web server (see *http://www.apache.org*). The hypertext preprocessor PHP [5] is used as connection between web server (user input) and executable code. PHP is a server side language, with syntax similar to C language, and convenient for automatic generation of HTML code. Computational part of the programs is developed in C and GNU Octave [2]. GNU Octave is a high-level language with syntax compatible to the one of Matlab.

For compilation of C code we have used the standard version of GNU gcc compiler which is part of the Linux distribution. Gnuplot [1] is used for creation of images. JavaScript [3] is used for some parts of user interface.

At the client side no special software is required, that is, any operating system and any newer web browser should work fine.

III. Program description

THE programs described here were developed within three projects for mathematical lessons on line, notably, pilot project of the Croatian Academic and Research Network CARNet "Mathematics 1" (see *http://www.fesb.hr/mat1*), and recently started I-projects of the Croatian Ministry of Science and Technology "Mathematics 2" and "Mathematics 3".

Two of our programs are general tools which can be used in teaching and solving wide range of mathematical and engineering problems (*NetPlot*, *Octave On-line*), and two (*FAni*, *FRed*) cover special areas of mathematical theory.

A. NetPlot

THE program *NetPlot* (*http://lavica.fesb.hr/netplot*) is an interactive tool for plotting functions. It is designed primarily for students of engineering and natural sciences. The user can plot and explore five most commonly used function types: functions of one variable, parametric functions of one variable, functions in polar coordinates, functions of two variables, and parametric functions of two variables.

The user can select function type, variable ranges and some other options which define how the function will be ploted. If the function is three dimensional, the user

can rotate the resulting image to get the desired view of the function in 3-D space.

NetPlot is actually a web interface to Gnuplot [1]. Although Gnuplot is an extremely powerful plotting tool, in order to make it simple to use *NetPlot* uses only limited number of Gnuplot options. However, the entire set of functions predefined in Gnuplot in available in *NetPlot*, as well. *NetPlot* has a short but complete Help-page which describes all features of the program. The Syntax-page describes all available constants, functions and operators. Further hints on how to use the program are given in the Examples-page.

As an example, in Fig. 1 we show how *NetPlot* plots the well known "sombrero" given by

$$f(x,y) = \mathrm{besj0}(\sqrt{x^2+y^2}+1),$$

where $\mathrm{besj0}(x)$ is the J_0 Bessel function which is predefined in Gnuplot, and x is given in radians. In this plot variable ranges are $x \in [-10,10]$ and $y \in [-10,10]$, mesh is 40×40, and the hidden 3-D surfaces are not displayed.

B. FAni

THE acronym *FAni* stands for "Function Animation". *FAni* (*http://lavica.fesb.hr/FAni*) displays the animated convergence of the given series of functions (Taylor series, Fourier series, ...).

The usage of the program is very simple. The only parameter that must be supplied is the n-th (general) term of the given series of functions, $g_n(x)$. You should keep in mind n starts from 1, that is, $n = 1, 2, 3, \ldots$. Since in some cases the first term of the series, or several starting terms, cannot be described with the general rule which describes all other terms, the starting terms can be entered in a separate field. This field can also remain empty. The user change the default ranges of the variables, where the independent variable x is marked red, and the dependent variable y is marked blue. The ranges are entered from-to and the dependent variable ranges can be left empty. The user can enter the sum of the series, if it is known. In this case the animation of the series towards the desired function can be monitored. The user can define how output will look like by combination of several other options.

Similar to *NetPlot*, *FAni* is also basically an interface to Gnuplot [1]. Upon pressing the `Animate` button, a CGI script is called which forms a command file for Gnuplot. Then Gnuplot is called in the background, and it produces PNG plots of first 10 or 20 partial sums of a given series of functions. Finally, these plots are joined together in the animated GIF which is sent back to the browser.

FAni also has a short Help-page which describes all features of the program, the Syntax-page which describes all available constants, functions and operators, and the Examples-page which gives four useful examples on how to run a program. All examples are active and can be tried out.

