

**PROGRAM PACKAGE ERA-DD FOR THE SOLVING TWO-DIMENSIONAL
BOUNDARY VALUE PROBLEMS ON QUASISTRUCTURED GRIDS**

Dmitriy Beljaev, Alexander Kozyrev, Viktor Sveshnikov

The Institute of Computational Mathematics and Mathematical Geophysics SB RAS,
pr. Akademika Lavrentieva, 6, Novosibirsk, 630090, Russia.

E-mail: victor@lapasrv.sccc.ru

Abstract

The program package for the solving boundary value problems on quasistructured not co-ordinated grids by the iterative domain decomposition method on the subdomains interfaced without overlapping is developed. The description of computational algorithm, the data structures for work with quasistructured grids is resulted. The graphic interface for automation of initial data input and examples of numerical calculations are given.

Key words: program package, quasistructured grids, data structures, graphic interface, domain decomposition method.

1. INTRODUCTION

A software package for solving Poisson's equation in a plane or an axially symmetric domain that has almost arbitrary shape with the Dirichlet and the Neumann boundary conditions on the quasistructured mismatched grids has been designed. The need for such a development occurred in the process calculating of electric fields in the High Current Electronics problems, characterized by a strong heterogeneity of different scales and flows of charged particles. The solution is sought by the iterative method of decomposition of a computational domain into subdomains, without overlapping [1]. The boundary of a computational domain is approximated by straight lines or arcs of circles. A special graphical user interface was developed [2], enabling graphical and numerical data entry into the geometry of a computational domain, boundary conditions and quasistructured grids. The original rectangular grid near the boundary undergoes a local modification, consisting in a shift of the boundary nodes to the points of intersection of coordinate lines with the boundary. In the subdomains with a strong heterogeneity of solutions there is possibility of a local increase in the number of nodes. This ensures the adaptability of a grid. Original differential equation in subdomains can be approximated by the finite volume method [3], and the Poincare-Steklov equation at the interface can be approximated by a system of linear algebraic equations [1], which is solved using the conjugate gradient method. The paper describes the data structures for quasistructured grids, the package structure and gives an example of numerical calculations.

2. FORMULATION OF THE PROBLEM AND SOLUTION ALGORITHM

In the closed 2D domain $\bar{G} = G \cup \Gamma$, where Γ is the boundary of G , it is required to find a solution of the boundary value problem

$$\Delta u = g_1, \quad u|_{\Gamma_1} = g_2, \quad \frac{\partial u}{\partial n}|_{\Gamma_2} = g_3. \quad (1)$$

Here $u = u(T)$ is an unknown function, $g_1 = g_1(T)$, $g_2 = g_2(T)$, $g_3 = g_3(T)$ are a set of functions ($T = (x, y)$ is a current point, where x, y are the Cartesian or cylindrical coordinates), Δ is Laplace operator, $\vec{n}$ is normal to Γ . We assume that Γ_2 consists of segments parallel to the coordinate axes.

We decompose of the computational domain G . Construct the rectangle $\bar{R} = \{0 \leq x \leq D_x, 0 \leq y \leq D_y\}$, where D_x, D_y are a set of values around G . Build in $\bar{R}$ a rectangular uniform macrogrid

$$\bar{\Omega}_H = \left\{ X_I = IH_x, Y_J = JH_y, \quad I = \overline{0, N_x}, J = \overline{0, N_y}, \quad H_x = \frac{D_x}{N_x}, H_y = \frac{D_y}{N_y} \right\},$$

where N_x, N_y are a set of integer numbers, with the steps $H_x, H_y \gg h$ (h is a maximum grid step on which problem (1)) is approximated. The domain G is divided into subdomains G_k . External subdomains are excluded from the calculations. The interface consists of segments of the coordinate lines Ω_H . A point of intersection of coordinate lines is called macronodes.

