

## Automation of initial data input in the program package ERA-DD for solving two-dimensional boundary value problems on quasi-structured grids\*

A.N. Kozyrev, V.M. Sveshnikov

**Abstract.** A graphical user interface of the program package ERA-DD for the input and editing of initial data for a two-dimensional boundary value problem is described. The boundary of calculation domain is approximated by pieces of straight lines and circle arches. The input of the information about the boundary is provided graphically and numerically. A special quasi-structured grid is constructed in order to solve the boundary value problem using a domain decomposition method without overlapping of subdomains. The algorithm of constructing the quasi-structured grid is proposed. The input of the information about the problem can be carried out by means of the Wizard dialog.

### 1. Introduction

The necessity of creation of the graphical user interface (GUI) for solving practical problems with a complex configuration of calculation domain boundary is caused by aspiration to facilitate to the user-engineer process of the input and editing of initial data. The GUI described in this paper is a sequel of [1].

The boundary of calculation domain is represented in the form of geometric objects: pieces of straight lines and circle arches. The following ways are provided for its input: 1) graphical with a computer mouse, 2) numerical, and 3) graphical-numerical. The latter two are caused by the fact that frequently, an initial data for the problem description is the design drawing, in which the sizes of details are precisely specified.

The boundary value problem is solved on a quasi-structured grid which is constructed in two stages. At the first stage, in a rectangle surrounding the calculation domain, a uniform rectangular grid with steps essentially exceeding a maximum step of a resulting grid on which the boundary value problem is approximated and solved is constructed. Then in each macrocell (subdomain) a rectangular uniform subgrid is constructed. Nodes of the latter subgrid near the boundary are subject to the local updating [2] that is a shift of nodes to points, crossing coordinate lines of this subgrid with the boundary. Such a quasi-structured grid has two advantages: first, it is adaptive as it takes into account the boundary configuration and physical

---

\*Supported by the Russian Foundation for Basic Research under Grant 08-01-00526.

heterogeneity in the domain and, second, it has a simple structure. The direct input of data to such a grid with a big number of subdomains is problematic. In this paper, effective technique implemented in the GUI and allowing automation of the given procedure is proposed.

The boundary value problem is solved by the domain decomposition method [3]. For its realization, the processor of graphic interface delivers necessary numerical data.

## 2. Formulation of a boundary value problem.

### Quasi-structured grids. Algorithm of solving a boundary value problem

Consider in a closed 2D domain  $\overline{G} = G \cup \Gamma$  with the boundary  $\Gamma = \Gamma_1 \cup \Gamma_2$  a boundary value problem

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

Here  $u = u(x, y)$  is a required function,  $g_1 = g_1(x, y)$ ,  $g_2 = g_2(x, y)$ ,  $g_3 = g_3(x, y)$  are given functions, where  $x, y$  are the Cartesian coordinates,  $\Delta$  is the Laplace operator, and  $\vec{n}$  is a normal to  $\Gamma$ .

Let us specify a rectangle  $R = \{0 \leq x \leq D_x, 0 \leq y \leq D_y\}$  around the calculation domain, where  $D_x, D_y$  are given values ( $R \supset \overline{G}$ ).

We construct in  $R$  a rectangular uniform macrogrid

$$\Omega_H = \{X_I = IH_x, Y_J = JH_y, I = \overline{0, N_x}, J = \overline{0, N_y}\}, \quad (2)$$

$$H_x = \frac{D_x}{N_x}, \quad H_y = \frac{D_y}{N_y},$$

where  $N_x, N_y$  are given integers producing steps  $H_x, H_y$  that are essentially greater than  $h$  – a maximum step of the grid on which problem (1) is approximated.

Thereby, we decompose the calculation domain  $G$  to nonempty subdomains  $G_k = G_{I,J} = G \cap R_{I,J} \neq \emptyset$ ,  $k = \overline{1, K}$ ,  $(I, J) \in M$ , where  $R_{I,J}$  is an open rectangular subdomain representing a macrocell of the grid  $\Omega_H$ ,  $K$  is a known integer defined by the geometry of  $\overline{G}$ , and  $M$  is a set of pairs of indices  $(I, J)$  for which the intersection of  $G$  and  $R_{I,J}$  is nonempty.

