A least squares method for solving initial-boundary value problems

R.A. Walentyński
Department of Building Structures' Theory, Silesia University of Technology, PL44-101 Gliwice, Poland,
rwal@kateko.bud.polsl.gliwice.pl

Abstract

This paper presents the application of the refined least squares method. The refinement makes it possible to solve problems with not only boundary, but also initial and non-continuous conditions. *Mathematica* is used to develop algorithms and carry out computations. It enables us to extend fields of approximate analytical method applications and allow them to be regarded as computer ones. *Mathematica* makes it possible to solve unstable and ill-conditioned tasks which are too difficult for numerical methods.

1 Introduction

The problem of approximate solution of boundary value problems with *Mathematica* was already considered by Barrère & Carmasol [1, 2]. They applied the Galerkin method.

The least squares method is a well known method in mathematics. It is used to approximate data sets or functions with other functions or to approximate solutions of differential equations. Boundary conditions in traditional approach have to be satisfied by the approximating functions. The same conditions should be satisfied by the functions in Galerkin method. The least squares method was used previously only for solving boundary value problems.

Demanding that boundary conditions have to be satisfied by definition limits the application fields of the analytical approximation. It is difficult to find a set of functions which satisfy all boundary conditions, especially

for multidimensional tasks with non-continuous conditions.

The least squares method is based on integration. Therefore it is not sensitive to steep gradients. *Mathematica* as a tool which can easily handle analytical or numerical integration is very useful. Computations made with exact or arbitrary precision allows us to solve ill-conditioned problems.

The paper presents a refinement of analytical approximation with the method of least squares. It is based on the idea that, like in the Finite Element Method, boundary conditions can be added to the minimized functional. The functions which we use to approximate the tasks do not have to satisfy boundary conditions. Hence it is very simple to handle them. Moreover it is possible to solve the problems with initial conditions, as presented in the example. Evaluation of an error of approximation is much more straightforward.

2 The method description

Let us consider, to focus our attention, the domain D limited by the boundary $\Gamma = \Gamma_I \cup \Gamma_{II}$, Majchrzak & Mochnacki [4], in which the function is described by an equation,

$$\mathcal{L}(U) = b, \tag{1}$$

with conditions,

$$x \in \Gamma_I : \; U\,|_{\Gamma_I} = U_d, \quad x \in \Gamma_{II} : \; q\,|_{\Gamma_{II}} = q_d, \tag{2}$$

where x is a point on the boundary of the domain D, and $q = n \cdot \mathrm{grad}U$. Vector n is normal to the boundary. Let us introduce functions of residuum (error or defect of the solution),

$$R = \mathcal{L}(U) - b, \tag{3}$$

$$R_I = U - U_d, \tag{4}$$

$$R_{II} = q - q_d. \tag{5}$$

If the U_o is an approximate solution of the boundary value problem then residue functions are not equal to zero. We can build the following functional on the functions (3), (4) and (5),

$$\mathcal{F} = \int_{(D)} (R\,s)^2 \,\mathrm{d}D + \int_{(\Gamma_I)} (R_I\,s_I)^2 \,\mathrm{d}\Gamma_I + \int_{(\Gamma_{II})} (R_{II}\,s_{II})^2 \,\mathrm{d}\Gamma_{II}, \tag{6}$$

where s, s_I and s_{II} are scale coefficients (or scale functions).

The problem of these scale coefficients is very important. They are used to satisfy compatibility of physical measure units of all functional's summands. The more important role of them is that they can shape approximation paths. Enlargement of some of the scale coefficients results

in quicker convergence of that condition or equation which have these coefficients. The cost of that manipulation is slower convergence of other conditions or equations. The problem of scale coefficients has not been properly examined yet. It requires further research.

To approximate solutions by the minimization of the functional eqn (6), we can use the Ritz method. We can predict the solution in a form of linear combination of independent functions u_i. The best results are obtained with functions based on monic Tchebyshev T polynomials. Burden & Faires [3].

$$U_o = \beta_1 u_1 + \beta_2 u_2 + \ldots + \beta_m u_m, \tag{7}$$

where β_i are unknown coefficients.

If the function U_o only approximates the actual solution U then the functional eqn (6) is a positive number. The condition of minimization is equivalent to the system of m algebraic equations.

$$\frac{\mathrm{d}\mathcal{F}}{\mathrm{d}\beta_i} = 0. \tag{8}$$

For linear problems that system of equations is linear $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$. Square matrix $\mathbf{A}$ is symmetrical and positive definite.

