Updates Concerning Numerical Calculus Through Numerical Engineering Software

Romulus Militaru *, Liviu-Adrian Călin,**
George-Cristian Călugăru *, Adrian-Lorel Georgescu*****

**Department of Applied Mathematics, University of Craiova, Romania (e-mail: militaruromulus@yahoo.com)*
***Faculty of Mathematics and Computer Science, University of Craiova, Romania (e-mail:adi.calin.nds@gmail.com)*
****Faculty of Automatics, Computers and Electronics, University of Craiova, Romania (e-mail: {calugaru.george.nds , georgescu.adrian.nds}@gmail.com)*

Abstract: Numerical Engineering Software is a dedicated solution for numerical calculations. The project contains many numerical methods that have been optimized and can represent the basis for numerical solving of real life technical problems. An earlier version of Numerical Engineering Software was presented in the previous number of this journal. This paper is dedicated to the presentation of the new abilities of Numerical Engineering Software which can appeal to a larger number of users and can result in improved processing of data. The new version consists of a slightly modified structure, mainly regarding the fifth chapter which now deals with systems of differential equations and Sturm-Liouville problems. The project consists of five chapters: Matrix Algebra, Polynomial Approximations, Roots of Equations, Numerical Integration and Differential Equations.

Keywords: numerical calculus, linear systems, polynomial interpolation, eigenvalues and eigenvectors, differential equations, mathematical software

1. DESCRIPTION OF NUMERICAL ENGINEERING SOFTWARE

Numerical Engineering Software (NES) is a project dedicated to solving numerical analysis problems which can relate to real-life technical problems. The project has evolved through several stages of development starting with a prototype and ending with the current version. The prototype consisted of a small project affected by many restrictions of the implemented methods which were very few and its sole purpose was to serve as an experiment regarding the interface. The software solution we propose must have a user-friendly interface which needs no prior programming skills and even has a Romanian version in order to appeal to a larger number of users that need to solve technical problems with the least amount of effort possible. Once the frame of the interface was set and a pattern emerged, several development stages occurred focused on selection and optimization of methods, Militaru (2009). For instance, we figured that the input data should be inducted in the simplest way possible. On this subject we experimented with a text editor resulting in the ability to store data in special „.nes" extension files, „nes" standing for Numerical Engineering Software.

Numerical Engineering Software runs as a cross platform application with proficient results on Windows, Solaris and Linux.

Strictly regarding the numerical algorithms implemented the first idea of optimization we had was imposing the precision of calculation for the processed data. We achieved just that in most of the implemented methods reaching precisions of up to 15 decimal places depending on the given problem.

The next aspect of optimization was improving the computational cost of the algorithms selected for implementation. In some cases, we had to choose one method for solving one specific problem from a range of methods dealing with that specific problem and in this case the selection was based on inherent computational cost. In other cases, the method used for approaching a specific problem was subject to improvements regarding its computational cost.

For more complex problems such as superior order differential equations we developed our one procedures starting from a 4-th order Runge-Kutta algorithm which is widely used in practice, Burden (2004), Philips (1999). The same algorithm was used for the development of the system of differential equations procedure.

One of the most particularized methods which is now implemented in Numerical Engineering Software is the method regarding Sturm-Liouville problems with the mention that the methods deals with all types and coefficients.

The last stage of optimization was the invention of numerical methods to deal with specific problems. In this case, so far, we developed the Militaru method for the calculation of extreme eigenvalues of a real, square, symmetric matrix, Militaru (2006).

These were the principles of optimization used to improve the quality and appeal of Numerical Engineering Software. A section of the present paper will be dedicated to presenting the improvements and particularizations of Numerical Engineering Software.

Further development sessions will take place focusing on transfer function processing, non-linear equations and systems of equations and improvements regarding general appearance and graphic visualizations.

2. NUMERICAL METHODS IMPLEMENTED IN NUMERICAL ENGINEERING SOFTWARE

2.1 Matrix Algebra

The notion of matrix is one of the fundamental elements of mathematics. The processing of matrices is essential in many calculations that find applicability in many areas from computer science to engineering.

The calculation of a linear algebraic system can be used for the solving of problems depending on a finite number of freedom degrees, represented through ordinary differential equations or through equations with partial derivates are transformed with the help of finite differences into linear systems. This type of systems can also be used in electrical wiring studies, structure analysis, building projects etc.

