Field Precision LLC

Mesh 8.0

Conformal Triangular Mesh Generator

PO Box 13595, Albuquerque, NM 87192 U.S.A.
Telephone: +1-505-220-3975
Fax: +1-617-752-9077
E mail: techinfo@fieldp.com
Internet: http://www.fieldp.com

Contents

1 Introduction

The finite-element technique is based on the division of space into small volumes (*elements*). When the element size is much less than the scale for variations of the computed quantities, the governing partial differential equations can be converted to a large set of linear equations. Such a set can be solved on a digital computer. Unique physical properties are associated with each element. Therefore, elements should not straddle the surface between different material types. The implication is that the boundaries between elements should follow the boundaries between objects in the solution space. A spatial division that meets this criterion is called a *conformal mesh*.

The **Mesh** program handles the conformal division of space for two-dimensional solutions. The information generated is passed to technical programs like **EStat** and **PerMag** for solution and analysis. **Mesh** performs the following functions:

- Definition of object boundaries.
- Transformation of boundary information to a set of conformal triangles.
- Mesh visualization and repair.

The first function is usually performed with the **Mesh** *Drawing Editor*. In this full-functioned CAD program, you define the boundaries of physical objects as a set of line and/or arc vectors. The editor can also import information from other CAD programs in the form of `DXF` files. **Mesh** performs the second function automatically following your specifications of element sizes to ensure numerical accuracy. The program has extensive plotting capabilities for completed meshes with the option to fine-tune mesh parameters in selected areas.

To get started, we suggest you read the first two sections of the following chapter. Then, run the walkthrough example at the beginning of the manual for the solution program(s) in the package you obtained. Return to this manual when you are ready to build you own solutions:

- Chapter 3 discusses essential background to understand how **Mesh** implements triangular meshes.
- Chapter 4 introduces the *Drawing Editor* through a detailed example. Reference material on the features of the editor is given in Chap. 5.
- Chapter sect:scripttomesh shows how to run the program to generate meshes, and review the plotting and repair capabilities.
- You can also enter vector information on object boundaries directly with a text editor or with text input from an external program. This feature is useful for making quick changes to a system or modifying a geometry by an external control loop. Chapter 7 discusses the format of the **Mesh** input script while Chap. 8 covers advanced capabilities controlled through script directives.
- Finally, Chap. 9 describes a unique advanced capability, the creation of meshes directly from photographic information.

2 TC - TriComp program control

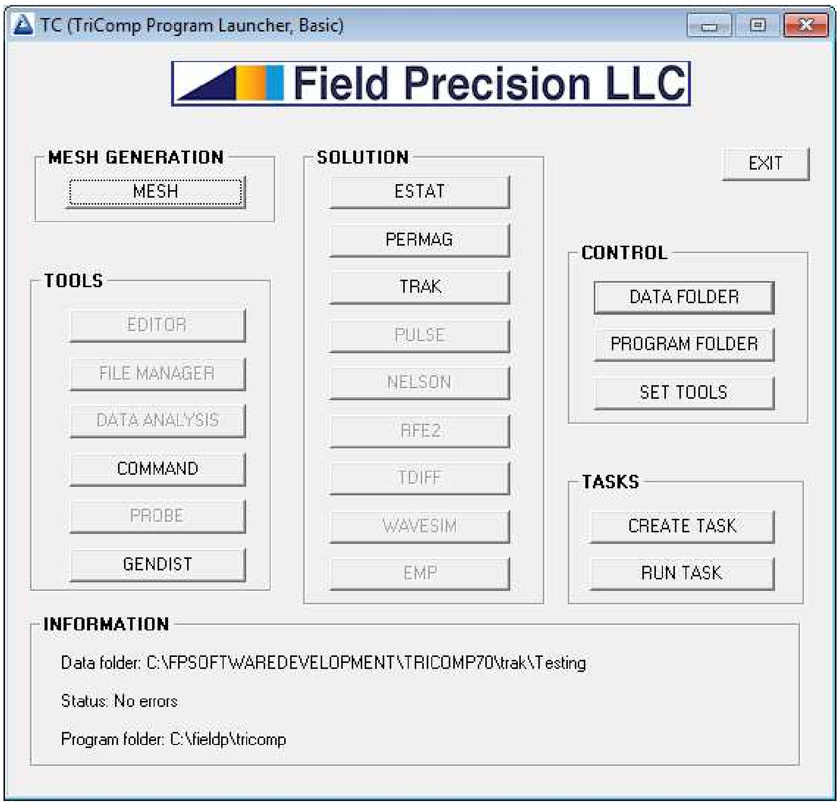
2.1 Introduction

The first step to run **TriComp** programs is to set up **TC**, the program controller. **TC** helps you to organize your work in two ways:

- Only one shortcut is required on your desktop, minimizing clutter.
- All programs open in a specified data directory, eliminating redundant trips through the directory tree.

The installation program sets up a shortcut to `tc.exe` on the Windows desk top and in the start menu.

The **TC** display is divided into five function groups: *Mesh generation*, *Solution*, *Tools*, *Control* and *Information*. The first two groups reflect the activities of a **TriComp** simulation: first create a mesh and then proceed to the physical solution and analysis.



Data folder: C:\FPSOFTWAREDEVELOPMENT\TRICOMP70\trak\Testing

Status: No errors

Program folder: C:\fieldp\tricomp

Figure 1: **TC** screenshot

Table 1: **TriComp** solution programs

Program	**Function**
EStat	Electrostatics with dielectrics or conductors. Supports anisotropic materials.
PerMag	Magnetostatics with coils, permanent magnets and magnetic materials. Supports anisotropic and non-linear materials.
Trak	Charged-particle transport and gun design. Uses field input from **EStat** and **PerMag**.
Pulse	Dynamic magnetic fields with arbitrary time-variation. Supports non-linear magnetic materials and eddy current effects.
Nelson	AC/RF magnetic fields in the non-radiative limit with eddy-current effects. Applications to high-frequency magnetic equipment and induction heating.
RFE2	RF electric fields in the non-radiative limit for RF heating and biomedical applications.
TDiff	Static and dynamic thermal transport in solid materials with temperature-dependent properties. Option to accept input power distributions from **RFE2**, **Nelson** and **WaveSim**.
WaveSim	Frequency-domain solutions for electromagnetic fields. Application to resonators, microwave equipment and scattering.
EMP	Time-domain electromagnetic solutions with applications to pulsed-power equipment, digital circuits and electromagnetic interference.

