Regular paper

APPLICATION OF THE MESHFREE RADIAL POINT INTERPOLATION METHOD (RPIM) TO SOLVE ELECTROSTATIC PROBLEMS

Nina ZAYAKOVA[1], Kostadin BRANDISKY[2]

Abstract: This paper compares the Finite Element Method (FEM), as the one of the most important numerical methods, and the meshfree method Radial Point Interpolation Method (RPIM), as newly developed numerical technique, for solving several electrostatic problems.

Keywords: Meshfree methods, Numerical integration, Computational Electromagnetics.

INTRODUCTION

The finite element method (FEM) is one of the most widely used methods of solving partial differential equations, because of its versatility and efficiency. Its implementation requires the spatial domain with complex geometry to be divided into elements, i.e. it relies on meshes or elements that are connected together by nodes. Some of the limitations using a mesh structure are: high cost in creating a FEM mesh, difficulties in adaptive analysis, limitation in the analyses of some problems. Thus, the concepts of meshfree or meshless method become increasingly attractive.

A meshfree method is a method used to establish system algebraic equations for the whole problem domain without the use of predefined mesh for the domain discretization. The major differences between FEM and meshfree method are in domain representation and shape function interpolation. In meshfree methods the problem domain and its boundaries are represented by scattered nodes. The sets of nodes are called field nodes and they do not form a mesh, which means that information on the relationship between the nodes is not required. Since there is no mesh, the field variable at any point within the domain is interpolated using function values at field nodes within a small local support domain of the point. The shape functions are valid only for a particular point of interest and usually change when the point of interest changes.

The predecessors of the meshfree methods like the collocation methods can be traced seventy years ago, but starting from the early 1990s the meshfree concept gained popularity in computational modeling. As a result, a number of meshless methods have been proposed, such as the element free Galerkin (EFG) method, the meshless local Petrov–Galerkin (MLPG), and the point interpolation method (PIM) [1].

All the above meshfree methods are based on weak-forms. According to the formulation procedures there are also two other categories of meshfree methods: those based on collocation techniques (strong-form methods) and those based on combination of weak-form and collocation techniques. The main differences among these two categories are the approximation technique used to obtain the shape functions and how the numerical integration is performed. In the weak-form methods a set of background cells is used to form the system equations. The strong-form methods are truly meshfree because there is no need of background cells for integration, but they are often unstable and less accurate. The third category uses background cells for integration to the least extent.

In this paper the Radial Point Interpolation Method (RPIM) was applied in modeling several two-dimensional electrostatic problems.

MESHFREE RADIAL POINT INTERPOLATION METHOD (RPIM)

General properties

The radial point interpolation method (RPIM) belongs to the first category of methods. Its formulation is based on the global Galerkin weak-form for problem equations [1]. It is characterized by a shape function that possesses the Kronecker delta property, a characteristic not present in all meshfree methods. It uses radial basis shape functions which shape parameters should be properly selected. Global background cells structure is required for integration in the Galerkin weak-forms. The essential boundary conditions are enforced as in FEM.

RPIM is a stable and robust method and has been successfully applied to electrostatic problems discussed in this paper.

RPIM formulation applied to electrostatic problems

The analysis of electrostatic field is performed in a 2D domain Ω (x-y plane), bounded by a boundary contour Γ. The electrostatic field is described by the Poisson equation for non-homogeneous medium in terms of the electric potential u:

$$\frac{\partial}{\partial x}\left(\varepsilon_x \frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(\varepsilon_y \frac{\partial u}{\partial y}\right)=-\rho \qquad (1)$$

with Dirichlet boundary conditions on Γ: $u = u_0$

where:

ρ – volume charge density, a function of $x, y;$

[1] Technical University of Sofia, Kliment Ohridski 8, Sofia 1000, Bulgaria, E-mail: mynik@abv.bg

[2] Technical University of Sofia, Kliment Ohridski 8, Sofia 1000, Bulgaria, E-mail: kbran@tu-sofia.bg