Coefficients of the matrix $\mathbf{A}$ can be computed with the formula,

$$a_{ij} = \int_{(D)} c_i\, c_j\, s^2\, \mathrm{d}D + \int_{(\Gamma_I)} c_{Ii}\, c_{Ij}\, s_I^2\, \mathrm{d}\Gamma_I + \int_{(\Gamma_{II})} c_{IIi}\, c_{IIj}\, s_{II}^2\, \mathrm{d}\Gamma_{II}, \tag{9}$$

and coefficients of the vector $\mathbf{b}$ with the formula,

$$b_i = -\int_{(D)} c_i\, w\, s^2\, \mathrm{d}D - \int_{(\Gamma_I)} c_{Ii}\, w_I\, s_I^2\, \mathrm{d}\Gamma_I - \int_{(\Gamma_{II})} c_{IIi}\, w_{II}\, s_{II}^2\, \mathrm{d}\Gamma_{II}, \tag{10}$$

where c_k, c_{Ik} and c_{IIk} are coefficients which stand by the β_k and w, w_I and w_{II} are free terms in eqns (3), (4) and (5) respectively.

The solution of the system of eqns (8) is the vector $\mathbf{x}$ of coefficients β_k.

3 Example – the *Mathematica* implementation

The way of thinking, presented above, can be easily expanded on systems of equations or problems with initial-boundary conditions. Moreover we can solve any type of equations: differential, integral and so on. The functional eqn (6) can be built for any of the problems.

The *Mathematica* implementation is also very easy. To show this let us consider the following hyperbolic partial differential equation,

$$\frac{\partial^2 T}{\partial x^2} - \frac{\partial T}{\partial y} - \frac{\partial^2 T}{\partial y^2} = 0, \tag{11}$$

in domain $x \in (0, 10)$ and $y \in (0, 10)$, with two boundary conditions,

$$y \in (0,10),\ x = 0:\ T(x,y) = 0, \quad y \in (0,10),\ x = 10:\ \frac{\partial T}{\partial x} = 0, \tag{12}$$

and with two initial conditions,

$$x \in (0,10),\ y = 0:\ T(x,y) = \mathcal{H}(1-x), \quad \frac{\partial T}{\partial y} = 0, \tag{13}$$

where $\mathcal{H}(x)$ is a Heaviside unit step function.

In *Mathematica* language we can define for this task, the following functions of residuum,

```
rr:= (D[T[x,y],{x,2}] - D[T[x,y],y] - D[T[x,y],{y,2}])/sx1/sy1

r[1]:= ((T[x,y] - UnitStep[1-x])/sx1/.y->0)

r[2]:= ((D[T[x,y],y])/sx1/.y->0)

r[3]:= ((T[x,y])/sy1/.x->0)

r[4]:= ((D[T[x,y],x])/sy1/.x->bx)
```

where we introduce scale coefficients,

```
sx1 = Sqrt[20]; sy1 = Sqrt[20]
```

Coefficients of the linear system can be computed from the formulas,

```
a[i_,j_]:=a[i,j]=a[j,i]=
Expand[integ[Expand[cr[i] cr[j]]]+
integ1[Expand[cbr[i,1] cbr[j,1]+cbr[i,2] cbr[j,2]]]+
integ2[Expand[cbr[i,3] cbr[j,3]+cbr[i,4] cbr[j,4]]]]

b[i_]:=b[i]=
integ[Expand[cr[i] wr]]+
integ1[Expand[cbr[i,1] wbr[1]+cbr[i,2] wbr[2]]]+
integ2[Expand[cbr[i,3] wbr[3]+cbr[i,4] wbr[4]]]
```

Functions `integ`, `integ1`, and `integ2` are author defined functions computed by the *Mathematica* `Integrate`. They are used to increase the speed calculations. Symbols `cr`, `wr`, `wbr` and `cbr` are coefficients c_i and w_i in eqns (9) and (10), respectively. The computations of these coefficients has been made analytically. The task is ill-conditioned but the solution of linear system can be made by *Mathematica* with an arbitrary precision.

The result of computations of approximation of the task by the set of 361 functions is presented in the Figure 1. It is very easy to evaluate the approximation error. The obtained results can be substituted into the eqn (11). The diagram of its residuum is presented in Figure 2. It gives information only where the eqn (11) is not well satisfied. If we compare the absolute of the the maximal value of the residuum with the absolute of the maximal value of any of the summand in the equation, for example with functions $f(x,y) = \frac{\partial^2 T}{\partial x^2}$ and $g(x,y) = -\frac{\partial^2 T}{\partial y^2}$, (Figures 3 and 4), we receive the relative error to each of the of eqn (11) residuum components.