2.2 Setting the program and data directories

Figure 1 shows a screen shot of the **TC** dialog. Note that only some of the command buttons are active. **TC** activates buttons only if detects the corresponding executable in the *program directory*. Table 1 summarizes the functions of programs included as buttons in the **TC** *Solution* section. The button for **Mesh** should always be active, while entries in the *Solution* section depend on which **TriComp** programs you have purchased. If no buttons are active, **TC** is not pointing to the directory where you installed the executable programs. In this case, click the *Program Folder* button. In the dialog move to the directory that contains the executables and click *OK*. The corresonding buttons in the *Mesh* and *Solution* areas will become active.

A good practice is to collect all input and output files for related solutions in a specific data directory. Use the *Data Folder* button to set the current location. Programs that are opened after the change will read and write to the directory. (Note that the setting does not affect previously-opened programs.) The information area at the bottom of the control box shows: 1) the current data directory, 2) the last operation performed and 3) the program directory.

2.3 Tools

This section and the next cover advanced capabilities of **TC**. You will probably often use additional software tools working with **TriComp** and other technical programs. You can set **TC** to launch your favorite utility programs. To define a text editor, press the *Set tools* button in the *Control* section. In the following dialog, choose the *Editor* option. Then, move to the appropriate directory and select the program. When you exit the procedure, the *Editor* button in the *Tools* section becomes active. The freeware **ConText** editor has been included along with syntax-highlighting definition files for **TriComp** input files. You can use the *Set tools* command to define two other utilities. For the file manager option, select a program like **FreeCommander** or **Windows Explorer**. Use the *Data analysis* button for a spreadsheet, plotting program or mathematical analysis software.

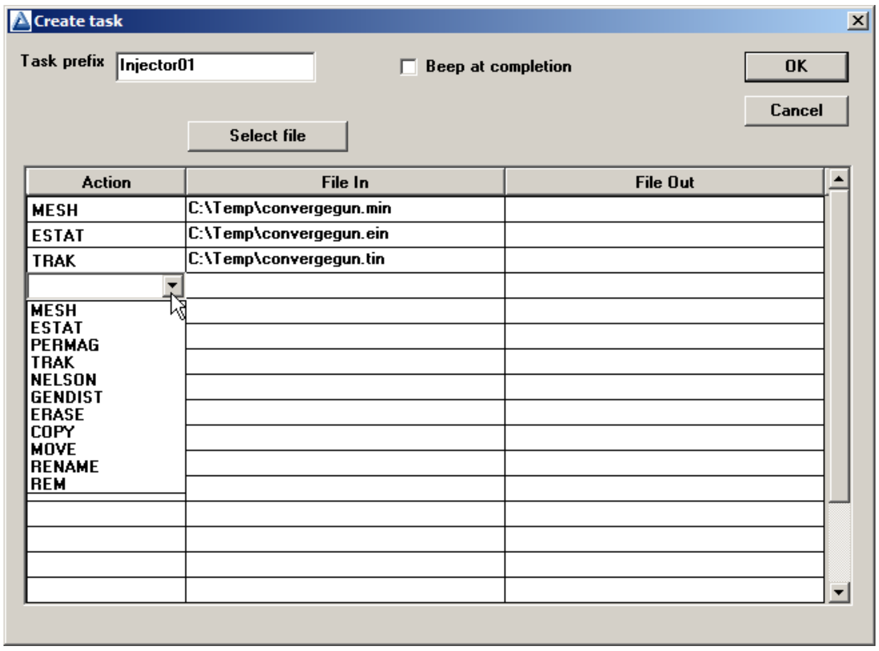
The *Command* button opens a DOS window if you want to run programs from the command prompt or under the control of your own batch files. The **Probe** utility plots standard history files created by initial-value solution programs (**Pulse**, **TDiff** and **EMP**). Finally, **GenDist** is used to generate, to plot and to analyze distribution files for the **Trak** program.

2.4 Creating and running tasks

TriComp programs are optimized for the latest generation of multi-core or multi-processor PCs. The Professional solution programs (**EStat**, **Permag**, **Trak**,) feature true parallel operation. A second feature in both the Basic and Professional programs is the capability to run multiple independent calculations simultaneously. All solution programs can run in the background if launched from a Windows batch file. Background operation is automatic and faster than running in a window. The *Create task* and *Run task* commands in **TC** make it easy to use batch files. With the commands you can 1) quickly define multi-step calculations (*tasks*) in an interactive dialog, 2) launch simultaneous tasks in the background and 3) find out which tasks are running.

The *Create task* button calls up the dialog of Figure 2. Supply a file prefix `FPREFIX` that indicates the function of the task. The task information will be stored in a DOS batch file `FPREFIX.BAT` created in the current **TC** data directory. Commands in the file are compatible with all recent Windows versions including Windows 7. Each row represents an operation (batch file command). The first column defines the action. Clicking on a cell brings up a menu that includes all **TriComp** programs capable of background operation that are installed on the users computer. In addition, several relatively safe DOS commands are included (`ERASE`, `COPY`, `MOVE`, `RENAME` and `REM`). All commands operate on a file (*FileIn* column). The DOS commands `COPY`, `MOVE` and `RENAME` require a second file name (*FileOut* column). You can type file names in the cells. By default, the files are in the **TC** working directory, but you can include path information if the files are in other directories. Alternatively, you can click in a cell and then pick the *Select file* command to use the standard Windows dialog for choosing files anywhere on the computer. For the **TriComp** programs, the dialog displays only files with appropriate suffixes (*e.g.*, `*.EIN` or `*.SCR` for **EStat**).

Click the *OK* button when the sequence is complete to create the batch file. Here is an example:



Action	File In	File Out
MESH	C:\Temp\convergegun.min	
ESTAT	C:\Temp\convergegun.ein	
TRAK	C:\Temp\convergegun.tin	

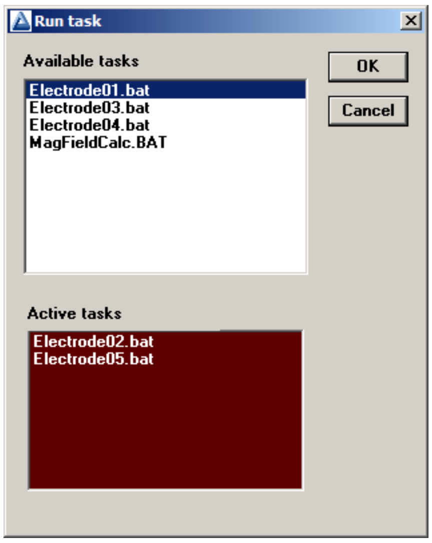
Figure 2: Create task dialog

```
REM TriComp batch file, Field Precision
START /B /WAIT C:\fieldp\tricomp\mesh64.exe C:\Temp\convergegun
START /B /WAIT C:\fieldp\tricomp\estat64.exe C:\Temp\convergegun
START /B /WAIT C:\fieldp\tricomp\trak64.exe C:\Temp\convergegun
ERASE *.?ls
START /B /WAIT C:\fieldp\tricomp\notify.exe
IF EXIST Electrode01.ACTIVE ERASE Electrode01.ACTIVE
```