In the subdomains, construct a uniform rectangular subgrid:

$$\bar{\Omega}_{h,k} = \{x_{i_k} = X_I + i_k h_{x,k}, y_{j_k} = Y_J + j_k h_{y,k}, \quad i_k = \overline{0, n_{x,k}}, j_k = \overline{0, n_{y,k}}\},$$

with the steps $h_{x,k} = \frac{X_{I+1} - X_I}{n_{x,k}}, \quad h_{y,k} = \frac{Y_{J+1} - Y_J}{n_{y,k}}$. Without essential loss of generality, we

assume that $n_{x,k} = 2^{m_{x,k}}, n_{y,k} = 2^{m_{y,k}}$, where $m_{x,k}, m_{y,k} > 0$ are integers. In order to adapt to the boundary subdomains, let us make a local modification of the subgrid consisting in a shift of the nearest boundary nodes to the points of intersection of coordinate grid lines with the boundary. Combining the subgrids forms resulting quasistructured grid $\bar{\Omega}_h$. Boundary value problem (1) is replaced by the problem

$$\Delta_h u_h = g_1, \quad u_h|_{\Gamma_1} = g_2, \quad l_h u_h|_{\Gamma_2} = g_3, \quad u_h|_{\gamma} = v_h, \quad (2)$$

by the finite volume method, where u_h is an approximate value of the unknown function u , and Δ_h, l_h are the approximation of the Laplace operator and the operator of normal derivatives, v_h is a trace u_h on γ . The solution to problem (2) is reduced to solving discrete subproblems in the subdomains. These subproblems in each subdomain lead, in fact, to the solution of nine-point grid equations.

We introduce the grid ω_h at the interface, consisting of nodes of subgrids $\bar{\Omega}_h^{(k)}$ and not containing macronodes. Let N be the number of nodes ω_h . In [1], it is shown that the function v_h can be found by solving a system of linear algebraic equations

$$Av_h + b = 0,$$

$$\Delta_h^{(p)} v_h = g_h.$$

Here $A = \{a_{i,j}, i = \overline{1, N}, j = \overline{1, N+P}\}$ is a rectangular matrix, $b = \{b_i, i = \overline{1, N}\}$, $g_h = \{g_{h,i}, i = \overline{N+1, N+P}\}$, $v_h = \{v_{h,i}, i = \overline{1, N+P}\}$ are vectors (P is the number of macronodes), $\Delta_h^{(p)}$ is an approximation of the Laplacian on the grid pattern, which includes the grid nodes ω_h and one of macronodes.

The solution of this system is an iterative decomposition method of the form

$$v_h^{(n+1)} = \Lambda(v_h^{(n)}, \bar{A}v_h^{(n)}), \quad n = 0, 1, \dots \quad (3)$$

where Λ is a function that determines a specific algorithm. The meaning of (3) lies in the fact that in this iterative process we use only the vector $v_h^{(n)}$ as it is and the action of a matrix-vector $\bar{A}v_h^{(n)}$. These requirements are satisfied, for example, in a family of rapidly convergent iterative methods in the Krylov subspace [3].

Knowing v_p , we find the functions u_h in the subdomains by solving problem (2).

3. DATA STRUCTURE FOR A QUASISTRUCTURED GRIDS

For every k , an internal or a boundary subdomain contains all the components of the structured subgrid $\bar{\Omega}_{h,k}$, including the external nodes. Among the nodes belonging to the computational domain $\bar{G}$, there are different internal nodes, whose coordinates are not stored, as they are calculated by simple rules, that follow from the structure $\bar{\Omega}_{h,k}$ and the boundary or modified nodes, whose coordinates are stored in the array.

Arrays for the grid ω_h at the interface is not specifically introduced as this grid essentially consists of nodes, which are projections on the interface of the subgrid $\bar{\Omega}_{h,k}$.

Let us describe stored scalars and arrays, following the rules of writing Fortran arrays and introduce a local numbering.

1. External data generated by graphical preprocessor.

1.1. Scalars

1.1.1. nx, ny are the number of intervals of the macrogrid Ω_H , that is actually the number of subdomains in their respective areas.

1.1.2. xb, yb, xe, ye are coordinates of the lower left and the upper right corners of the rectangle circumscribing the computational domain G .

1.2. Array

1.2.1. NNSG (nx, ny, 2) is the number of intervals in the subgrid $\bar{\Omega}_{h,k}$.

2. Internal data, formed in the quasistructured grids processing.

2.1. Scalars

2.1.1. nngi is the total number of nodes in the subdomains that are involved in the calculations. It includes all the nodes of subdomains. Each subdomain is represented as a single rectangle, for which all the nodes are stored, including the sides of the rectangle and

macronodes. At the boundaries of the conjugation of two subdomains, nodes are stored for each of them, that is, coincident nodes are repeated.

2.1.2. *nlmn* is the number of modified nodes.

2.2. Arrays for subgrids in the computational domain

2.2.1. *ITSG* (*nx*, *ny*) are type of subdomains (-1 - external, 0 - internal, 1 - boundary).