ε – permittivity, a function of *x, y;*

The weak formulation derived from Galerkin approach is

$$\iint_{(\Omega)} (\mathbf{L}\delta\mathbf{u})^T (\mathbf{Q}\mathbf{L}\mathbf{u}) d\Omega + \iint_{(\Omega)} \delta\mathbf{u}^T \rho d\Omega = 0 \tag{2}$$

where
$$\mathbf{L} = \begin{bmatrix} -\frac{\partial}{\partial x} & -\frac{\partial}{\partial y} \end{bmatrix}^T \tag{3}$$

and
$$\mathbf{Q} = \begin{bmatrix} \varepsilon_x & 0 \\ 0 & \varepsilon_y \end{bmatrix} \tag{4}$$

The domain is now represented by a set of arbitrary distributed field nodes sequentially numbered from 1 to N for the purpose of electric potential approximation. Then, the RPIM shape function is used to approximate the potential at any point of interest using a set of nodes in the local support domain of the point.

$$u^h = \mathbf{\Phi^T u} = \begin{bmatrix} \phi_1 \phi_2 ... \phi_n \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ ... \\ u_n \end{bmatrix} = \sum_I^n \phi_I u_I \tag{5}$$

where $\mathbf{\Phi}$ and $\mathbf{u}$ are the vectors of nodal shape functions and the nodal potentials of support domain nodes, respectively.

From Eq. (5) it is obtained

$$\delta u^h = \mathbf{\Phi}^T \delta\mathbf{u} = \sum_I^n \phi_I \delta u_I \tag{6}$$

The electric field intensity is computed as

$$\mathbf{E} = \mathbf{L}u^h = \mathbf{L}\mathbf{\Phi}^T\mathbf{u} \tag{7}$$

Substituting Eq. (6) in (7) can be obtained

$$\mathbf{E} = \begin{bmatrix} -\frac{\partial}{\partial x} \\ -\frac{\partial}{\partial y} \end{bmatrix} \begin{bmatrix} \phi_1 & \phi_2 & ... & \phi_n \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ ... \\ u_n \end{bmatrix} = \mathbf{B}_{(2xn)}\,\mathbf{u} \tag{8}$$

Substituting eqs. (8), (6) and (4) in the first term of Eq. (2)

$$\iint_{(\Omega)} (\mathbf{L}\delta\mathbf{u})^T (\mathbf{Q}\mathbf{L}\mathbf{u}) d\Omega = \iint_{(\Omega)} (\mathbf{B}\delta\mathbf{u})^T \mathbf{Q}\mathbf{B}\mathbf{u} d\Omega =$$

$$= \delta\mathbf{u}^T \underbrace{\left(\iint_{(\Omega)} \mathbf{B}^T\mathbf{Q}\mathbf{B}\; d\Omega \right)}_{\mathbf{K}_s} \mathbf{u} \tag{9}$$

Here the matrix $\mathbf{K}_S$ in the local support domain is

$$\mathbf{K}_S = \iint_{(\Omega)} \mathbf{B}^T\mathbf{Q}\mathbf{B}\; d\Omega \tag{10}$$

Until this stage, $\mathbf{u}$ is the vector of nodal potentials of nodes belonging to the support domain, and $\mathbf{K}_S$ is the submatrix with contributions from the nodes of the support domain to the global matrix. If we now define the global vector of the nodal potentials $\mathbf{U}$ and the global matrix $\mathbf{K}$, the Eq. (9) can be generalized as

$$\iint_{(\Omega)} (\mathbf{L}\delta\mathbf{u})^T (\mathbf{Q}\mathbf{L}\mathbf{u}) d\Omega = \delta\mathbf{U}^T\mathbf{K}\mathbf{U} \tag{11}$$

where $\mathbf{K}$ is the global matrix, assembled from $\mathbf{K}_S$:

$$\mathbf{K}_{(N\times N)} = \begin{bmatrix} K_{11}\; K_{12}\; ...\; K_{1N} \\ K_{21}\; K_{22}\; ...\; K_{2N} \\ ..... \\ K_{N1}\; K_{N2}\; ...\; K_{NN} \end{bmatrix} \tag{12}$$