The operations listed perform a complete **Trak** calculation in the background and then erase all listing files. The example has some notable features:

- The operations are performed sequentially because data from one operation may be used in the next. To run calculations in parallel, define and run multiple tasks.
- You can modify the file with an editor if you are familiar with DOS commands.
- The DOS commands recognize the standard wildcard conventions (* for any character grouping, ? for any character).
- The programs adds the command `notify.exe` to the task sequence if *Audio alarm* was checked. In this case, the computer beeps when a task is completed.
- The final command to erase a file `FPREFIX.ACTIVE` is added to all batch files. The presence of the file indicates that the task is running.

Click the *Run task* button when you have created tasks or moved predefined task files to the data directory. The dialog (Figure 3) organizes tasks into two groups: ones that are available to run and ones that are currently running (*i.e.*, `FPREFIX.ACTIVE` has been detected). To launch a



Electrode01.bat
Electrode03.bat
Electrode04.bat
MagFieldCalc.BAT

Electrode02.bat
Electrode05.bat

Figure 3: Run task dialog

task, choose one from the top list and click *OK*. The program creates a file `FPREFIX.ACTIVE` and runs the batch file. The program sequence runs silently in the background. In the meantime, you can prepare other inputs or run other tasks.

3 Mesh generation concepts

3.1 Foundation mesh

The first step in a finite-element solution is to divide the solution volume into small parts (*elements*). **Mesh** creates elements for two-dimensional solutions with triangular cross-sections in the x-y or z-r planes. There are three steps in the mesh generation procedure:

- Fill the solution cross-section with an baseline set of triangles of appropriate sizes.
- Shift nodes so that the element facets lie close to material boundaries.
- Set the region identity of nodes and elements.

We shall refer to the first operation as generation of the *foundation mesh* and to the second as *conformalization* of the elements.

The foundation mesh is a set of approximately uniform triangles that fill a rectangular area large enough to contain the solution. Figure 4 shows an example. The active solution area may or may not fill the entire rectangle. Because the foundation mesh is logically structured, we can determine elements surrounding a node and neighboring nodes using index operations. **Mesh** analyzes the boundary vectors of each object you define and moves selected nodes so that they lie on given points, lines or arcs. The node displacements must be small enough so that all sets of three connected points in the resulting mesh form valid triangles. Conformalization may fail if the foundation mesh is poorly suited to the object boundaries. Therefore, it is essential to pick good dimensions for the triangles and sometimes to introduce variations in triangle size (*variable resolution*).

While working in **Mesh**, we refer to the coordinate axes as x rr z (horizontal direction) and y or r (vertical direction). The designation of planar or cylindrical symmetry is for user convenience and does not affect the operation of **Mesh**. The interpretation of elements as columns of infinite extent in the z direction or figures of revolution about z occurs only in the **TriComp** solution programs. There are two points to remember in preparing a mesh for a cylindrical solution:

- The horizontal axis corresponds to the z direction and all objects must be symmetric in θ.
- The vertical axis corresponds to r. The minimum coordinate value along the vertical direction must satisfy $r \geq 0.0$. Remember that a z-r plot is not the same as a longitudinal section through a cylindrical system. Negative values of r are undefined.

The foundation mesh rectangle extends from x_{min} to x_{max} (or z_{min} to z_{max}) along the horizontal axis and from y_{min} to y_{max} (or r_{min} to r_{max}) along the vertical axis (Fig 4). The node indices are in the range $1 \leq K \leq K_{max}$ along x or z and $1 \leq L \leq L_{max}$ along y or r. Because extra points must be added to implement boundary conditions, the total number of points in the foundation mesh is $I_{max} = (K_{max} + 2)(L_{max} + 2) - 1$.

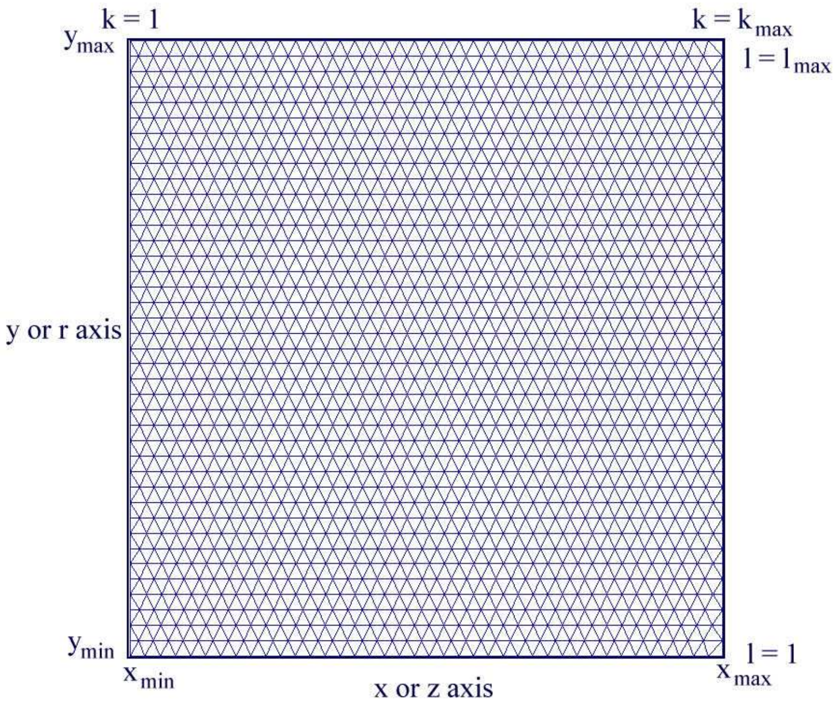


Figure 4: Foundation mesh parameters

3.2 Regions

A *region* is an area of the solution space where nodes and elements have common material properties. **Mesh** assigns an integer number (from 1 to 250) to all nodes and elements of a region. The numbers are assigned in the order that the regions appear in the script. In **EStat** solutions, regions may represent electrodes, dielectrics, space-charge clouds, fixed-potential boundaries or symmetry boundaries. Alternatively, in a thermal solution regions could represent fixed-temperature bodies, heat sources or materials with different values of thermal conductivity, mass density and heat capacity. The regions defined in **Mesh** are neutral – the same mesh could be used for a variety of physical solutions.