The part of control engineering that relies on the study of systems as abstract entities is called system theory. According to system theory, Marin (2007), there are many types of systems: continual, discrete, with distributed parameters, with concentrated parameters etc. An important aspect in system theory is the study of the properties of a system. In the case of a linear system, the study of stability can be performed, in this case stability being a system property. In the case of non-linear systems, stability ceases to be a system property and is studied for every evolution. To study the stability of a system we have two types of criteria: algebraic and frequency related. Algebraic criteria are applied on the characteristic polynomial and stability is demonstrated through solving determinants. The two main algebraic criteria are called Routh and Hurwitz and are based on the solving of determinants, respectively of the Routh table. Other properties of systems that can be determined by solving determinants are the controllability and observability properties.

Equilibria can be studied with the help of the signs of the eigenvalues belonging to the linearization of the equations about the equilibria. That is to say, for every equilibrium point the Jacobian matrix must be evaluated, and then, after finding the resulting eigenvalues, the equilibria can be categorized. Then the behaviour of the system in the neighbourhood of each equilibrium point can be determined from the quality point of view, (or even quantitatively determined, in some instances, by finding the eigenvector(s) associated with each eigenvalue). An equilibrium point is hyperbolic if none of the eigenvalues have zero real part. If all eigenvalues have negative real part, the equilibrium is a stable node. If at least one has a positive real part, the equilibrium is an unstable node. If at least one eigenvalue has negative real part and at least one has positive real part, the equilibrium is a saddle point. If all eigenvalues are identical null we have an equilibria line.

The methods included in the first chapter are:

1. Gauss elimination method of inverting a matrix;

2. iterative method of inverting a real matrix;

3. iterative Seidel-Gauss method for large real linear systems;

4. iterative Seidel-Gauss method for real sparse matrix linear systems;

5. LR factorization for solving a real linear system;

6. LR factorization for solving real tridiagonal matrix systems;

7. LR factorization for solving real pentadiagonal matrix systems;

8. Fadeev method for determining the characteristic polynomial of a real square matrix;

9. Danilevski method for estimating the eigenvalues and corresponding eigenvectors for a real square matrix;

10. LR method for the estimating the eigenvalues of a real square matrix (covering all particular cases);

11. Militaru method for the calculation of the extreme eigenvalues of a real symmetric matrix (based on successive approximations);

12. Krylov method for the calculation of the characteristic polynomial;

13. Leverrier method for the calculation of the characteristic polynomial;

14. Jacobi method for estimating the eigenvalues of a real symmetric matrix;

15. QR factorization for solving a real linear algebraic system;

16. LR method for estimating the eigenvalues of a real square matrix;

17. QR method for matrix transformations;

18. Matrix transformations through a triangularization procedure;

19. A triangularization procedure for solving a real linear algebraic system;

20. A pivotal condensation method for the numerical calculation of a real square matrix determinant.

This chapter has considerably grown in size. From the previous number of the journal, methods 12 to 20 have been implemented in this chapter and another method for the calculation of the pseudo-inverse of a matrix is currently being refined and prepared for implementation. The Matrix Algebra chapter of NES approaches a wider range of problems and offers true support regarding accuracy of results.

2.2 Polynomial Approximations

In control engineering in general and in signal processing (SP) in particular we can view the interpolation table as a series of samples acquisitioned from a signal with a given sampling period. The sampling frequency must be at least double the size of the signal frequency. In practice, the sampling frequency is sometimes more than tens of times bigger. If the sampling frequency is less than double than the frequency of the signal, then the so called aliasing phenomenon occurs. The aliasing phenomenon can be interpreted as ambiguity in the frequency domain. Looking at it from a systemic point of view, the aliasing phenomenon represents a pole expansion. Marin (2007)

The sampling process is the basis for numerical computers and automation equipment today, mostly because a sampled signal is easier to use and store.

Given an interpolation table, values from the first column (x) can be interpreted as sampling moments and the second column (f(x)) can be interpreted as the values of a signal a certain moment in time. It is recommended that the sampling period be uniform.

Using Numerical Engineering Software you can take a table of such samples, insert it in a special extension ".nes" file characteristic to the software, process it and view the original shape of the signal or determine the value of the signal at another moment in time, all of them with an imposed accuracy.