2.2.2. *IANG* (*nx*, *ny*) are initial addresses of the subarrays *ITNG*, *RUNG* of nodes in the subdomains.

2.2.3. *ITNG* (*nngi*) are type of nodes in the computational domain (-1 - external, 0 - internal, $a > 0$ - modified, where *a* means the address of the coordinates in the array *COLM* (see below)).

2.2.4. *RUNG* (*nngi*) are values of the unknown functions in the nodes in the computational domain (the same components as in the array *ITNG*).

2.2.5. *COLM* ($3 * nlmn$) are coordinates of the modified nodes. For each node, *x*, *y*, *ig* are stored. The latter is number of a boundaries, to be considered in a node.

4. THE STRUCTURE OF THE SOFTWARE PACKAGE

The package consists of macromodules *Graf*, *Geom*, *CaCoe*, *SolverU*, *Service*, which operate in the above-listed sequence, recording the results in to files. Macromodule *Graf* is a preprocessing package, a macromodules *Geom*, *CaCoe*, *SolverU* form a processor, and macromodule *Service* is postprocessor. Let us give a brief description of the macromodules.

Macromodule *Graf* is used to specify input information via graphical interface. The output information is written in the following files:

- *era.dat*, *point.dat* — the geometry of the boundary conditions,
- *fqgrid.dat* — quasistructured grids.

Macromodule *Geom* is used for processing the geometry of the computational domain. Its functions include the following procedures: construction of grids, the local modification of the grid, the layout of nodes in subdomains. Initial data are information about the geometry of the computational domain, boundary conditions and quasistructured grids, stored in files *era.dat*, *point.dat*, *fqgrid.dat*. Output data:

- scalars, that are stored in the file *fscalar.dat*,
- arrays for nodes in subdomains: *ITSG*, *IANG*, *ITNG*,
- an array of coordinates of local-modified nodes *COLM*.

All arrays are stored in a file *fsgeom.dat*

Macromodule *CaCoe* calculates the coefficients of difference equations in subdomains. Coefficients are stored for each near node to the boundary, as well as for each node, which lies on the boundary with the Neumann condition. It should be emphasized that for the boundary nodes with the Neumann condition are constructed grid equations and they are the values of the desired function, and for the boundary nodes with the Dirichlet condition this macromodule is responsible for sending the values of this function which are defined at the external boundary to the array *RUNG*. Initial data are contained in the files *era.dat*, *point.dat*, *fqgrid.dat*, *fscalar.dat*, *fsgeom.dat*. Output data:

- scalars in the file *fscalar.dat*, complemented with new values,
- array RUNG.

Macromodule SolverU realizes an iterative process of subdomains, and macromodule Service is a quasistructured grids for the image and display isolines solutions.

5. THE GRAPHIC INTERFACE FOR INITIAL DATA AUTOMATION

The main objective of the graphic interface for the solving the boundary value problems on quasistructured grids is a convenient way of setting and editing the geometrical and functional information about geometry of the computational domain, boundary conditions and quasistructured grids on which calculations are carried out.

The interface structure includes the following components: a graphic window of visualization, a menu, a toolbar with buttons of management, the dialogue boxes.

The graphic window of visualization serves for displaying the geometry of the computational domain.

The menu serves for managing the data files and, also, for choosing the drawing elements in a visualization window and contains the points: *File* serves for creating, saving and opening files; *Editing* serves for editing the provided information; *Data* serves for setting a system of coordinates, a quasistructured grid and the boundary conditions; *Processing* serves for starting the process of solving the defined boundary value problem; *View* serves to specify the scale, which determines the number of pixels per unit length, and the background grid, which is rectangular and serves as a reference when drawing an area; *Wizard* serves setting all data about the problem in a step-by-step mode and, finally, *Help*.

Toolbar contains 16 buttons of management divided into four groups. In the first group, there are 3 buttons for operating files. In the second group, there are 3 buttons for model increase/reduction in a graphic window. In the third group, there are 8 buttons for the input and editing of a domain. In the fourth group, there are 2 buttons for viewing numerical parameters of the problem. All the buttons have emerging tooltips.

Dialog boxes are intended for input/output of information associated with the problem. The boxes consist of titles, buttons of managing of a window (curtailing, restoring, closing a window), fields for input/output, buttons of confirming of the operations executed in a window.