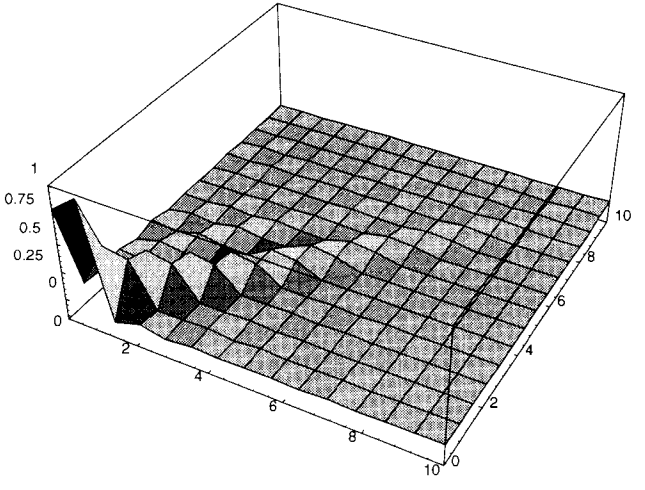

Figure 1: Approximate solution of the initial-boundary problem.

4 Final remarks and conclusions

The presented method can be easily applied to the initial-boundary problems and systems of equations. These equations could be differential, integral or integral-differential ones.

The main difference to the Galerkin, classical least squares method and other methods is that the satisfaction of boundary conditions by functions u_i is not essential. This is not essential and from the point of view of the author not advisable. It is not straightforward to find functions which satisfy non-continuous boundary conditions. Numerical experiments, carried out by the author, shows that the functions not "tied" to the boundary are more "flexible" to approximate the problem.

The method is also stable. As was shown above, there were no problems with solving hyperbolic and parabolic equations. The results were

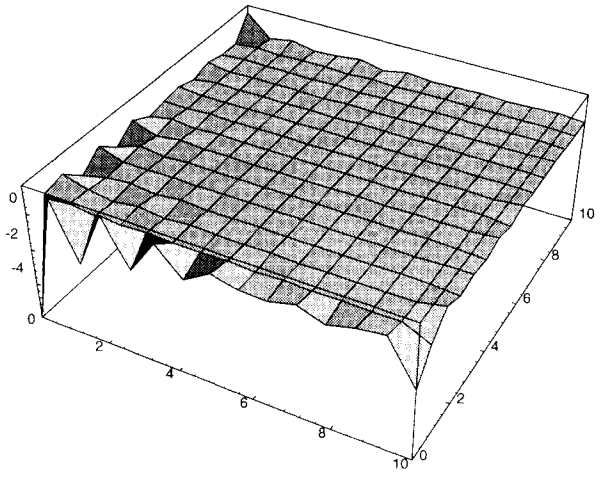

Figure 2: Error of the differential equation approximation.

much more accurate than those obtained with finite differences method. In the presented example finite difference method is unstable because of high gradients near the point $(0, 1)$.

The subject of research of the author is the theory of shells. These tasks are badly conditioned. One way to avoid instability is to search for the analytical solution. An exact solution is possible almost only for one-dimensional problems. Some solutions obtained with *Mathematica* were presented in [5]. The *Mathematica* as a tool which can solve equations with an arbitrary precision is very useful. The least squares method is also of much use here. The shell analysis by the method will be presented in future contributions.

Analytical integration with *Mathematica* was applied if possible. It is specially useful for ill-conditioned tasks like, for example, axially symmetrical problems. The precalculated integrals presented in the paper often make computations faster than with the numerical `NIntegrate` procedure.

The better approximation can be done by adding other functions to the eqn (7). The matrix $\mathbf{A}$ and vector $\mathbf{b}$ can be built up from ones obtained in the previous step. The Cholesky-Banachiewicz method is very useful here. It is more precise than Gauss Elimination for the type of problems considered. It has another positive feature. It does not need pivoting so the new elements could be added to the previously decomposed matrix $\mathbf{A}$. This increases the speed of the linear system's solution.

It is worth noting that *Mathematica* can handle complex numbers, so it is possible to solve the system of linear equations with non-positive definite matrix $\mathbf{A}$, by the Cholesky-Banachiewicz method.