In a macrocell  $R_k = R_{I,J}$ ,  $(I, J) \in M$ , we construct the uniform rectangular subgrid

$$\overline{\Omega}_{h,k} = \{x_{i,k} = X_I + ih_{x,k}, y_{j,k} = Y_J + jh_{y,k}, i = \overline{0, n_{x,k}}, j = \overline{0, n_{y,k}}\} \quad (3)$$

with the steps  $h_{x,k} = \frac{X_{I+1} - X_I}{n_{x,k}}$ ,  $h_{y,k} = \frac{Y_{J+1} - Y_J}{n_{y,k}}$ . The subgrid  $\overline{\Omega}_{h,k}$  will be adapted to the boundary with the local modification [2] using shifts of nodes

closed to the boundary to points of intersection of the coordinate lines with the boundary.

Let us note that under the above assumptions,  $\bar{\Omega}_{h,k}$  are quasi-matched in the sense that on the pieces  $\gamma_k$  of the boundary all the nodes of sparse grids coincide with those of dense grids.

The solution to initial problem (1) is found on the resulting grid:

$$\bar{\Omega}_h = \bigcup_{k=1}^K \bar{\Omega}_{h,k}. \quad (4)$$

In addition to the above-considered grids, we introduce the grid  $\omega_h = \bar{\Omega}_h \cap \gamma$  and the subgrids  $\omega_{h,k} = \bar{\Omega}_{h,k} \cap \gamma$ ,  $k = \bar{1}, \bar{K}$ .

The algorithm of solving problem (1) consists in constructing the function  $u$  in such a way that the function  $u|_\gamma$  is calculated on the boundary  $\omega_h$ , and then, using the obtained values as Dirichlet boundary conditions, the boundary value problems in subdomains are solved, that is, the function  $u$  in the whole domain is defined. The basic problem is finding  $u|_\gamma$ . For its solution, a special system of linear algebraic equations [3] providing approximate values  $u_h$  of the required function at the nodes of the grid  $\omega_h$  is constructed. The boundary value problems in subdomains are approximately solved on the subgrids  $\bar{\Omega}_{h,k}$  (3) by finite volume methods.

### 3. Graphical user interface

The main objective of the GUI for solving the boundary value problems on quasi-structured grids is a convenient setting and editing the geometrical and functional information about the geometry of a calculation domain, boundary conditions, and quasi-structured grids on which calculations are carried out.

The GUI is implemented in the form of a Windows application developed in the programming system Borland C++ Builder 6. A screenshot of the application GUI is shown in Figure 1.

The GUI consists of the following components:

- a visualization window,
- a menu,
- a toolbar,
- dialogue boxes.

**The visualization window** displays a geometry of a calculation domain.

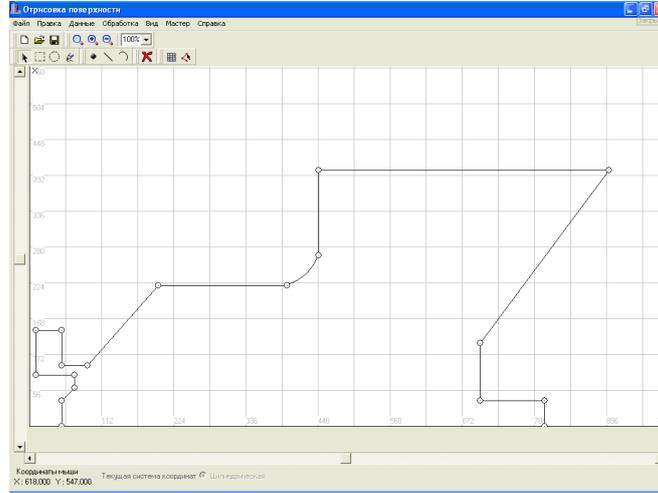


Figure 1. Graphical user interface

**The menu** manages a work with data files and provides a selection of drawing elements in the visualization window. The menu contains the following items:

**File** contains standard commands for work with data files: **New**, **Open**, **Save**, and **Exit**.

**Data** manages data for a boundary value problem and for a quasi-structured grid. It contains the following commands:

**Coordinate system** command chooses a coordinate system (cylindrical or Cartesian);

**Origin binding** command binds the origin of the coordinate system with a point of the graphical page (by default it is the bottom left corner);