This is how the interpolation procedures can be used as a solution for a technical problem.

The methods included in this chapter, Chatelin (1983), Leader (2004), Militaru (2008):

1. Lagrange interpolating polynomial;

2. Discrete least square approximation;

3. Newton interpolating polynomial;

4. Cubic Spline with free boundary;

5. Cubic Spline with clamped boundary.

The graphic approximation section is very flexible and allows 2D graphs to be configured with the help of several options. These options allow the user to select the colour and step length of the represented graphic profile.

This chapter doesn't contain any new methods but significant improvements have been made to the ones already implemented. The Newton and Lagrange interpolating polynomials now posses the ability of imposing precision of calculation which is the ultimate improvement regarding interpolation procedures, Militaru (2003).

2.3 Roots of Equations

Methods for finding roots of equations are basic numerical methods.

Over the centuries, mathematicians have developed a variety of methods of solving equations. Numerical methods for locating roots of equations can often be easily programmed. Using the capabilities of modern computers, one can explore in detail these age-old recipes (capabilities, restrictions, speed of convergence).

The methods included in this chapter are, Demidovici (1973), Ebâncă (2005):

1. Bairstow method for the calculus of the roots (real or complex) of an algebraic equation;

2. Bernoulli root finding method.

The user has the possibility to impose the precision of the estimations of the roots.

The Bernoulli root finding algorithm is newly inducted in Numerical Engineering Software.

2.4 Numerical Integration

The need to approximate the definite integral of a function often arises in the case when the function has no antiderivative or the antiderivative is difficult to manipulate numerically, and also in the case of a function known only for her values in a discrete set of data points, whose antiderivative is no longer possible to determine.

From the control engineering point of view simple integrals can define optimization or robust control criteria which have to be minimized or maximized depending on the type of problem:

The methods included in this chapter are:

1. Newton integration method for the numerical evaluation of a simple integral;

2. a method for the numerical evaluation of a double integral over a measurable convex domain with polygonal boundary.

The development of this chapter has been finalized by improving the method for evaluating double integrals over measurable convex domains with polygonal boundary. The method now contains a procedure which allows imposing the accuracy of the calculus and now the method enjoys proficient results.

2.5 Differential Equations

Differential equations play a major role in scientific calculations and mathematical representations.

In system theory, linear differential equations with constant parameters are used to describe linear time invariant SISO (Single Input-Single Output) systems. Systems of ordinary differential equations can describe linear time invariant MIMO (Multi Input-Multi Output) systems, Marin (2006). Partially differential equations are used to describe distributed parameters systems.

Differences equations are used to describe discrete systems.

In control engineering, system engineers deal mostly with non-linear phenomena. However, in some conditions a non-linear mathematical model can be very good approximated through a linear model, Marin (2007).

Ordinary or partially differential equations represent mathematical models for the majority of mechanical and electrotechnics engineering problems: the study of efforts to which resistance elements are exposed: bars, pillars, thin plates, thick plates, pipes; the study of electrical fields in dielectrics problems, magnetic and thermal fields, the propagation of all types of waves and the list goes on.

Once the physical phenomenon and the differential equations that govern it, boundary conditions and coefficients are stated, only one issue remains: the solving of this mathematical model. For various reasons: physical differences, boundaries with an unusual geometry etc., the solving of this mathematical model will consist of the search for an approximate solution using numerical computation.

The methods included in this chapter are, Militaru (2008), Popa (2010), Press (2007):

1. Euler method for the solving of a Cauchy I-st order problem;

2. Runge-Kutta 4-th order method for solving Cauchy I-st order problems;

3. a Runge-Kutta 4-th order based method for the solving of superior order differential equations;

4. a Runge-Kutta 4-th order based method for the solving of systems of differential equations;

5. a finite difference method for solving a Sturm-Liouville problem.

Since the previous number of this journal, methods 3 to 5 have been implemented successfully. Because of the implementation of the method for dealing with Sturm-Liouville problems the original name of the chapter of Cauchy Problems has been extended to that of Differential Equations.

This chapter has been, by far, the most challenging part of the development of Numerical Engineering Software. We developed specialized numerical functions based on methods like the Runge-Kutta 4-th order, for the evaluation of high order differential equations and systems of differential equations.

In the case of Sturm-Liouville problems the implemented method is based on finite differences and is appropriate for every type of boundary conditions (Neumann, Dirichlet or mixed).