The user has a possibility to work with the graphic interface in the two ways: 1) to set a necessary information, addressing to points of the menu and to buttons of the toolbar, in a sequence defined at one's choice, 2) by means of the *Wizard*, which consistently opens a necessary dialogue box.

The basic operation which is most often applied in the interface is the input of a geometrical object, that is, a straight line or a circle arc. A closed sequence of geometrical objects forms a contour of computational domain boundary.

For the input of a geometrical object three ways are possible: 1) graphic, 2) numerical, 3) graphic-numerical.

First, let us consider the input of a straight line. A simple way of inputting is graphic. It consists in the following. The corresponding button on the toolbar becomes active, which is indicated by the mouse showing the initial, and the final points which are then automatically connected. Coordinates of the mouse cursor are displayed in the left bottom corner of the screen with mouse moving.

It is possible to allocate any point or a group of points. The allocated point is represented by a square, while not allocated point — by a circle. Building the next piece of a straight line is carried out by

indication to one new point which automatically connected with previous allocated point, and then it becomes allocated.

The numerical way of building a straight line consists in setting the coordinates of two points in a special dialog box, which then are connected.

The concept of basic points is introduced. The basic points are such points which register in a special table, and then can repeatedly be used in the geometry description.

Building an arc of a circle is similarly carried out. Essential difference from a straight line is that in a dialog box which appears to activation of the corresponding button, there is a possibility of setting a radius of a circle.

Certainly, when building a computational domain boundary contour there is a possibility of connecting of pieces of straight lines and arcs of circles by above described the ways.

Any geometrical object can be allocated. It is provided by two ways. The first way: with a mouse construct a subdomain as a rectangle or a circle. Then all the objects which are completely inside a given subdomain are considered to be the allocated objects. The second way consists in consecutive allocation of objects by means of the mouse (similar to the point allocation). The allocated object is represented by red color unlike a non-allocated object, which is represented by black color.

Boundary conditions for the problem in question are set by indication of their types in a special dialog box, allocation of geometrical objects and indication to a concrete condition on the allocated objects.

The input of quasistructured grids is carried out in a special dialog box and includes two stages: 1) building a rectangular macrogrid $\overline{\Omega}_H$ defining domain decomposition, 2) building subgrids $\overline{\Omega}_{h,k}$ in the subdomains formed at the first stage.

The input information can be subject to editing which is carried out either graphically or by changing of numerical data about objects. The first graphic way of editing consists in allocation of one object or several objects, their moving and / or changing of their position and size. For realization of the second way of editing, a special dialog box, which contains a table of numerical data about objects, the kind of a piece of a line, coordinates of points of the beginnings-ends, a circle radius (if necessary), number of a boundary condition. Editing in this case consists in changing the numerical information in the table. Any object can be removed.

The input information is exposed to the analysis of completeness and compatibility. A warning message is displayed in case of error detection.

An important feature of the present graphic interface allowing inexperienced user to considerably simplify the work with it, is the possibility of setting the information by means of special *Wizard*. Making active this function, the user does not care of opening dialog boxes. This is done by *Wizard* which consistently opens the dialog box intended for the input of the information and is supplied with tooltips.

The input information can be saved on a disk in a file with an original name, and then be read out from the disk.

When finished the input and editing graphic-numerical data are converted into numerical information, which then is used computational modules of the package.

6. EXAMPLES OF NUMERICAL CALCULATIONS

Consider the following test problem:

$$\Delta u = 0, \quad u|_{\Gamma_1} = g(T), \quad \frac{\partial u}{\partial \vec{n}}|_{\Gamma_2} = 0.$$

The electric field is calculated in a quarter of a cylindrical condenser. Here T is a current point, $\Gamma_1 = \Gamma_{1,1} \cup \Gamma_{1,2}$, where $\Gamma_{1,1}$, $\Gamma_{1,2}$ are a quarter of concentric circles with radius $R_1 = 0.1$ and $R_2 = 1$,

Γ_2 are straight line segments parallel to the coordinate axes, $g = \begin{cases} 0, & T \in \Gamma_{1,1}, \\ 1, & T \in \Gamma_{1,2} \end{cases}$

Fig. 1 shows quasistructured grids, on which the calculation is carried out, and a picture of equipotential lines (a darker color filling corresponds to a greater potential).