In Eq. (11), the vector $\mathbf{U}$ is the global vector of potentials that collects the nodal potentials in the entire domain. Substituting Eq. (5) into the second term of Eq. (2), for whole domain it is obtained

$$\iint_{(\Omega)} \delta\mathbf{u}^T \rho d\Omega = \delta\mathbf{U}^T \iint_{(\Omega)} \mathbf{\Phi}\rho d\Omega = \delta\mathbf{U}^T\mathbf{F} \tag{13}$$

By substituting eqs. (11) and (13), Eq. (2) becomes

$$\delta\mathbf{U}^T\mathbf{K}\mathbf{U} - \delta\mathbf{U}^T\mathbf{F} = \delta\mathbf{U}^T[\mathbf{K}\mathbf{U} - \mathbf{F}] = 0 \tag{14}$$

Thus,

$$\mathbf{K}\mathbf{U} = \mathbf{F} \tag{15}$$

which is the final discretized system of equations.

Computation of the shape functions

The radial basis functions (RBF) are used to develop the RPIM shape functions. In contrast to polynomial PIM where singularity problem always persists, the RPIM shape functions are augmented with polynomials and thus singularity problem is avoided. The RPIM interpolation can be written as

$$u(\mathbf{x}) = \sum_{i=1}^n R_i(\mathbf{x}) a_i + \sum_{j=1}^m p_j(\mathbf{x}) b_j = \mathbf{R}^T(\mathbf{x})\mathbf{a} + \mathbf{p}^T(\mathbf{x})\mathbf{b} \tag{16}$$

where

$R_i(\mathbf{x})$ is RBF;

n is the number of RBFs;

$p_j(\mathbf{x})$ is monomial in space coordinates $\mathbf{x}^T = [x, y]$;

m is the number of polynomial basis functions, if $m=0$ pure RBFs are used.

In the RBF $R_i(x)$, the only variable is the distance between the point of interest x and the node at x_i.

$$r = \sqrt{(x - x_i)^2 + (y - y_i)^2} \tag{17}$$

Radial basis functions are a very powerful interpolation tool. Some of the most commonly used RBFs are the multiquadrics function, the Gaussian function, the thin

plate spline function and the logarithmic radial basis function.

In this paper, a multiquadrics RBF was chosen, which is given by

$$R_i(x,y)=\left[r_i^2+(\alpha_c d_c)^2\right]^q \tag{18}$$

In multiquadrics RBF there are two shape parameters: α_c and q, to be determined by the analyst for improving the performance of the method. d_c is characteristic length which is usually the average nodal spacing for all the nodes in the local support domain.

The polynomial term is not always necessary, but has been found that: RPIM shape functions with pure RBFs usually cannot pass the patch tests; the polynomials improve the accuracy of results, reduce the sensitivity of shape parameters and improve the interpolation stability for some RBFs.

A support domain including n field nodes is formed for the point of interest at $\mathbf{x}$. The coefficients a_i and b_j can be determined by enforcing Eq. (16) to be satisfied at each of these n nodes. This leads to n linear equations, one for each node. The matrix form of these equations is given by

$$\mathbf{U_S}=\mathbf{R_o a}+\mathbf{P_m b} \tag{19}$$

where

$$\mathbf{U}_S=\{u_1\ u_2\ ...\ u_n\}^T \tag{20}$$

- vector of function values at each node.