The way of inducting data is very simple and natural, constituting an advantage of Numerical Engineering Software.

3. THE INTERFACE OF NUMERICAL ENGINEERING SOFTWARE

As we described in the previous number of this journal, Militaru (2009b), the interface of Numerical Engineering Software consists of dedicated application windows. Each application window is composed of drop-down menus and option buttons. Drop down menus are generally placed in an application window to select the method of calculation. The rest of the options are present because they represent parameters or restrictions of a given method.

Each particular application window emerged from a need to bring the options closer to the user.

For ease of understanding and for better display of the benefits of Numerical Engineering Software we will focus mainly on the application window present in the Romanian version of the software.

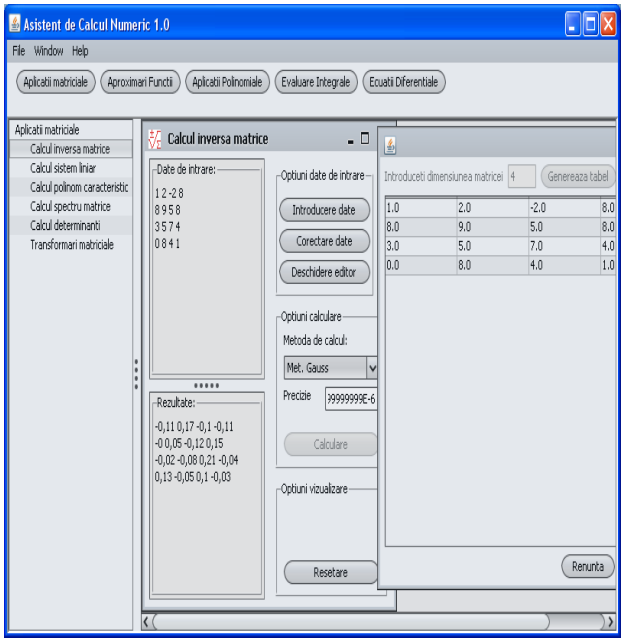

Fig. 1. Calculating the inverse of a matrix using NES

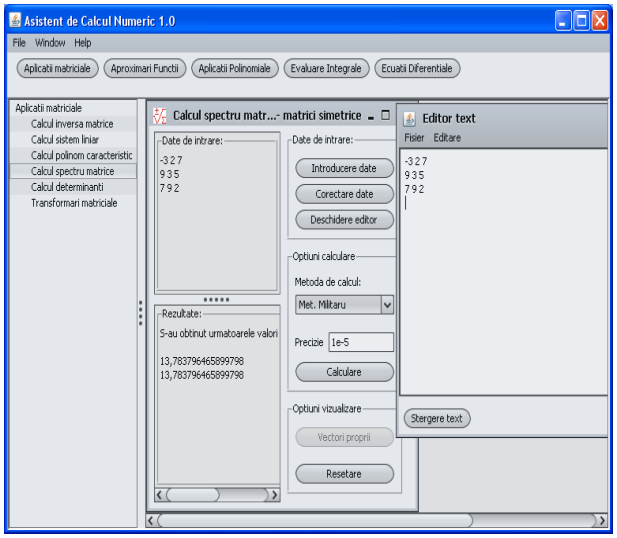

Fig. 2. Militaru method for the estimation of the extreme eigenvalues of a real symmetric matrix

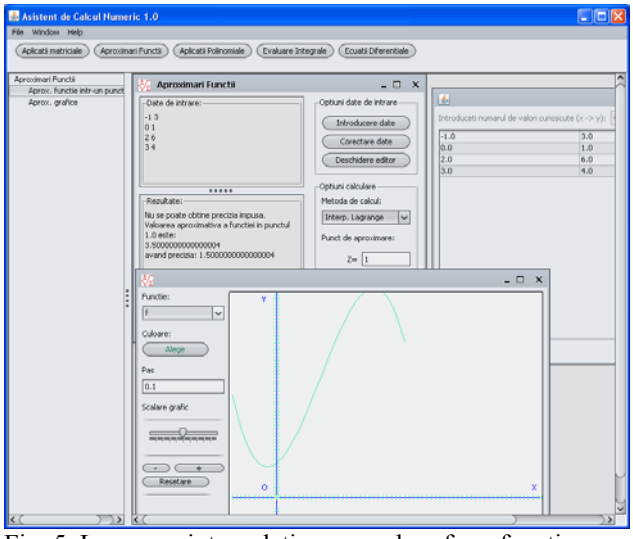

Fig. 5. Lagrange interpolation procedure for a function together with the graphic profile visualization.