**Quasi-structured grid** command manages a work with the quasi-structured grid;

**Boundary conditions** command manages a work with boundary conditions.

**Processing** starts a process of solving the boundary value problem defined.

**View** has two commands—**Scale** and **Background grid**. Scale  $P_\xi$  is specified for each coordinate axis ( $\xi = x, y$ ) as  $P_\xi = L_\xi^{(m)}/L_\xi^{(s)}$ , where  $L_\xi^{(m)}$  and  $L_\xi^{(s)}$  are respectively the sizes of a given domain in the user units and in pixels. The background grid is rectangular and it serves as a reference grid for domain drawing. Clicking on a command, a corre-

sponding dialog box is shown in which a scale or a number of lines of the grid in one direction ( $x$  or  $y$ ) can be set.

**Wizard** provides setting of all the data about the problem in a step-by-step order. It consists in subsequent show of five windows: 1) specification of key parameters of the problem such as coordinates system, scale, origin binding, 2) initialization of a background grid and the domain drawing window, 3) specification of boundary conditions, 4) specification of the quasi-structured grid, 5) starting the process of solving the boundary value problem defined.

**The toolbar** is shown in Figure 2. It contains 16 buttons divided into four groups. All buttons have tool tips.



**Figure 2.** Toolbar

In the first group, there are three buttons intended for work with files. These buttons duplicate commands of the **File** menu item.

In the second group, there are three buttons for zoom the domain in the visualization window — first one shows the whole allocation domain and last two buttons apply zoom-in and zoom-out operations. Additionally, the zoom scale can be chosen as a percentage in the combo-box.

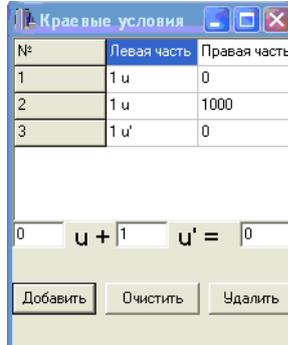
In the third group, there are eight buttons for domain specification and edit. The first four buttons are intended for objects selection: mouse selection tool (adds an object to selection when the **Ctrl** key is pressed), rectangular selection tool, circle selection tool, and freehand selection tool. The next three buttons allow user to set a point, a straight line, or a circle arch. The last button is intended for removing selected objects.

In the last group, there are two buttons for viewing numerical parameters of the problem. The first button click shows a table containing the information about geometrical objects. The last button click shows a data table with basic points' coordinates.

**Dialog boxes.** In Figure 3, a dialog box for setting boundary conditions is shown as an example. At the bottom of the input area, there is a template for setting a concrete boundary condition.

The **Add** button adds a new boundary condition specified in the template to the list of boundary conditions, the **Remove** button removes a selected boundary condition from the list, and the **Clear** button removes all boundary conditions.

Another example of the dialog box is shown in Figure 4. The box displays the table of numerical data about geometrical objects specified. It contains



id	x	y	type	BC
0	151	113	Point	
1	461	381	Point	
2	224	451	Point	
3	144	259	Point	
4	89	169	Point	
5	0	1	Line	0
6	1	2	Line	0
7	2	3	Line	0
8	3	4	Circle	100 0

**Figure 3.** Boundary conditions dialog      **Figure 4.** Domain parameters table

an object ID, starting and ending points coordinates, an object type, radius (for a circle arch), and a boundary condition number. This dialog box allows user to view, input, and edit data.

**Work with GUI.** A user can work with GUI in two ways: by specifying the necessary information via menu commands and toolbar buttons in an order of his/her will or using the Wizard providing data preparation in a step-by-step order, which consequently opens necessary dialog boxes.

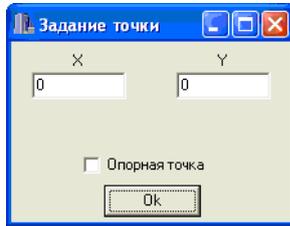
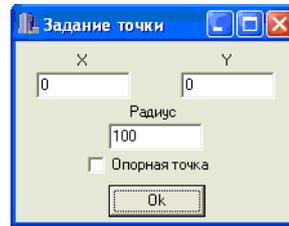
The basic operation often applied in the GUI is an input of a geometrical object, that is, a straight line or a circle arch. A connected sequence of geometrical objects forms a contour of the calculation domain boundary.