The moment matrix of RBFs is

$$\mathbf{R_0}=\begin{bmatrix} R_1(r_1)\ R_2(r_1)...\ R_n(r_1)\\ R_1(r_2)\ R_2(r_2)...\ R_n(r_2)\\ ...\quad ...\quad ....\ .... \\ R_1(r_n)\ R_2(r_1)...\ R_n(r_n)\end{bmatrix}_{n\times n} \tag{21}$$

where $R_i(r_k)$ is defined by Eq. (17)

the polynomial moment matrix is

$$\mathbf{P_m^T}=\begin{bmatrix} 1 & 1\ ... & 1\\ x_1 & x_2\ ... & x_n\\ y_1 & y_2\ ... & y_n\\ ... & ...\ ... & ...\\ p_m(x_1) & p_m(x_2)...\ p_m(x_n)\end{bmatrix}_{(m\times n)} \tag{22}$$

the vectors of coefficients for are:

$$\mathbf{a}^T=\{a_1\ a_2\ ...\ a_n\};\ \mathbf{b}^T=\{b_1\ b_2\ ...\ b_n\} \tag{23}$$

However, there are $n+m$ variables in Eq. (19). The additional m equations can be added using the following m constrained conditions

$$\sum_{i=1}^{n} p_j(\mathbf{x})a_i=\mathbf{P_m^T a}=0\quad j=1,2,3...,m \tag{24}$$

Combining Eq. (19) and (24) yields:

$$\tilde{\mathbf{U}}_\mathbf{s}=\begin{bmatrix}\mathbf{U_s}\\ \mathbf{0}\end{bmatrix}=\begin{bmatrix}\mathbf{R_0}\ \mathbf{P_m}\\ \mathbf{P_m^T}\ \mathbf{0}\end{bmatrix}\begin{Bmatrix}\mathbf{a}\\ \mathbf{b}\end{Bmatrix}=\mathbf{G a_0} \tag{25}$$

Because the matrix $\mathbf{R}_0$ is symmetric the matrix $\mathbf{G}$ will also be symmetric. Solving Eq. (25) we obtain

$$\mathbf{a_0}=\begin{Bmatrix}\mathbf{a}\\ \mathbf{b}\end{Bmatrix}=\mathbf{G^{-1}}\tilde{\mathbf{U}}_\mathbf{s} \tag{26}$$

Eq. (16) can be re-written as

$$u(\mathbf{x})=\mathbf{R}^T(\mathbf{x})\mathbf{a}+\mathbf{p}^T(\mathbf{x})\mathbf{b}=\left\{\mathbf{R}^T(\mathbf{x})\ \mathbf{p}^T(\mathbf{x})\right\}\begin{Bmatrix}\mathbf{a}\\ \mathbf{b}\end{Bmatrix}= \tag{27}$$

$$=\left\{\mathbf{R}^T(\mathbf{x})\ \mathbf{p}^T(\mathbf{x})\right\}G^{-1}\tilde{U}_s$$

Finally, after some transformations, we can obtain:

$$u(\mathbf{x})=\mathbf{\Phi}^T(\mathbf{x})\mathbf{U}_s=\sum_{i=1}^{n}\phi_i u_i \tag{28}$$

where

$\mathbf{\Phi}^T(\mathbf{x})=\{\phi_1(\mathbf{x})\ \phi_2(\mathbf{x})...\ \phi_n(\mathbf{x})\}^T$ - vector of the RPIM shape functions, corresponding to the nodal potentials vector.

RPIM shape function does satisfy the Kronecker delta function property (i.e., the shape function is unit at the node and zero at other nodes in the support domain), because it is created to pass through the nodal values. This simplifies the enforcement of Dirichlet boundary conditions just as it is done in the FEM (using the penalty method or other methods).

Program implementation and possibilities

A computer program MfreeC has been written in programming language C# to implement the meshfree RPIM method to electrostatic problems in 2D domains.

In RPIM, the implementation of the Galerkin procedure leads to the computation of integrals over the entire domain, requiring the definition of a set of quadrature points or a background mesh for the integral estimation. Thus, the application of RPIM requires the introduction of two sets of nodes: the *field nodes*, where the shape functions are defined, and the *quadrature points*, for computation of the integrals. The two grids of nodes can be independent.

In this work, the background cells used for numerical integration are rectangular and the integration points are located following the standard Gauss quadrature scheme.