There are two categories of regions: *filled* and *open*. The boundary of an open region may consist of any collection of points, lines and arcs. In this case, **Mesh** assigns the current region number in the processing sequence to nodes on the region boundaries and does not renumber any elements. For example, in **EStat** open regions often represent symmetry or fixed-potential boundaries. An open region consisting of a set of unconnected points can be used to simulate a wire grid. The lines and arcs that define an open region may be connected or unconnected at their end points. The only limitation is that the vectors of a single region may not cross each other. Filled regions represent solid bodies like electrodes or dielectrics. Here, line or arc vectors define a closed curve in x-y or z-r space. After processing the boundary, **Mesh** assigns the current region number to all elements and nodes enclosed by the boundary.

To conformalize a single point, **Mesh** moves the nearest unclamped node to a location on the region boundary and sets the node region number. The program clamps the node in position so that it is not moved by subsequent operations. For lines and arcs, **Mesh** firsts displaces and clamps nodes at the start and end points. The program then walks from the start to the end along a path that moves between logically-connected points, displacing and clamping points that are closest to the vector. The process yields a series of triangular elements with sides aligned along the boundary vector.

3.3 Region order

The order in which regions appear in the **Mesh** script is important because the currently-processed region overwrites any previously-defined region numbers assigned to nodes or elements in the shared space. For example, suppose we wanted to calculate electrostatic fields near a circular wire with an insulating jacket. The procedure is first to define a circular filled region to represent the insulating dielectric. This is followed by a region that defines a smaller circle inside the first to represent the constant-potential wire cross-section. All elements inside the smaller circle will assume the second region number. The nodes on the wire boundary will have the fixed-potential condition because the wire region was entered last.

The overwrite process requires that regions with special boundary conditions should appear near the end of the script so that shared nodes on the boundary assume the correct region number. For example, suppose we have a solution where an electrode and dielectric share a common surface. The electrode region must appear *after* the dielectric so that nodes on the common surface retain the number of the constant-potential region. If you observe a solution that appears to have non-physical field lines, the likely cause is incorrect region ordering in the **Mesh** script.

3.4 Region definition example

The example of Fig. 5 clarifies how to set up regions in **Mesh**. In an electrostatic solution inside a grounded box, upper and lower electrodes with mirror symmetry about $y = 0.0$ have potentials +1500 V and -1500 V respectively. The problem has two symmetry planes. The potential satisfies the condition $\phi(x, y) = -\phi(x, -y)$. Therefore, the condition $\phi = 0.0$ holds in the plane $y = 0.0$. Similarly, the potential has the property $\phi(x, y) = \phi(-x, y)$. This means that the field component E_x equals zero in the plane $x = 0.0$, or $\partial\phi(0, y)/\partial x = 0.0$. We need only determine the potential in the first quadrant, assigning a Dirichlet condition along the boundary at $y = 0.0$ and the special Neumann condition along $x = 0.0$. We can then determine potentials at other location from the symmetry relationships. Figure 5 shows the modified geometry with boundary conditions and region numbers.

The rule in **Mesh** is first to enter all regions that do not have fixed boundary conditions (such as dielectrics) and then to enter fixed-potential objects and Dirichlet boundaries. In this way, important boundaries are not accidently overwritten. To begin, we set elements in the foundation mesh that represent the vacuum region with $\epsilon_r = 1.0$. Region 1 is a filled region that occupies the entire foundation mesh rectangle. The second step is to add the dielectric support as Region 2. This is a closed region inside Region 1. Mesh shifts boundary vertices that have not already been fixed and assigns the number 2 to the nodes and enclosed elements. Region 3 is the electrode. The nodes on the shared boundary with the dielectric are changed to the fixed-potential condition. The unspecified points along the left-hand boundary automatically satisfy the specialized Neumann condition. The final step is to define an open region (Region 4) to set the Dirichlet condition $\phi = 0.0$ along the bottom, right and top boundaries. Figure 6a shows an equipotential plot for the resulting solution. Figure 6b shows a solution where the dielectric support region was incorrectly placed at the end of the script. The potential lines

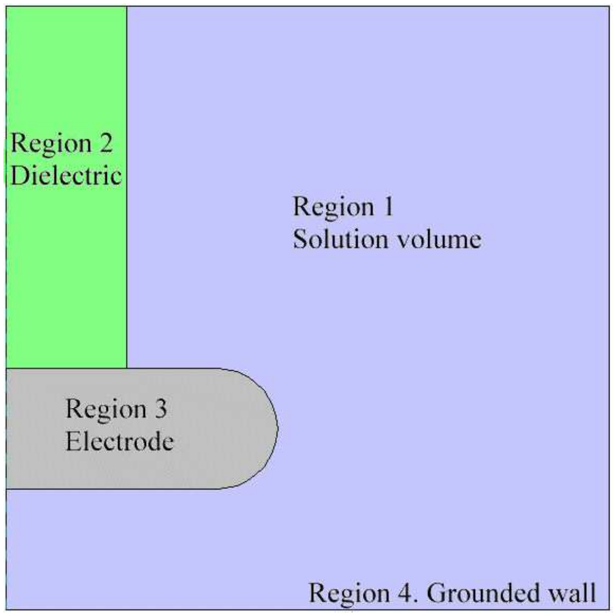


Figure 5: Example illustrating the effect of region order. Only the first quadrant is shown.

exhibit non-physical variations where the dielectric contacts the fixed-potential electrode and boundary.

3.5 Choosing element sizes

Although **Mesh** handles most operations automatically, you must sometimes make decisions on the sizes of elements in the foundation mesh. The program defaults are at best a guess, and human judgement is necessary to achieve the optimal balance of accuracy versus run time. The choices become easier with experience. The following pointers may help to get you started.