C. FRed

THE acronym *FRed* stands for "Fourierov red" which is the Croatian translation of "Fourier Series". Due to the rather special field of mathematics it covers, *FRed* is primarily intended for the sophomore students and engineers. *FRed* (*http://bolek.fesb.hr/FRed*) computes the coefficients of the Fourier expansion of the periodic function $f(x)$ which is defined on the interval $[a,b]$ and then periodically extended to the real axis. The Fourier expansion of f is given by [4, §6.12]

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{\infty} a_k \cos\frac{k\pi x}{L} + b_k \sin\frac{k\pi x}{L},$$

where $L = (b-a)/2$. The Fourier coefficients a_k and b_k are given by

$$\begin{aligned} a_0 &= \int_a^b f(x)dx, \\ a_k &= \int_a^b f(x)\cos\frac{k\pi x}{L}dx, \\ b_k &= \int_a^b f(x)\sin\frac{k\pi x}{L}dx. \end{aligned}$$

The above integrals are numerically computed with the trapezoidal rule [4, §7.2]: for $n+1$ equidistant integration nodes $x_0, x_1, \ldots, x_n$ where $x_0 = a$ and $x_n = b$, we have

$$\int_a^b f(x)dx = \frac{b-a}{n}\left(\frac{f(x_0)+f(x_n)}{2} + \sum_{i=1}^{n-1} f(x_i)\right).$$

The program is used as follows. User is provided with the input form for entering required data. By selecting `Graph` button, the input is checked for errors and properly formated, and the Octave m-file which performs the actual computation is called. Since the integrals are computed with the trapezoidal formula, the results are more accurate if the required number of series' terms m is sufficiently small with respect to the number of integration nodes n.

The primary output is graphical, consisting of two images. The first image displays the graph of the given function and its approximation with the first m terms of the Fourier series. The second image displays the distribution of the first m harmonics $h_k = \sqrt{a_k^2 + b_k^2}$. The user can also choose to display numerical values of the computed coefficients. Notice that the number of integration nodes is limited to 999, and the number of coefficients is limited to 99. However, these limitations can easily be extended if needed.

As an example, in Fig. 2 we show the computation of the first 41 Fourier coefficients ($m = 20$) for the function $f(x) = \sqrt{(}\sin(x))$ on the interval $[0,2]$ by using 400 integration nodes.

Illustrating numerical way of obtaining Fourier coefficients is important, since in this manner students can readily see the Fourier transform at work, without having to solve tedious integrals. Even more, since the