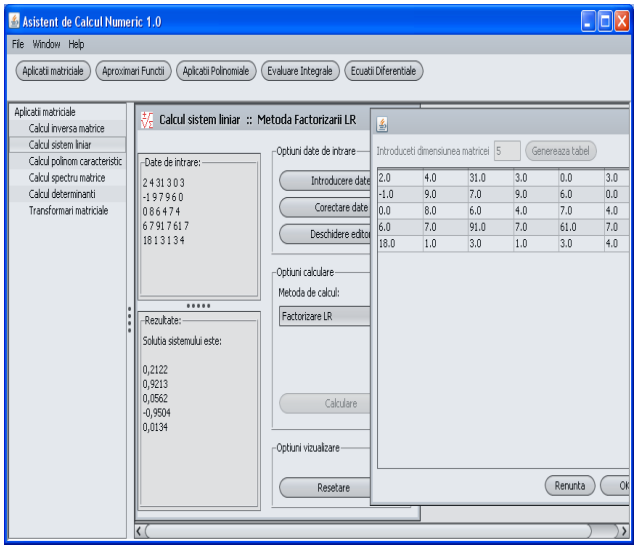

Fig. 3. Solving a linear algebraic system through LR factorization

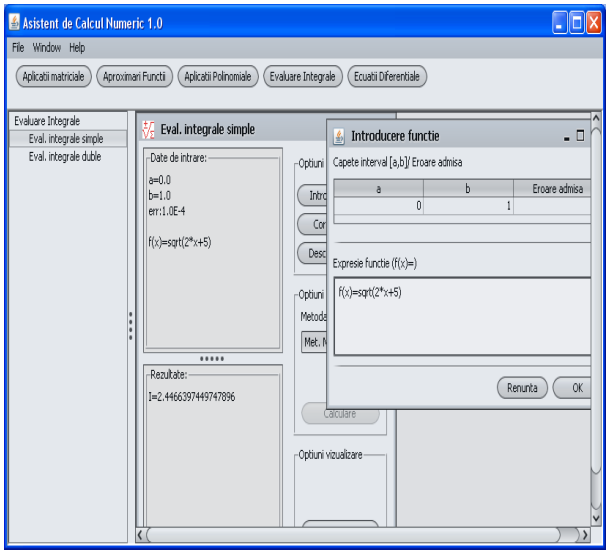

Fig 6. Evaluating a simple definite integral using NES

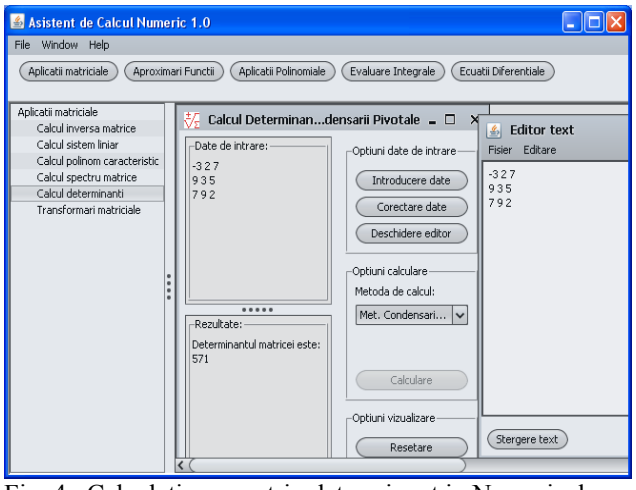

Fig. 4 . Calculating a matrix determinant in Numerical Engineering Software using the .nes files