- As a minimal requirement, elements must be small enough to resolve objects in the solution space. Large elements may lead to errors in the mesh generation process.
- The visual appearance of a mesh is usually a good guideline – curves should be relatively smooth.
- A common error of first-time code users is to include too many elements. Your goal should be to achieve good accuracy with the shortest run time. Bulldozing through a solution by creating huge meshes leads to inelegant solutions that are slow to generate and difficult to analyze. Furthermore, reducing element size doesn't always improve accuracy. For tiny elements, accumulated floating-point errors may actually degrade accuracy.
- A key issue is how to know when elements are small enough. Here, the important point to remember is that numerical codes need not be mysterious black boxes. To gain confidence, you can perform benchmark calculations to make comparisons with theory and/or experiments.
- A second issue is how to check accuracy in a calculation of a complex system where an exact benchmark is not possible. The standard approach in numerical work is to repeat

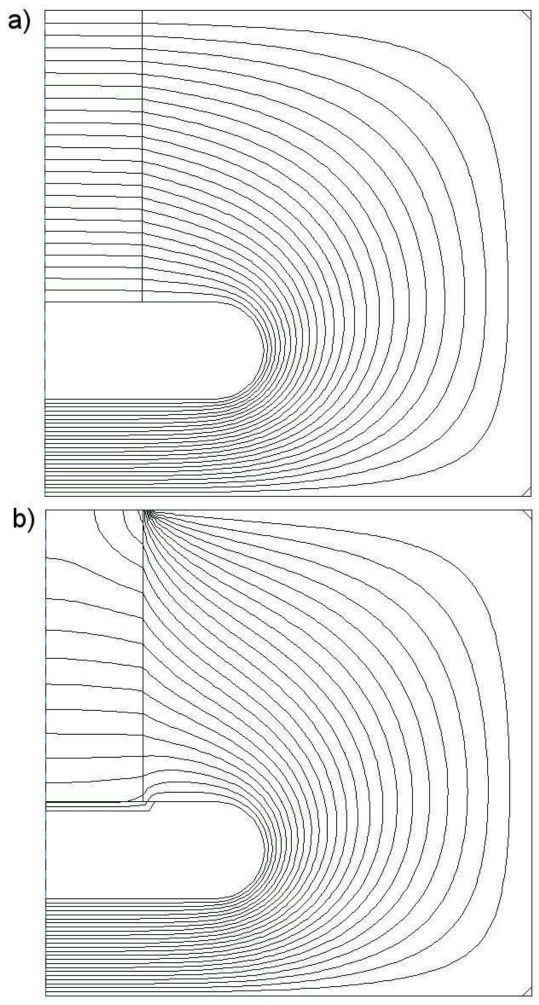


Figure 6: Equipotential lines for the example. *a*) Correct region ordering. *b*) Incorrect region ordering.

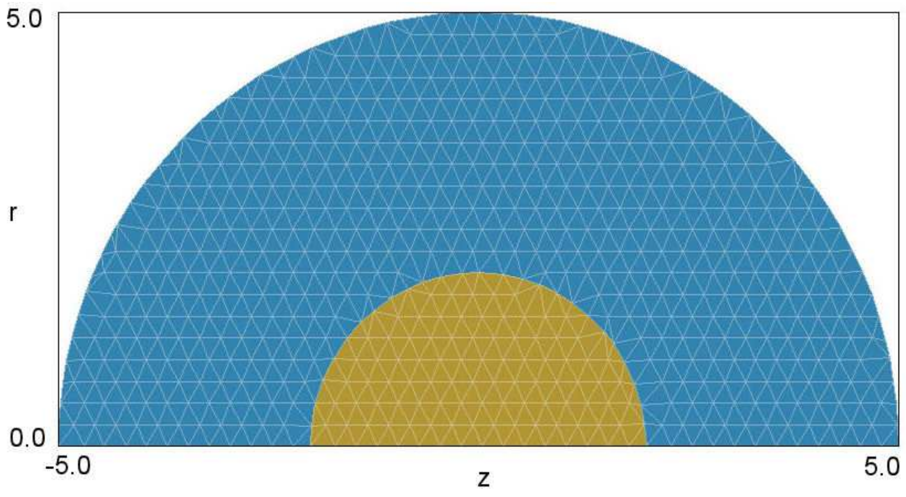


Figure 7: Spherical capacitor benchmark example with element size 0.25 cm.

the calculation with reduced element size and to compare results. If changes are relatively small, the initial element size was sufficient. For this process, you should exercise caution if objects in the solution space have sharp edges. Although the field may be correct in the spaces between such objects, calculated quantities at the discontinuity are undefined. Reducing the element size simply gives a different value of numerical infinity, a meaningless quantity.

To illustrate a benchmark calculation, consider the spherical capacitor of Fig. 7. The system consists of an air space between electrodes with an inner radius of R_i and outer radius R_o. The theoretical capacitance is given by:

$$C = \frac{4\pi\epsilon_0}{1/R_i - 1/R_o}. \tag{1}$$

For $R_i = 2.0$ cm and $R_o = 5.0$ cm, Eq. 1 yields C = 3.709 pF. An electrostatic calculation was performed for the geometry of Fig. 7 with different uniform element sizes. The figure shows the coarsest mesh with triangle base and height approximately equal to 0.25 cm. The capacitance was determined by two methods: 1) a volume integral of electrostatic field energy over the air elements and 2) a surface integral of the normal component of electric field to determine induced charged. Table 2 lists the results. The volume integral method provided good accuracy (0.24%) at the coarse resolution, and there was no discernable error for element size ≤ 0.10 cm. The surface integral method improved continuously with decreasing element size. The error was about 5.0% at 0.10 cm and 0.48% at 0.01 cm. The calculation time increased from less than 1.0 s at 0.25 cm to 69.0 s at 0.01 cm.

Table 2: Results – benchmark simulation of a spherical capacitor

Element size (cm)	**Capacitance** Volume integral (pF)	**Capacitance** Surface integral (pF)
0.25	3.718	3.259
0.10	3.710	3.524
0.05	3.710	3.616
0.02	3.708	3.670
0.01	3.708	3.690