The user can specify a geometrical object in any of three ways: graphically, numerically, or using the combined approach. When mouse is moved in the visualization window, user coordinates of the mouse cursor are displayed in the bottom left corner of the application window.

Using mouse selection tool, it is possible to select any edge point of the boundary objects by the left mouse click near a point. To select a number of edge points, do the left mouse click with holding the **Ctrl** key pressed. A selected edge point is marked with a small square. Other edge points are marked with circles.

A new straight line object is created graphically in the usual way by the left mouse click at starting and ending points of the straight line in the visualization window. The ending point of the straight line is marked as a selected point. To build a new straight line piece connected with the previous one, just click on a new ending point. This point is automatically connected with the ending point of the previous straight line and is marked as a new selection.

To create a new point numerically, click the **Point** tool in the toolbar, then populate the point coordinate values for the point in the dialog box appearing (Figure 5).

**Figure 5.** Straight line point setting**Figure 6.** Circle point setting

An advantage of the given way is that it is possible to set some points, then select them and thus construct a polyline connecting selected points.

The **Reference point** checkbox in this dialog box allows user to save this point in a special table for the aim of reusing it in other geometrical objects.

Building a circle arch is carried out in the similar way. An essential difference from a straight line is that in the corresponding dialog box there is a possibility of setting a radius of a circle (Figure 6).

When building a contour of the domain boundary, it is possible to connect pieces of straight lines and circle arches by the ways described above.

Any geometrical object can be selected using a selection tool. With mouse selection tool, you can select or add to selection (if the **Ctrl** key is pressed) any geometrical object already specified. With other selection tools, you specify an area for selection and all objects completely laying in this area are selected. A selected object is drawn with the red color unlike a nonselected object drawn with the black color.

The boundary conditions for the problem are set by specification of their kinds in a special dialog box (see Figure 3), selection of geometrical objects, and assignment them a concrete boundary condition.

The input of a quasi-structured grid (4) is carried out in two stages: building a rectangular macrogrid (2) defining a domain decomposition and building subgrids (3) in subdomains. Input of the information at the first stage is carried out in a special dialog box. The direct input of the information at the second stage, which is about subgrids in subdomains, is difficult for the user, as the number of subdomains can be essentially large. The following technique is applied for automation of the given process. A basic subgrid in one subdomain is set. Several subdomains are selected. For each subdomain of this group, a subgrid is constructed. Some other subdomains are selected, in which their own subgrid is constructed, and so on. In subdomains which were not selected at all, the basic subgrid is constructed. Selection is carried out by a rectangle, a circle, or a freehand figure drawn with mouse.

The introduced information can be a subject for editing, which is carried out graphically or by changing numerical data about objects. The graphical way of editing consists in allocation of one object or several objects, their moving and/or changing their position and sizes. For realization of the second way of editing, a special dialogue box consisting of the table of numerical data about objects is shown (see Figure 4). Editing in this case consists in changing the numerical information in the table. Any object can be removed.

The introduced information is a subject for analysis on completeness and compatibility. A warning message is displayed in the case of an error detected.

An important feature of the GUI is allowance for unexperienced user to considerably simplify the work with the help of the Wizard. Invoking the Wizard, the user does not care of opening dialog boxes. It is done by the Wizard, which consequently opens dialog boxes intended for inputting the information and supplies this process with helps.

The introduced information can be saved to a file and then can be loaded from a file.

#### 4. Conclusion

The developed interface allows user to operatively enter the initial data about a two-dimensional boundary value problem and a quasi-structured grid. The special Wizard assists to a notice user. This interface is a component of the program package ERA-DD intended for solving boundary value problems by a domain decomposition method.

#### References

- [1] Malkova S.V., Sveshnikov V.M. About the user interface of a package ERA-2 // *Mathematical Technologies in Problems of Mathematical Physics*. — Novosibirsk: ICMMG, 1998. — P. 50–79.
- [2] Matsokin A.M. *Automation of a Triangulation of Domains with a Smooth Boundary at Solving Equations of the Elliptic Type*. — Novosibirsk, 1975. — (Preprint / Academy of Science of the USSR. Siberian Branch. Computing Center; 15).
- [3] Sveshnikov V.M. Direct method of decomposition without overlapping of subdomains for solving the boundary value problems on rectangular quasi-structured grids // *Computing Technologies*. — 2008. — Vol. 13, No. 2. — P. 105–116.