It has been found in [1] that the total number of quadrature points has to be from 3 to 9 times more than the number of field nodes.

There are several improvements that can be introduced in the current program. First of all, the program uses the Gauss elimination method for solving the banded

system of algebraic equations. That means that process of solving is rather slow but the precision of the solution is better. An iterative method can be used to solve the system of equations which will diminish the solution time. The most popular iterative method for this kind of symmetric positive definite and banded matrix is the preconditioned conjugate gradient method.

Second, the current program solves the Poisson equation on rectangular domains with homogeneous material. Since most of the real electrostatic problems deal with non-homogeneous materials, the desired improvement is to solve problems in arbitrary shaped domains with non-homogeneous materials.

These improvements will be implemented in the second version of the program.

Now, for the purpose of this paper which is to present the meshfree RPIM method and its feasibility for solving electrostatic problems, the program implements the algorithm on Fig. 1:

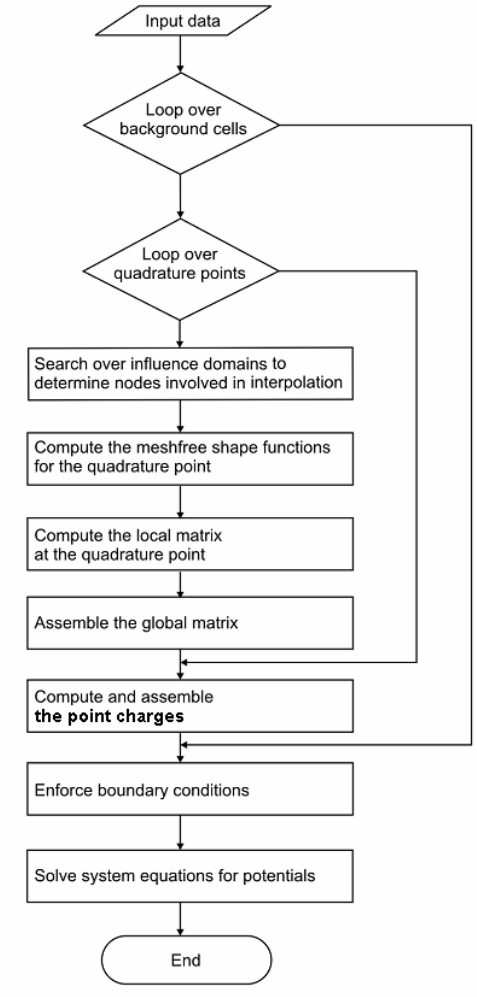

Fig. 1 - *Flowchart of MFreeC program*

RESULTS FROM TEST PROBLEMS

Problem 1: Square capacitor

A capacitor with square cross section having length of 1 meter is studied (Fig. 2). The boundary conditions are 100 V on the inner electrode and 0 V on the outer electrode.

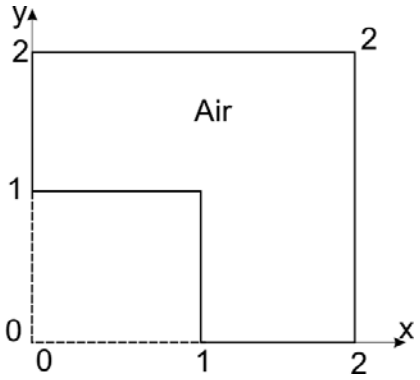

Fig. 2 - *Problem 1. Square capacitor*

Reference results for comparison have been computed using the finite element program FEMM 4.0 [2]. To obtain a very high accuracy, the problem has been solved with a very fine mesh of 197633 nodes and 393216 elements. In Table 1, the results for the potentials in three specific points are shown, using comparable number of field nodes in MFreeC and FEMM.

Table 1.