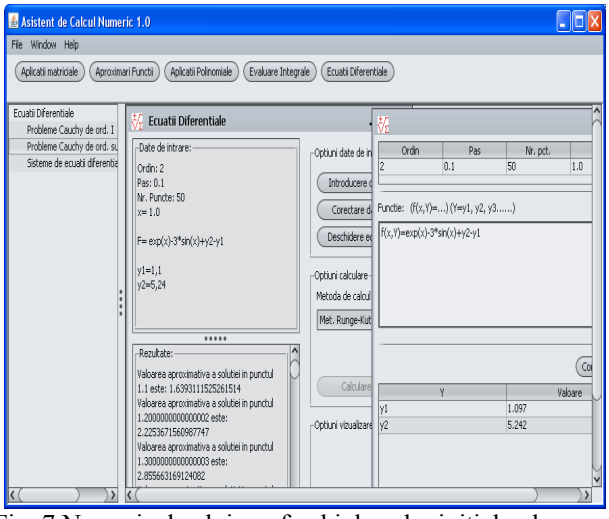

Fig. 7 Numerical solving of a high order initial value problem using NES

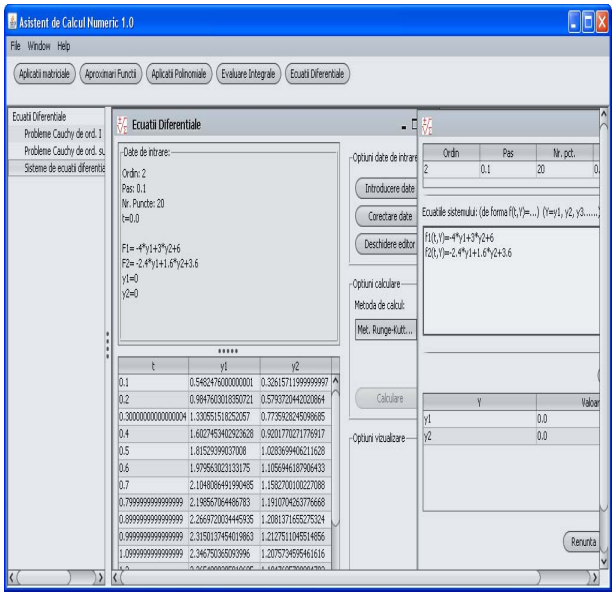

Fig. 8. Numerical Solving of a system of ordinary differential equations

4. OPTIMIZATION IN NUMERICAL ENGINEERING SOFTWARE

Optimization in the version of Numerical Engineering Software presented in the previous number of this journal resumed to the following aspects, Militaru (2009b):

1. Initial value problems for high order differential equations are solved numerically through a compact procedure based on a Runge-Kutta IV-th order method, depending only on the order of the equation and the integration step length imposed by the user.

2. Runge-Kutta method (IV-th order) allow the display of the approximate values of the exact solution belonging to an initial value problem for ordinary differential equations with an integration step length selected by the user. The algorithm has been improved in order to approximate the values of the exact solution of the given Cauchy problem, within a given accuracy, imposed by the user. The computational cost of algorithm is minimal.

3. Danilevsky method determines the coefficients of the characteristic polynomial, the eigenvalues and the corresponding eigenvectors for any real square matrix. All the particular cases are taking into consideration, the algorithm having a minimal cost of computation.

4. LR factorization method allows the estimation of the eigenvalues of a real square matrix. The algorithm gives the possibility of working with a given precision. It is complete with all particular cases and has a minimal cost of computation.

5. Militaru method, Militaru (2008), allows the numerical approximation of the extreme eigenvalues of a real symmetric matrix with a given precision, by a trace computation. The algorithm avoids the determination of the coefficients of the characteristic polynomial of the given matrix, or the use of similarity transformations, with the purpose of eliminating the intermediate stages of calculation which lead to numerical instabilities, contributing in the decrease of the amount of work. Thus, the algorithm benefits from an optimum cost of computation.

6. Lagrange polynomial interpolation allows the approximation of a function f given only for a discrete set of points which make up a sample. The algorithm evaluates the value of the function in a given data point z, based on a Lagrange interpolating polynomial, having incorporated a procedure which offers to user the possibility to impose the degree of the interpolating polynomial, Militaru (2003). Thus, the algorithm exploits better the results of calculus, contributing to a decrease in the amount of work involved, respectively the computational cost necessary to approximate $f(z)$ within a given accuracy. One can also visualize the approximate profile of the function with an option for scaling the graph and several other appearance options.

The same characteristics are also valid for the case of Newton polynomial interpolation.

Further aspects of optimization include:

- Design of a compact procedure for the calculus of systems of differential equations, RK4 based.