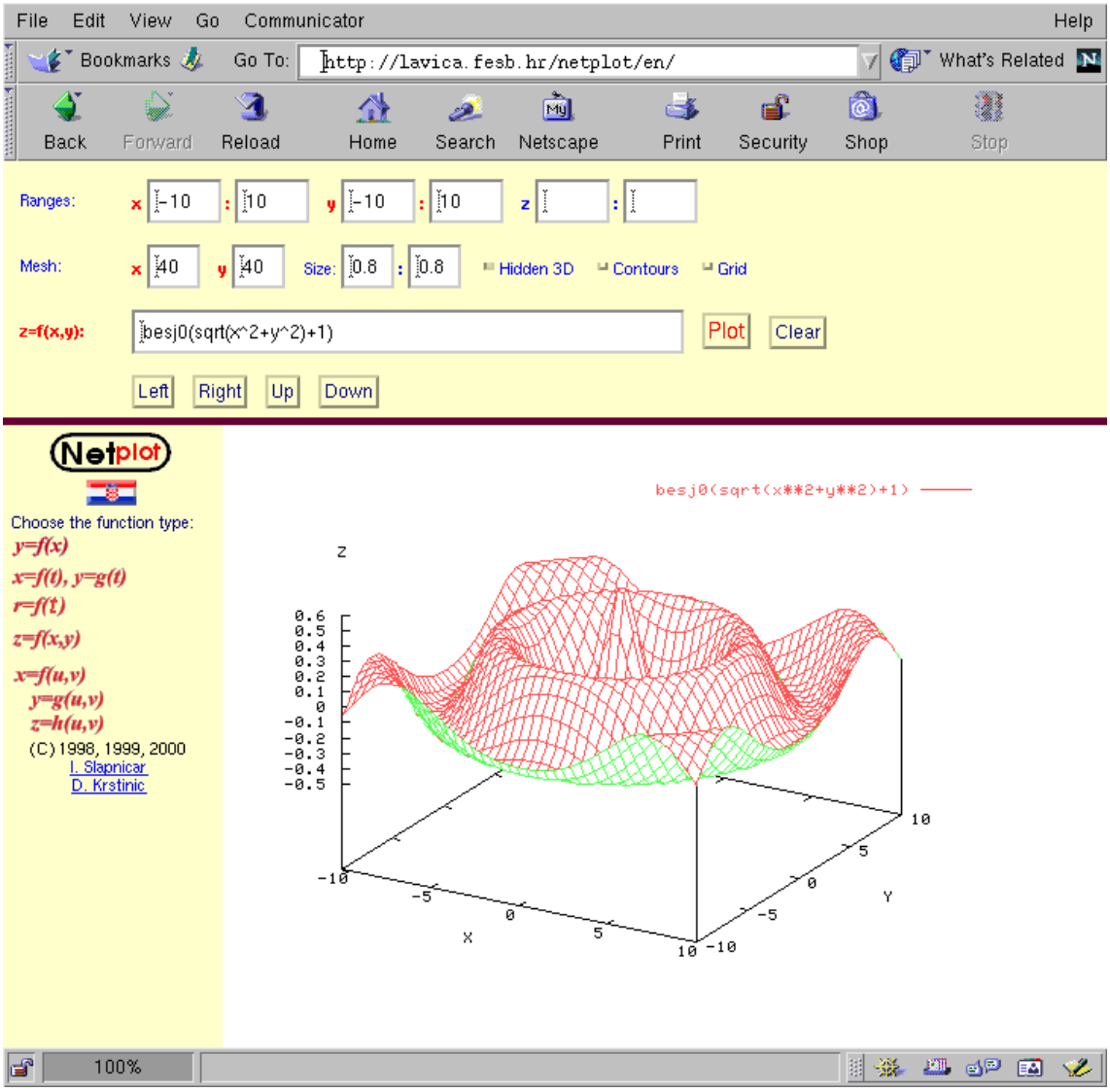

Fig. 1. "Sombrero" plotted by *NetPlot*

program uses numerical integration, the Fourier coefficients can be computed if the above integrals are not elementary solvable, as well. Notice that an example of the m-file which performs the computation is also available on-line.

D. Octave On-line

GNU Octave [2] is a high-level language primarily intended for numerical computations. In the on-line version (*http://bolek.fesb.hr/czr/en/on-line.php3*), the user can directly enter Octave commands in the command-input field, or upload already prepared Octave program files (the so called m-files) from the user's hard disk. After uploading, the uploaded file appears in command-input field and can be edited again. When the editing is over, the work can be saved to the user's hard disk again.

After pressing the `Compute` button, the commands in the command-input filed are executed, and the output is presented in a separate window. Output of the program contains both text and images mixed in the same window. All of the numeric (textual) output can be saved to automatically generated local file. Also, the user can change parts of the program in the command-input filed, and start the computation again. For the ease of use, the user has on line access to the complete Octave manual in two forms, through the GNU Info System and in the HTML version. A help interface for short help on particular command is also provided.

As GNU Octave is fully featured programming language, designed for mathematical computations, *Octave On-line* can be used for practical teaching of almost any part of applied mathematical theory, depending of teacher and students needs. The command-input filed can also be easily embedded in other teaching materials, thus providing students with the opportunity to immediately check on some practical examples what they have learned so far.