The results for computing time

	FEMM	MFreeC	FEMM
Number of nodes	**197633**	**341**	**417**
Number of elem./cells	**393216**	**75**	**768**
	Results	**Test results**	
Time (sec.)	**-**	**14**	**3**
(x, y)=(0.1, 1.1)	**89.6177**	**89.6313**	**89.6381**
(x, y)=(1.6, 1.7)	**9.43064**	**9.6012**	**9.5648**
(x, y)=(1.6, 0.1)	**38.8695**	**38.9532**	**38.9323**

The accuracy as a function of number of nodes is given bellow in Table 2.

Table 2.

The results as a function of number of nodes

Programs	**Nodes**	**0.1; 1.1**	**1.6; 1.7**	**1.6; 0.1**
Reference	**-**	**89.6177**	**9.43064**	**38.8695**
MFreeC	**340**	**89.6313**	**9.6012**	**38.9532**
Error (%)		**0.0152**	**1.8086**	**0.2153**
FEMM	**417**	**89.6381**	**9.5648**	**38.9323**
Error (%)		**0.0228**	**1.4226**	**0.1616**
	(x; y)	**0.2; 1.2**	**1.6; 1.8**	**1.6; 0.2**
Reference	**-**	**79.1618**	**6.27718**	**38.7084**
MFreeC	**96**	**79.257**	**6.5901**	**38.9895**
Error (%)		**0.1203**	**4.9850**	**0.7262**
FEMM	**113**	**79.2865**	**6.57065**	**38.93**
Error (%)		**0.1575**	**4.6752**	**0.5725**
	(x; y)	**0.25; 1.25**	**1.5; 1.5**	**0.75; .25**

Reference	**-**	**73.889**	**19.7541**	**23.9663**
MFreeC	**65**	**74.0749**	**21.0611**	**24.3591**
Error (%)		**0.2516**	**6.6163**	**1.6390**
FEMM	**65**	**73.9558**	**20.425**	**24.0518**
Error (%)		**0.0904**	**3.3963**	**0.3568**

Fig. 3 shows the absolute value of the error in MFreeC as well as in FEMM.

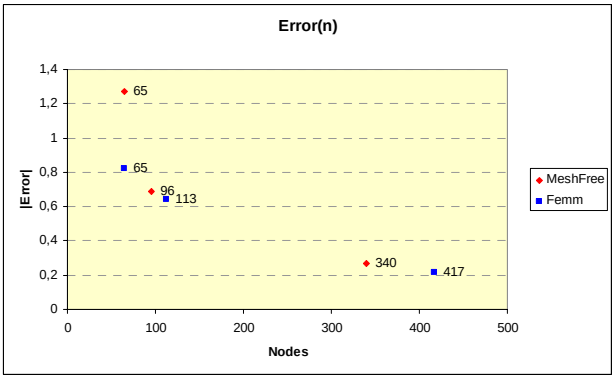

Fig. 3 - *Error= f(nodes) for Problem 1*

Problem 2: Poisson's PDE in square domain

The distribution of the potential in a square domain with side length of 1 meter (Fig. 4) is described by the Poisson's PDE:

$$\frac{\partial^2 V}{\partial x^2}+\frac{\partial^2 V}{\partial y^2}+1=0$$

with boundary conditions:

$V=0$ *on* $x=\pm 1$ *and* $y=\pm 1$

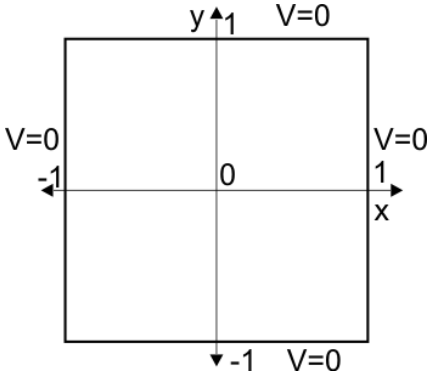

Fig. 4 - *Problem 2: Poisson's PDE for square region*

At x=0, y=0 the exact analytical decision is:

$V(0,0)=0.2947$

The results of MFreeC for 441 nodes and 100 background cells, computed for time of 24 sec., are:

$V(0,0)=0.2948$; Error: 0.034 %

The results in FEMM for 545 nodes and 1024 elements, computed for time of 2 sec., are:

$V(0,0)=0.2951$; Error: 0.136 %

The following comparison was made between the results of the two programs as the number of nodes decreases (Table 3):

Table 3.