- Design of a compact procedure for solving of Sturm-Liouville problems.

- Lagrange and Newton interpolating polynomials now require precision of calculation instead of the degree of the approximating polynomial.

- Double integral evaluation includes now precision requirement.

- Several modifications in matters of interface.

5. NUMERICAL EXAMPLES

5.1 Electrical Engineering

The currents $i_1(t)$ and $i_2(t)$ in the left and right loops of a closed circuit containing a resistance, a capacitance, an inductance and a voltage source, verify the following 2-nd order initial value problem:

$$\begin{cases} i_1'(t) = -4i_1(t) + 3i_2(t) + 6 \\ i_2'(t) = -2.4i_1(t) + 1.6i_2(t) + 3.6 \\ i_1(0) = 0; i_2(0) = 0 \end{cases}$$

Assuming that the switch in the circuit is closed at time t=0, we are looking to estimate the values of the currents $i_1(t)$, $i_2(t)$ at the moments $t_i = 0.1 \cdot i,\ i = \overline{1,9}$.

Using NES we obtain the following approximations:

Table 1. Numerical evaluations and errors for the system of differential equations

t	$i_1(t)$	$i_2(t)$	error1	error2
0.1	0.5482	0.3261	0.005	0.002
0.2	0.9847	0.5793	0.007	0.003
0.3	1.3305	0.7735	0.008	0.004
0.4	1.6027	0.9201	0.008	0.005
0.5	1.8152	1.0283	0.009	0.005
0.6	1.9795	1.1056	0.009	0.005
0.7	2.1048	1.1582	0.007	0.004
0.8	2.1985	1.1910	0.006	0.004
0.9	2.1985	1.1910	0.006	0.004

(error1 and error2 represents the absolute value of differences between the exact values of the currents snd theirs numerical approximations).

5.2 General Engineering

1. We consider the steady state heat transfer process in a long, thin bar, and we assume that the cross-sections of the bar are uniform, and that the temperature varies only in the longitudinal direction. We also assume that the bar is perfectly insulated, except at the ends. Supposing the cross-section area $A = 10^{-2} m^2$ and the length $L = 0.5m$, we intend to calculate the steady-state temperature of the bar. The left end is fixed at $300^0 C$ and the right end at $700^0 C$. We are looking to evaluate the temperature distribution along the rod, supposing a constant thermal conductivity k.

The steady-state temperature $T = T(x)$ verifies the following boundary value problem:

$$\begin{cases} -kT''(x) = 0, x \in (0, L) \\ T(0) = 300;\ T(L) = 700 \end{cases}$$

Observation: The exact solution is given by $T(x) = 800x + 300$.

Using NES and imposing a discretization of the interval $[0, L]$ into 10 equal subintervals one obtains:

Table 2. Numerical results for the above Sturm-Liouville problem

x_i	T_i
0.05	340
0.1	380
0.15	420
0.2	460
0.25	500
0.3	540
0.35	580
0.4	620
0.45	660

(T_i denotes the approximate value of the temperature in the point of abscise x_i).

One observes that the numerical results are in very good agreement with the exact ones.

2. Let the following boundary value problem:

$$\begin{cases} y''(x) = -\dfrac{2x}{1+x^2} y'(x) - \dfrac{1}{1+x^2},\ x \in (0;1) \\ y(0) + y'(0) = 0.4413 \\ 2y(1) - y'(1) = 0.2794 \end{cases}$$

We are looking to approximate the values of the exact solution $y(x)$ of the above problem in the points $x_i = i \cdot h,\ i = \overline{0, n}, h = 1/n, n \in \mathrm{N}^*$

We consider $n = 20$. Using NES we obtain

Table 3. Boundary value problem results and errors for the given Sturm-Liouville problem

x_i	y_i	*error*
0.05	0.0211	0.0003
0.1	0.0393	0.0003
0.15	0.0549	0.0003
0.2	0.0678	0.0003
0.25	0.0781	0.0003
0.3	0.0858	0.0003
0.35	0.0910	0.0002
0.4	0.0940	0.0003
0.45	0.0946	0.0002
0.5	0.0932	0.0002
0.55	0.0899	0.0002
0.6	0.0849	0.0002

0.65	0.0783	0.0002
0.7	0.0702	0.0001
0.75	0.0609	0.0001
0.8	0.0505	0.0001
0.85	0.0391	0.0001
0.9	0.0268	0.0001
0.95	0.0137	0.0001

Observations:

(i). $error$ represents the absolute value of difference between the exact value of $y(x)$ and the numerical approximation;

(ii). The exact solution is given by

$$y(x) = 0.4413 arctg(x) - 0.5\ln(1+x^2)$$

5.3 General Engineering

We intend to evaluate the following double integral $I = \iint_D \arcsin\sqrt{x+y}\,dxdy$, where the domain D is the parallelogram given by the vertices $M(-1;1)$, $N(1;-1)$, $P(2;-1)$, $Q(0;1)$.