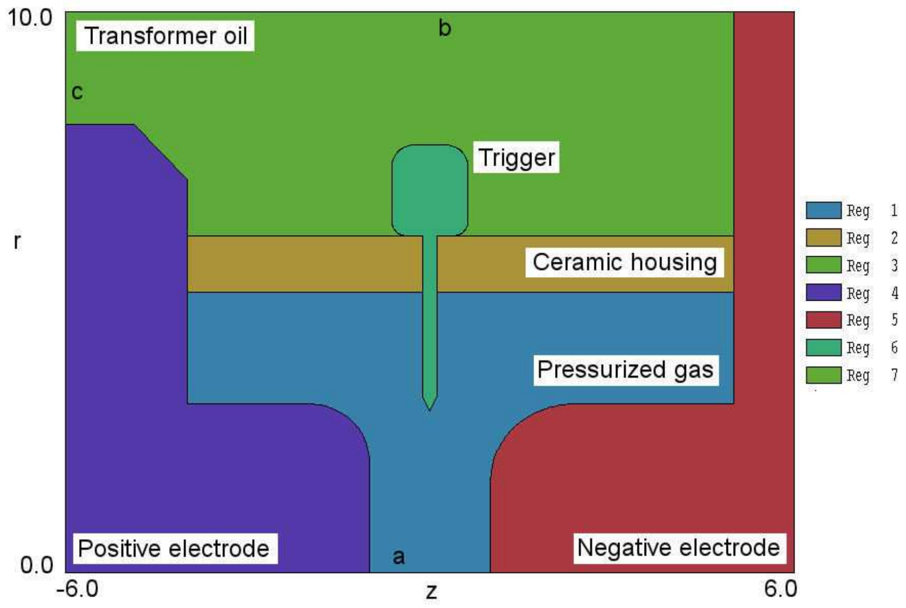


Figure 8: Application example – high-voltage spark gap switch

4 Building a Mesh script with the drawing editor

This chapter describes controls and capabilities of the **Mesh** drawing editor, a versatile two-dimensional CAD utility. Using the editor is the fastest way to create a script that describes your application geometry. To help you learn the editor functions, the discussions in this chapter are geared to the example shown in Fig. 8. You can follow the step-by-step instructions to create your first **Mesh** script. The following chapter gives details on the drawing editor capabilities.

4.1 Starting a new mesh

Before starting operations with the program, it is a good practice to plan a strategy. It is helpful to make a sketch of the geometry and to determine the required regions of the solution volume and the order for adding them to the mesh. The example of Fig. 8 is a spark-gap switch for a pulsed-power system. The assembly has cylindrical symmetry about the z axis (horizontal). The bottom edge of the solution volume may correspond to the axis ($r_{min} = 0.0$) or to a point off axis ($r_{min} > 0.0$).

The following unique material regions are required for an electrostatic solution:

- Region 1. High-pressure gas inside the spark-gap housing ($\epsilon_r \cong 1.0$).
- Region 2. Ceramic to contain the high-pressure gas ($\epsilon_r \cong 7.8$).
- Region 3. Transformer oil outside the switch ($\epsilon_r \cong 2.7$)
- Region 4. Positive high-voltage electrode

- Region 5. Negative high-voltage electrode
- Region 6. Trigger electrode

Following the discussion of Sect. 3.3, dielectric regions appear at the beginning of the list and fixed-potential regions at the end. The regions listed above have the *Filled* property – they have non-zero volume and are outlined by a closed set of line and arc vectors. We also need to deal with sections of the solution-volume boundary that are not occupied by fixed-potential electrodes (Dirichlet condition). By symmetry, there is no radial component of electric field in the on-axis gap between the electrodes (section marked a). This section of the boundary satisfies the condition:

$$\frac{\partial \phi}{\partial r} = 0.0. \tag{2}$$

Equation 2 is the special Neumann condition. This condition occurs automatically on unspecified boundary sections in a finite-element solution. We also leave the boundary at $r = r_{max}$ (section b) in the natural condition. In this case Eq. 2 does not hold exactly; nonetheless, a special Neumann boundary is an acceptable approximation if we are interested mainly in the fields near the gap. Finally, the section of the wall marked c should be at same potential as the positive electrode. To set the fixed-potential condition, we include an open line region of nodes designated as Region 7.

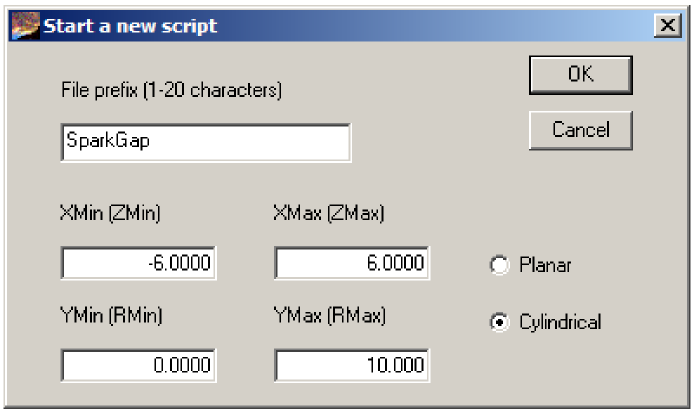
As part of the strategy, we need a plan for outlining the regions. We could make an exact outline for each region, but that would involve more work than is necessary. Instead, we take advantage of the **Mesh** overwrite principle with the following procedure:

1. Set up Region 1 (gas) as a rectangle that fills the entire solution volume.
2. Add Region 2 (ceramic) as a rectangle that extends the full axial length of the solution volume ($z = -6.0$ to $z = 6.0$) and between $r = 5.0$ and $r = 6.0$.
3. Add Region 3 (oil) as a rectangle that fills the space $-6.0 \leq z \leq 6.0$ and $6.0 \leq r \leq 10.0$.
4. Outline the fixed-potential electrodes (Regions 4-6) and the line along the boundary (Region 7).

To review, we start with the following information: 1) a sketch of the simulated system with dimensions, 2) a list of required regions and 3) a strategy to create the regions. We are now ready to begin work with the program.

4.2 Filling the solution volume and making a backup

To follow the operations described in the next sections, run **Mesh** from the **TC** program launcher. Click on *File/Create mesh/Graphics* or use the tool to begin a new script in the drawing editor. The program displays the dialog shown in Fig. 9. The file prefix is a descriptive name that will be applied to the **Mesh** input script (`SparkGap.MIN`) and output file (`SparkGap.MOU`). The numerical fields define the boundaries of the solution rectangle. Enter the dimensions shown in Fig. 9, check the radio button marked *Cylindrical* and click *OK*. Figure 10 shows the initial screen display.