The results as a function of number of nodes

	Nodes	**V(0,0)**	**Error (%)**
MFreeC	**441**	**0.2948**	**0.0339**
FEMM	**545**	**0.2951**	**0.1357**
MFreeC	**121**	**0.2954**	**0.2375**
FEMM	**145**	**0.2965**	**0.6108**
MFreeC	**81**	**0.2955**	**0.2715**
FEMM	**81**	**0.2938**	**0.3054**

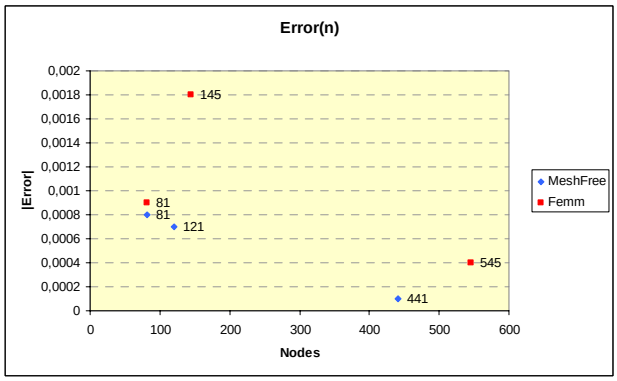

Fig. 5 - *Error= f(nodes) for Problem 2*

It can be seen that when the number of nodes is increased, the accuracy of MFreeC is better than the accuracy of FEMM.

Problem 3: Strip transmission line

The problem is shown in Fig. 6. It is described by Laplace equation with boundary conditions

V=0 V and V_0=100 V.

A quarter section of the domain is investigated because of its symmetry.

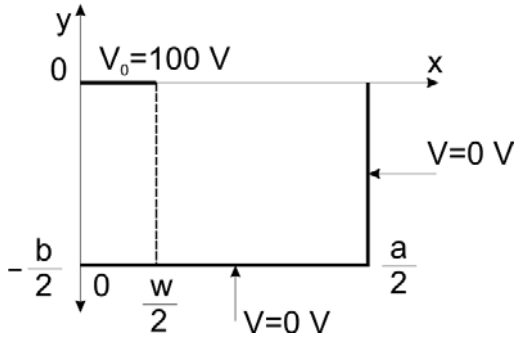

Fig. 6. *Strip transmission line*

First, the problem was solved by FEMM with high precision with 122104 nodes and 242764 elements and the result has been taken as reference.

Table 4.

The results for computing time

	FEMM	MFreeC	FEMM
Number of nodes	**122104**	**336**	**320**
Number of elem./cells	**242764**	**75**	**768**
	Reference	**Test results**	
Time (sec.)	**-**	**12**	**3**
(x, y) = (1, -0.7)	**22.5408**	**23.2996**	**23.2926**
(x, y) = (0.1, -0.1)	**89.6991**	**90.0286**	**89.9776**
(x, y) = (0.5,-1.0)	**21.0257**	**21.5596**	**21.4669**

The accuracy of the solution as function of number of nodes is shown below in Table 5 and Fig. 6:

Table 5.