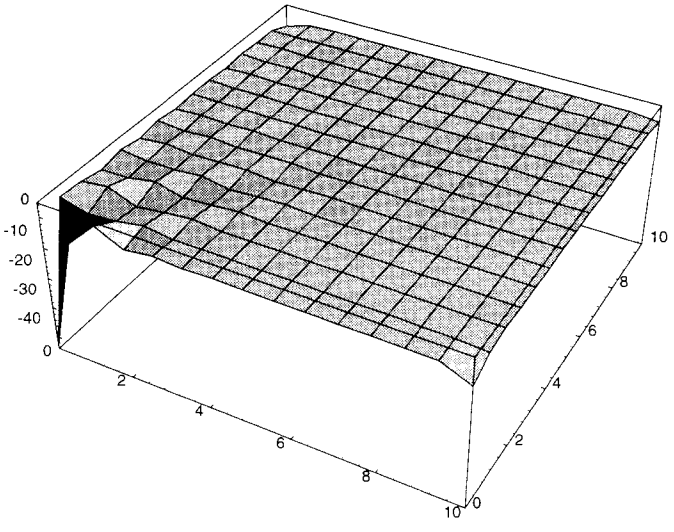

Figure 3: Function $f(x, y) = \frac{\partial^2 T}{\partial x^2}$.

The *Mathematica* lines, presented above, almost define the method. Only a few technical aspects were not presented here. The method utilizes one of the main advantages of the system – it saves the programmer time.

The result is given in the form of functions. It can be easily presented with *Mathematica* and used in other tasks. No interpolation and extrapolation is necessary.

The error estimation is very easy. One should only substitute the result into the residuum functions eqns (3), (4) or (5) to see "local satisfaction" of the problem or into functional eqn (6) to estimate the overall error.

Mathematica was applied in whole process of building the algorithms, testing them and solving the tasks. It is very useful for solving badly conditioned problems because it can compute with arbitrary precision and is "honest" returning the result omitting round-off error. It is a very important feature. The result can be presented graphically with wonderful *Mathematica* capabilities.

5 Acknowledgments

The contribution is supported by Wolfram Research which granted the registration fee. Author would like to express thanks to Mr Brett H. Barnhart from Wolfram Research for the organization of the support and Mr Duncan White for help in paper preparation.

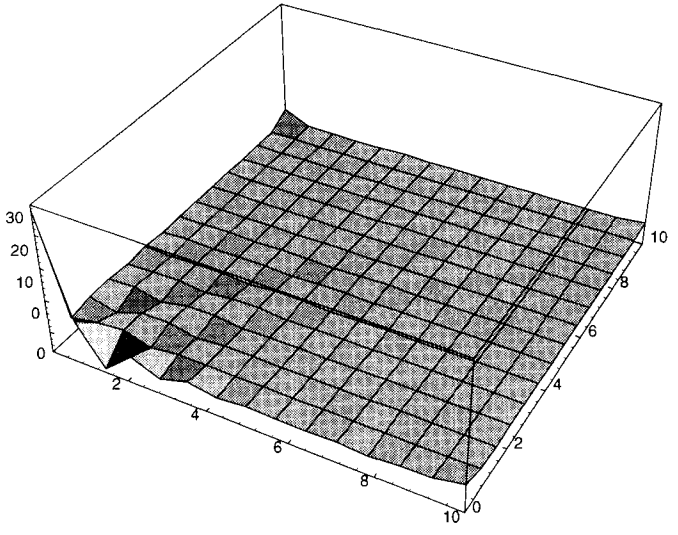

Figure 4: Function $g(x, y) = -\frac{\partial^2 T}{\partial y^2}$.

References

1. Barrère, R. Approximate symbolic calculations with a functional computing environment, in *Mathematics with Vision* (ed V. Keränen & P. Mitic), pp. 37 to 44, *Proceedings of the First International Mathematica Symposium*, Computational Mechanics Publications, Southampton, UK, Boston, USA, 1995.

2. Barrère, R. & Carmasol, A. Approximate solution of boundary-value problems using computer algebra software, in *Mathematics with Vision* (ed V. Keränen & P. Mitic), pp. 45 to 52, *Proceedings of the First International Mathematica Symposium*, Computational Mechanics Publications, Southampton, UK, Boston, USA, 1995.

3. Burden, R.L. & Faires, J.D. *Numerical Analysis*, PWS Publishing Company, Boston, 1993.

4. Majchrzak, E. & Mochnacki, B. *Numerical Methods*, Silesia University of Technology, Gliwice, 1994.

5. Walentyński, R. Application of *Mathematica* for solving analytical problems in theory of shells, in *Mathematics with Vision* (ed V. Keränen & P. Mitic), pp. 361 to 368, *Proceedings of the First International Mathematica Symposium*, Computational Mechanics Publications, Southampton, UK, Boston, USA, 1995.