Start a new script

File prefix (1-20 characters): SparkGap

XMin (ZMin): -6.0000

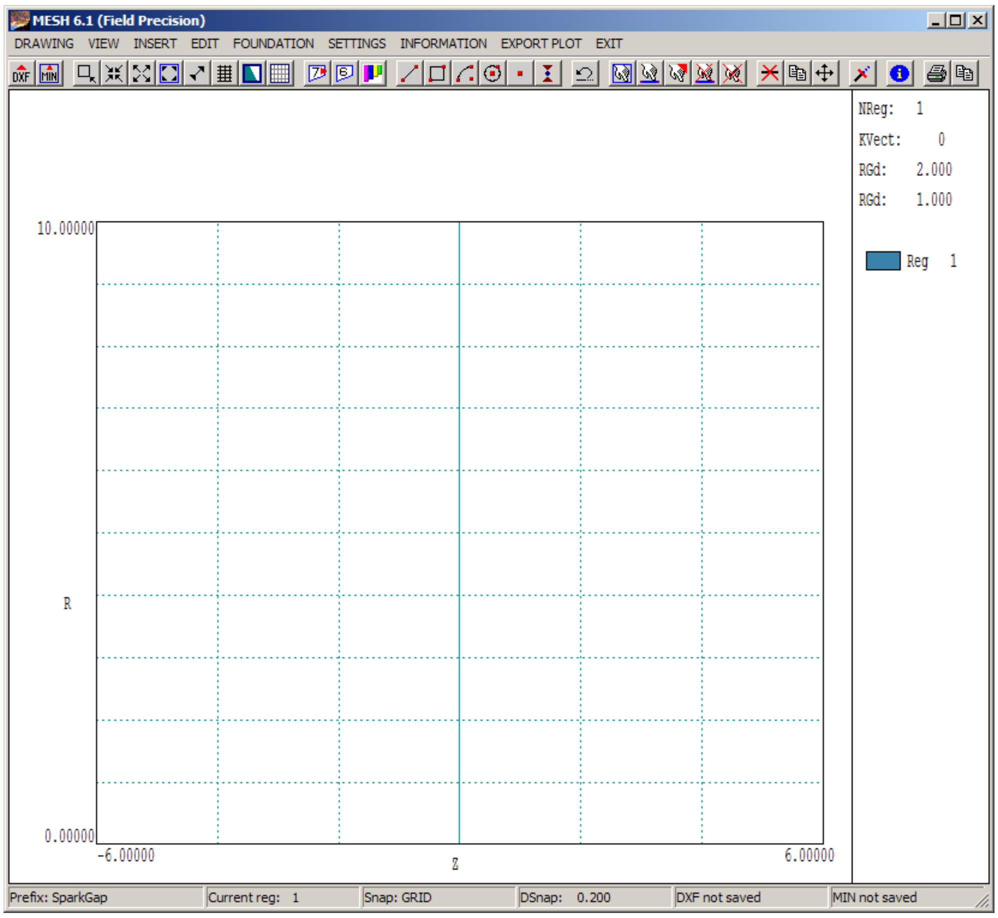
XMax (ZMax): 6.0000

YMin (RMin): 0.0000

YMax (RMax): 10.000

Planar [✓ Cylindrical]

Figure 9: Dialog to begin a new script in the graphics mode.



MESH 6.1 (Field Precision)

```
NReg:    1
KVect:     0
RGd:    2.000
RGd:    1.000
```

🟦 Reg 1

Prefix: SparkGap | Current reg: 1 | Snap: GRID | DSnap: 0.200 | DXF not saved | MIN not saved

Figure 10: Initial working environment of the drawing editor.

Take a moment to review the arrangement of the screen. Note that the main menu and toolbar at the top have been replaced by a set of drawing commands. The information window on the right-hand side shows the status of the drawing. Initially, there are no drawing vectors. The drawing window, which occupies the main part of the screen, shows only the boundaries of the solution rectangle defined in the dialog of Fig. 9. **Mesh** has picked a set of convenient display grid intervals – they are listed in the information window (2.0 in the horizontal direction, 1.0 in the vertical direction). The intervals change automatically as you zoom the view. The status bar at the bottom of the screen lists the script prefix and the current region for vector entry (initially set to Region 1). The bar also lists the mouse snap mode and snap grid distance (0.2). Input via the snap mode ensures that the endpoints of vectors that outline filled regions connect exactly. Section 4.4 gives more details about the snap mode. For the present, recognize that default snap mode is to points on the snap grid and that the mouse coordinates change in discrete units of 0.2.

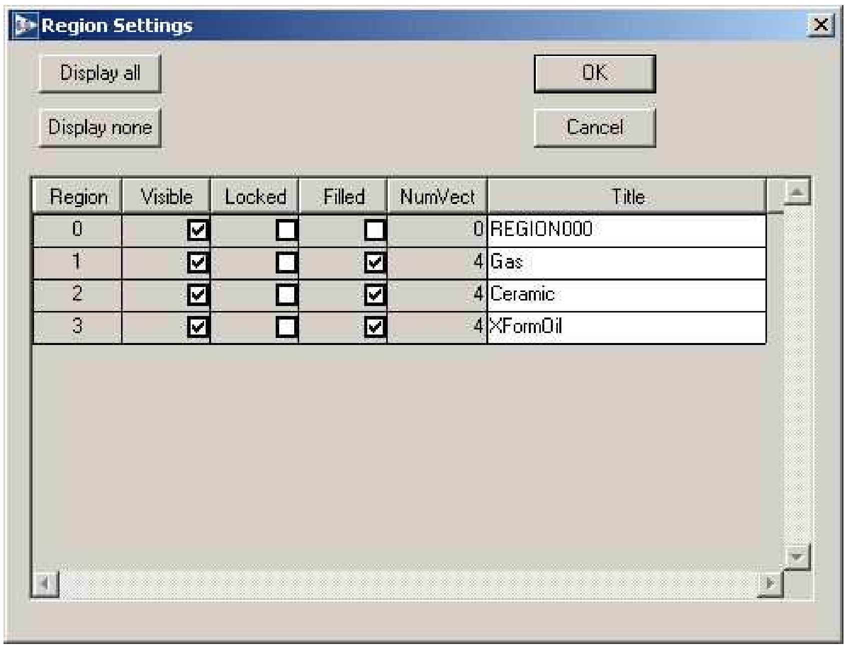
The first step to build the drawing is to define line vectors for Region 1 that outline the solution rectangle. Chose *Insert/Rectangle* from the menu or click on the rectangle tool. Note that the status bar changes to *coordinate entry mode*. Here, the first box in the bar displays a short instruction. The second and third boxes show the x and y (or z and r) coordinates. Move the mouse cursor toward the bottom-left corner of the solution rectangle. As the mouse moves, the coordinates change in steps of 0.5. Mesh also displays the magnetic snap point, a red box that moves between points of the snap grid as you move the cursor. Click the left button when the coordinates reach (-6.0,0.0) to clamp one corner of the rectangle. Then move the mouse to the upper-right corner (6.0,10.0) and click the left button. **Mesh** adds four line vectors displayed in the blue color of Region 1. Note that the entry for Region 1 in the information window has the letter *F* appended to show that the region has the *Filled* attribute. This is the default setting because Region 1 must always fill a non-zero volume. Additional regions initially have the *Open* attribute. You must identify which ones should be filled (see Sect. 4.3).