Using NES one obtains: imposing an accuracy of $\varepsilon = 10^{-3}$, $I = 1.570774$; imposing an accuracy of $\varepsilon = 10^{-4}$, $I = 1.570797$.

Note: The exact value is $I = 1.5707963$

6. CONCLUSIONS

Numerical Engineering Software has a steady growth rhythm and approaches a various range of numerical methods with value to real-life technical problems.

We emphasis that one of the crucial advantages of Numerical Engineering Software is the easy to use interface which facilitates the production of accurate results over a minimum time span. This means that a user has to be familiar only with the restrictions and input data characteristic to every method.

Precision of results is another advantage of NES, most methods being equipped with the possibility of imposing accuracy of the results to be obtained.

Strong components of the software project are the highly informative error messages which automatically point out the appropriate course of action.

Future development areas include non-linear equations and systems of equations and considerable improvements to the overall general appearance with emphasis on increasing the quality of graphical representations.

REFERENCES

Burden, R.L., Faires, J. (2004), *Numerical Analysis*, Brooks Cole.

Ciarlet, P.G. (1990), *Introduction à l'Analyse Numérique et l'Optimisation*, Ed. Masson, Paris.

Chatelin, F.(1983), *Spectral approximation of linear operators*, Academic Press, New York.

Demidovici, B., Maron, I. (1973), *Elements de Calcul Numérique*, Ed. Mir Moscou.

Ebâncă, D. (2005), *Metode numerice in algebră*, Editura Sitech, Craiova.

Ionescu, G., Ionescu, V., s.a (1987), *Automatica de la A la Z,* Ed. Stiințifică şi Enciclopedică, Bucureşti.

Leader, J.J. (2004), *Numerical Analysis and Scientific Computation*, Addison-Wesley.

Marin, C., Petre, E., s.a. (2006), *System theory problems*, Editura Sitech, Craiova.

Marin, C., Popescu, D. (2007), *Teoria sistemelor si reglare automata*, Editura Sitech, Craiova.

Mellor-Crummey, J., Garvin, J. (2004), *Optimizing sparse matrix vector product computations using unroll and jam*, International Journal of High Performance Computing Applications, 18(2), pg. 25-236.

Militaru, R. (2008*), Méthodes Numériques. Théorie et Applications*, Ed. Sitech,Craiova.

Militaru, R. (2006), *On the Newton's iterative method for the characteristic equation of a real symmetric matrix*, IEEE Computer Soc., Eighth International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC 2006), pg. 175-180.

Militaru, R. (2003), *A polynomial interpolation algorithm for estimating a numerical function*, Annals of University of Craiova, Math. Comp. Sci. Ser., Volume 30(2), pag. 1-7.

Militaru, R., Calin, L.A., Calugaru, G.C., Georgescu, A. (2009), *Efficient Computer Assisted Numerical Calculus through Numerical Engineering Software* , Proceedings of the Conference on Applied and Industrial Mathematics CAIM 2009, Constanta.

Militaru, R., Calin, L.A., Calugaru, G.C., Georgescu, A. (2009b), *Numerical Engineering Software-A New Tool for the Computer Assisted Numerical Calculus, Annals of the University of Craiova, Series Automation*, Computers, Electronics and Mechatronics, vol. 6 (33), No. 1, pg. 59-64.

Philips, G., Taylor, T.(1999), *Theory and Applications of Numerical Analysis*, Academic Press.

Popa, M., Militaru R. (2010), *Metode numerice în pseudocod - aplicații*, Ed. Sitech, Craiova.

Press, H.W., Teukolsky, S.A., Veterling, T.W., Flannery, B. (2007), *Numerical Recipes*, 3-rd Edition, Cambridge University Press.