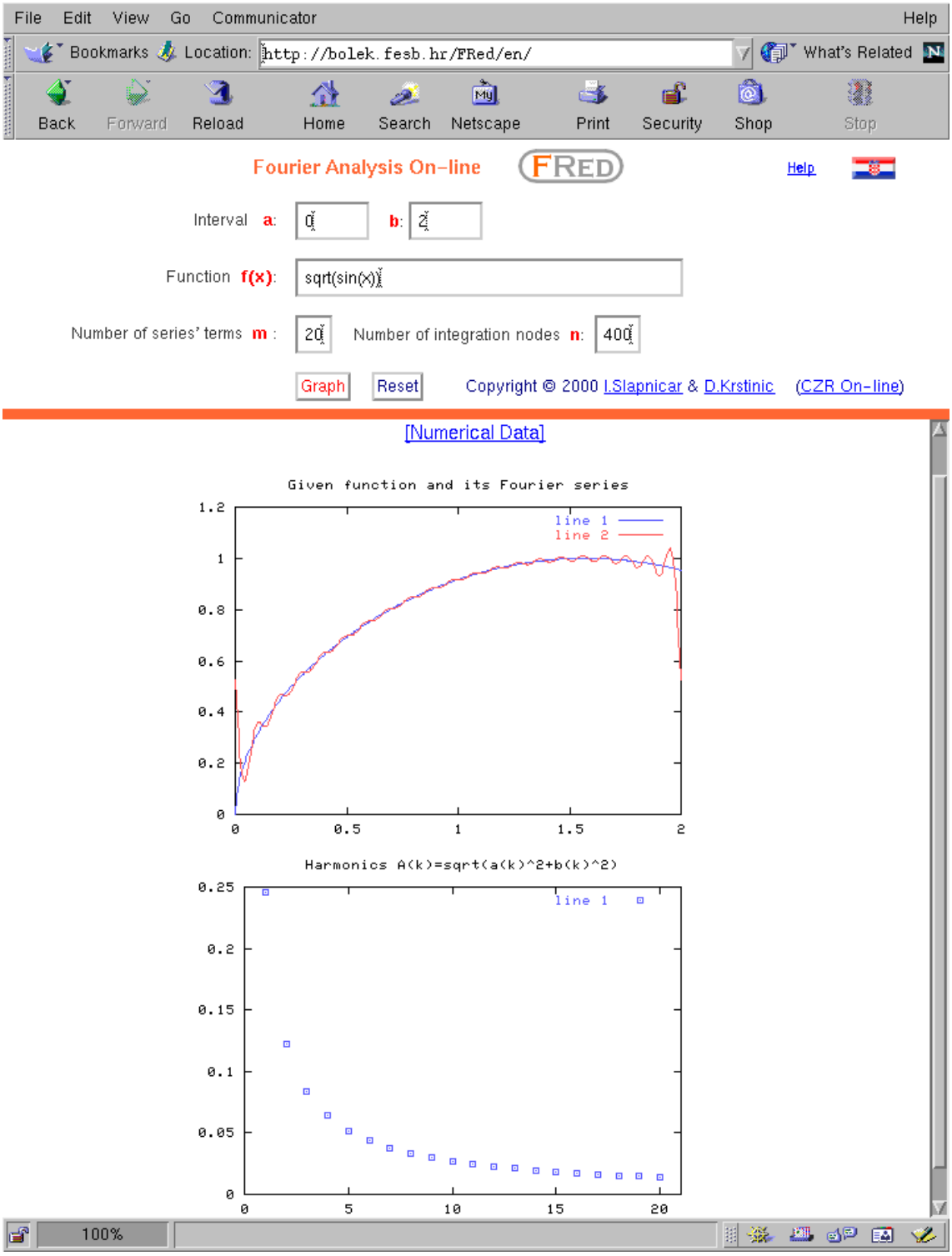

Fig. 2. Computation of Fourier coefficients

Although the fact that the student must be well acquainted with the GNU Octave programming language might be considered as a slight disadvantage, the benefits of being able to perform almost any numerical computation make it worth while.

Since the program is being executed on the server, and the user is provided with the fully featured programming language this application has necessary security checking in order to avoid the possible attacks.

IV. Principles of execution

THE four described programs differ slightly in a manner in which they perform their tasks.

NetPlot and *FAni* are executed by means of classical

CGI scripts written in C. When the execution starts, the CGI script reads the user input from the browser, and creates the command file for Gnuplot by using this data. Then the Gnuplot is started in the background by means of the Unix *popen* function, and the created command file is simply written to that pipe. Output of Gnuplot (the image file) is then saved to the hard disk. Finally, the CGI script writhes this image file back to the browser. Since the image file is temporarily written to the disk, the files that belong to different users must be distinguished. This is done by generating the unique session identifier for each user (usually the process identifier PID), and the image file and other temporary data are saved under the name which contains this identifier.

The principles of running *FRed* and *Octave On-line* are as follows. When the user opens the program site, the user interface for data input is generated. When the user is satisfied with the input data, by selecting `Graph` or `Compute` button, respectively, a PHP script is started. This script first performs input data checking in order to detect invalid input data or possible security attacks. Then, the script calls Octave which performs actual computation and/or data visualization. All of the Octave output is redirected to the calling PHP script, and with that data, HTML user output is generated. Notice that such design is convenient since the output is written directly to the browser without need for temporary data storage on the server's hard disk.

V. Conclusion

We have developed technology which enables us to write various programs for computing numerical solutions and visualization of solutions of problems in mathematics or applied mathematics. With help of our programs students can easily solve mathematical problems, in particular those arising in engineering, and therefore speed up their learning process. The programs can equally well serve as an aid to teachers in their classrooms or computer labs. Besides the described programs, the developed technology can be used to create other similar programs in various areas of applied mathematics, such as numerical integration, numerical solution of differential equations, applications of the finite element method, mathematical modeling of electrical, mechanical and environmental phenomena, etc.

Acknowledgments

We would like to thank Professor Krešimir Veselić, Fernuniversität Hagen, Germany, for encouraging us in writing Octave On-line. We also thank John W. Eaton from the Department of Chemical Engineering, University of Wisconsin–Madison, and Tom Weichmann, Buffalo State College, for their comments regarding security issues in Octave On-line.

References

[1] Thomas Williams and Colin Kelley, *Gnuplot, An Interactive Plotting Program*, 1998, `http://www.gnuplot.org`.

[2] John W. Eaton, *GNU Octave*, 1997, `http://www.octave.org`.

[3] *JavaScript Guide*, Netscape Communications Corp., 2000, `http://developer.netscape.com`.

[4] David Kincaid and Ward Cheney, *Numerical Analysis*, Brooks/Cole, Pacific Groove, 1996.

[5] Stig S. Bakken, *PHP Manual*, PHP Documentation Group, 2000, `http://www.php.net`.