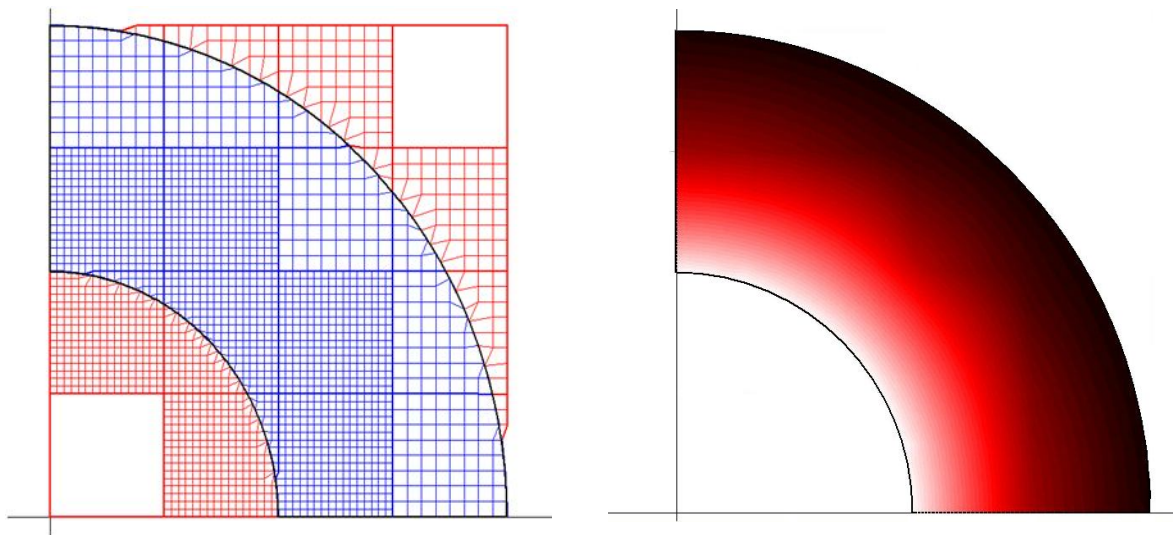
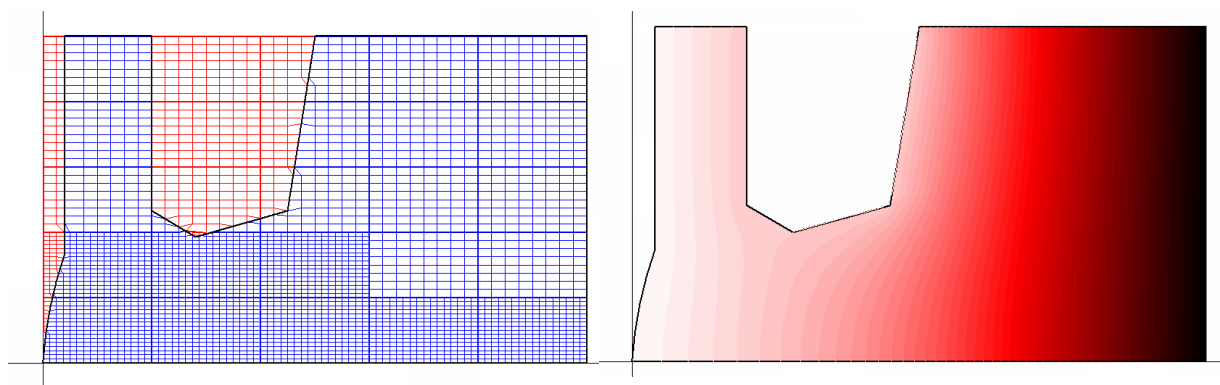


Fig. 1. Quasistructured grids and an electric field in a cylindrical condenser

Fig.2 similar to Fig 1 shows quasistructured grids and the result of calculating the electric field in an electron gun

Fig. 2. Quasistructured grids and a picture of an electric field in an electron gun



ACKNOWLEDGEMENTS:

This work is supported by the Russian Fundamental Research Foundation under Awards 10-01-90009.

REFERENCES

1. Sveshnikov V. M. (2010) A Construction of Direct and Iterative Decomposition Methods, Journal of Applied and Industrial Mathematics 4(3), pp. 1–12.
2. Kozyrev A.N., Sveshnikov V.M. (2009) Autmation of initial data input in program package ERA-DD for the solving of two-dimensional boundary value problems on quasistructured grids. Bull. of The Novosibirsk Computing Center Ser. Numerical Analysis, Novosibirsk, ICMMG, 14,. pp. 11-23.
3. Il'in V.P. (2001) Methods of finite differences and finite volumes for elliptic equations, Novosibirsk, ICMMG.