The results as a function of number of nodes

Programs	**Nodes**	**(1; -0.7)**	**(0.1; -0.1)**	**(0.5;-1)**
Reference	**-**	**22.5408**	**89.6991**	**21.0257**
MFreeC	**336**	**23.2996**	**90.0286**	**21.5596**
Error (%)		**3.3663**	**0.3673**	**2.5393**
FEMM	**320**	**23.2926**	**89.9776**	**21.4669**
Error (%)		**3.3353**	**0.3105**	**2.0984**
	(x; y)	**(1; -0.7)**	**(0.2; -0,1)**	**(1.6;-1.4)**
Reference	**-**	**22.5408**	**89.074**	**1.07936**
MFreeC	**176**	**21.4014**	**88.8807**	**1.033**
Error (%)		**5.0548**	**0.2170**	**4.2951**
FEMM	**167**	**23.9185**	**89.9858**	**1.07284**
Error (%)		**6.1120**	**1.0236**	**0.6041**
	(x; y)	**(1; -0.75)**	**(0; -0.25)**	**(1;-1.25)**
Reference	**-**	**21.1445**	**75.4514**	**7.00659**
MFreeC	**77**	**20.6973**	**74.7619**	**6.8629**
Error (%)		**2.1150**	**0.9138**	**2.0508**
FEMM	**79**	**23.0783**	**77.192**	**7.63767**
Error (%)		**9.1456**	**2.3069**	**9.0069**

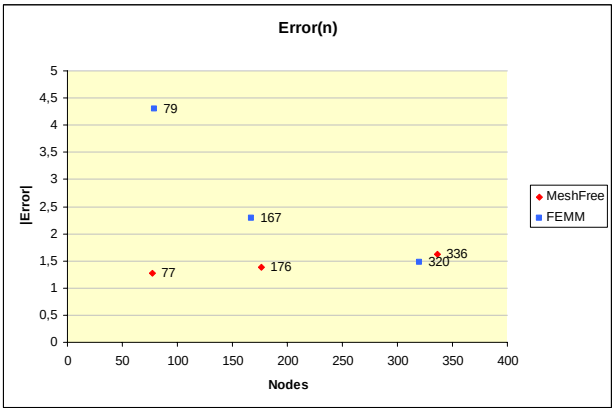

Fig. 6 - *Error= f(nodes) for Problem 3*

It can be seen that in this case increasing the number of nodes in MFreeC does not increase the accuracy, in contrast with FEMM.

CONCLUSIONS

RPIM is closer to FEM rather than the most meshfree methods, which this paper shows. The comparison was held in two directions. First, with respect to the amount of computing time, and second, with respect to the accuracy.

In the first direction, there is much more to be desired. The higher computing time in MFreeC is due to the search for field nodes that belong to the influence domain of a current quadrature point, in order to calculate its contribution for the global matrix. This search should be performed for each quadrature point. The numerical integration of the integrals is also much more time consuming than that of FEM.

In the second direction, it has been noticed that MFreeC accuracy in most cases is better than that of FEM. With approximately the same number of nodes the accuracy in the last two examples obtained by RPIM is much better. Of course we have to pay attention to the fact that RPIM has several parameters that has to be chosen by the analyst. With suitable choice of some of them better results could be obtained also in the first problem. The first parameter that can be improved is the number of background cells for integration and thus, the number of quadrature points. This will increase the accuracy of the numerical integration. The dimension of influence domain (α) is another parameter that could be changed. For the current investigation $\alpha = 3.0$ has been used, but higher α could improve the accuracy of RBF approximation.

In this paper it has been shown that the meshfree RPIM method in most of the cases has sufficiently good accuracy in solving electrostatic problems, better than that of FEM for comparable number of nodes.

REFERENCES

[1] G. R. Liu, Y. T. Gu: *"An introduction to meshfree methods and their programming"*, Springer, 2005.

[2] D. C. Meeker, Finite Element Method Magnetics, Version 4.0.1 (03Dec2006 Build), http://femm.foster-miller.net

Nina Zayakova was born on 17.12.1981 in Bourgas, Bulgaria. She is computer engineer since 2004 when she graduated from the Technical University of Sofia with excellent marks and received a Bachelor degree in Computer systems and technologies. In 2006 she is going to get a Master degree in Computer systems and technologies.

As Scholarship student of DAAD she spent two months at Technical University of Ilmenau in the summer of 2004. There she was working on the project "Multimedia Course on Computation of Electromagnetic Fields" in the framework of DAAD program "Academic Reconstruction in South-Eastern Europe". Some of her interests are: Scientific programming, Graphic Design, Drawing.