We will save the limits and drawing vectors before proceeding to the next step. Click on *Drawing/DXF/Export* or use the tool. Click *OK* to accept the default file prefix of `SPARKGAP`. The program creates a file `SPARKGAP.DXF` which records the information we have defined so far. The bare-bones DXF file is recognized by any 2D CAD program (AutoCAD, TurboCAD,...). In addition to drawing entities, it contains information specific to **Mesh** (such as region names) as comment lines. This information will be ignored by the CAD program. The exported DXF file may serve two purposes:

- Information transfer to other programs.
- Backup of an unfinished drawing.

4.3 Adding more regions and assinging names

Next, we shall outline a rectangular area to represent the cross-section of the ceramic insulator. Some of this area will be overwritten later by the electrode regions. We shall store the four vectors of the ceramic outline as Region 2. Click on *Insert/Start next region* in the menu or use the tool. Note that the current region is updated in the status bar. Again, use the rectangle tool. This time the corner dimensions should be (-6.0,5.0) and (6.0,6.0). If you make a mistake, use the *Edit/Undo last operation* menu entry or tool. In a similar way, add a third rectangular



Region	Visible	Locked	Filled	NumVect	Title
0	✓	☐	☐	0	REGION000
1	✓	☐	✓	4	Gas
2	✓	☐	✓	4	Ceramic
3	✓	☐	✓	4	XFormOil

Figure 11: Dialog to set region properties.

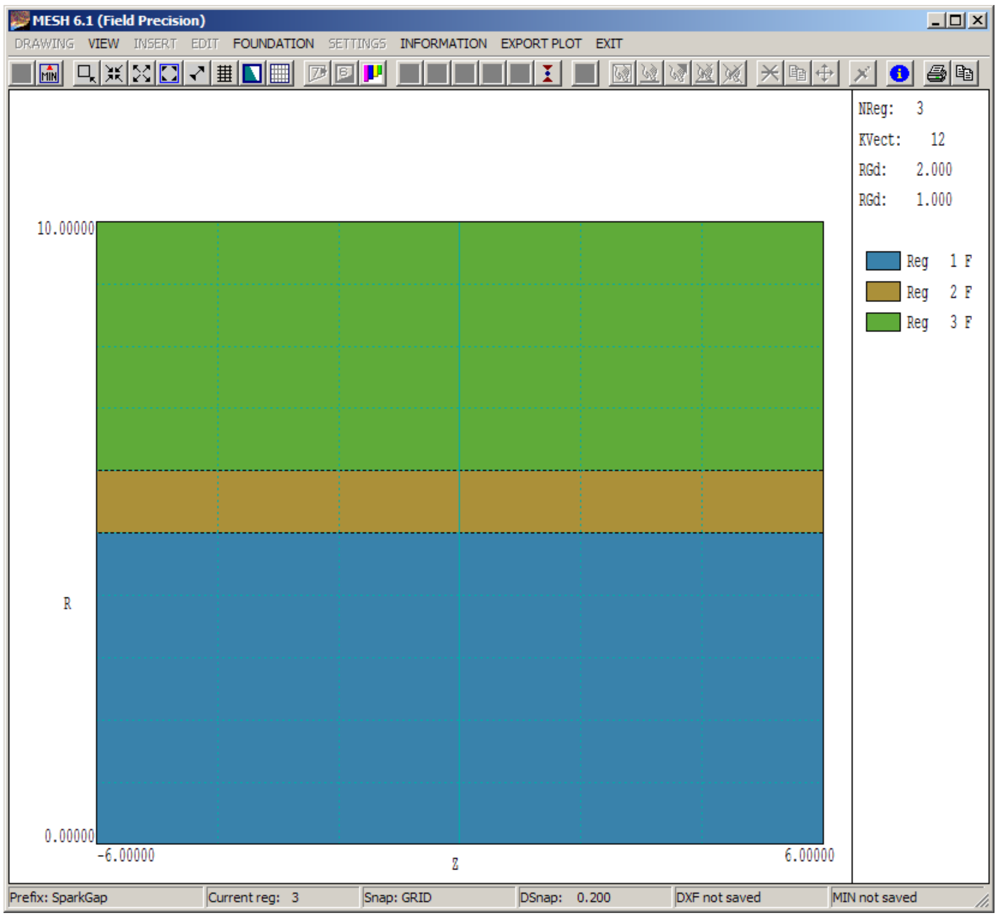
region to represent the transformer oil. Start Region 3 and then use the rectangle tool to create a box with dimensions (-6.0,6.0) and (6.0,10.0).

We now have three regions with generic names, one of which has the *Filled* attribute. Click the menu command *Settings/Region properties* to bring up the dialog shown in Fig. 11. The grid contains row entries for the three regions that we have defined along with an additional one corresponding to Region 0, a special layout region. You can enter alignment or template vectors in the layout region that will not be recorded as part of the **Mesh** script. In the *Filled* column, check the boxes for Regions 2 and 3. To make work easier in the drawing editor, we shall also assign descriptive names. These names will be recorded when we create a script. Figure 11 shows the appearance of the dialog when the operations are complete. Click *OK* to exit and to apply the changes.

We can check that the *Filled* regions are correct (*i.e.*, the vectors are connected and outline closed areas). Click on the menu command *Information/Show fill status* or use the tool. The screen plot changes from a vector display to one that shows the areas occupied by filled regions (Fig. 12). Note that all menu commands for entering and editing vectors have been deactivated. The program must be in the vector-display mode to create new entities. Click on the command again to return to the vector mode. This is a good point to backup your work using the *Drawing/DXF/Export* command.

4.4 Drawing lines, arcs and circles

The shape of region 4 (the positive electrode) is more complex. It presents a good opportunity to learn how to enter individual line and arc vectors and how to control the snap mode. Be sure to click on the command *Insert/Start next region* before performing the following steps. If you make a small mistake, use the *Edit/Undo last operation* command. If you make a big mistake, you can abandon the drawing and return to the last saved version. Click on *Exit/Abandon* to


MESH 6.1 (Field Precision)
DRAWING VIEW INSERT EDIT FOUNDATION SETTINGS INFORMATION EXPORT PLOT EXIT
10.00000
0.00000
-6.00000
6.00000
R
Z
Reg 1 F
Reg 2 F
Reg 3 F

```
NReg:   3
KVect:   12
RGd:    2.000
RGd:    1.000
```

Prefix: SparkGap | Current reg: 3 | Snap: GRID | DSnap: 0.200 | DXF not saved | MIN not saved

Figure 12: Appearance of Regions 1,2 and 3 in the fill display mode.

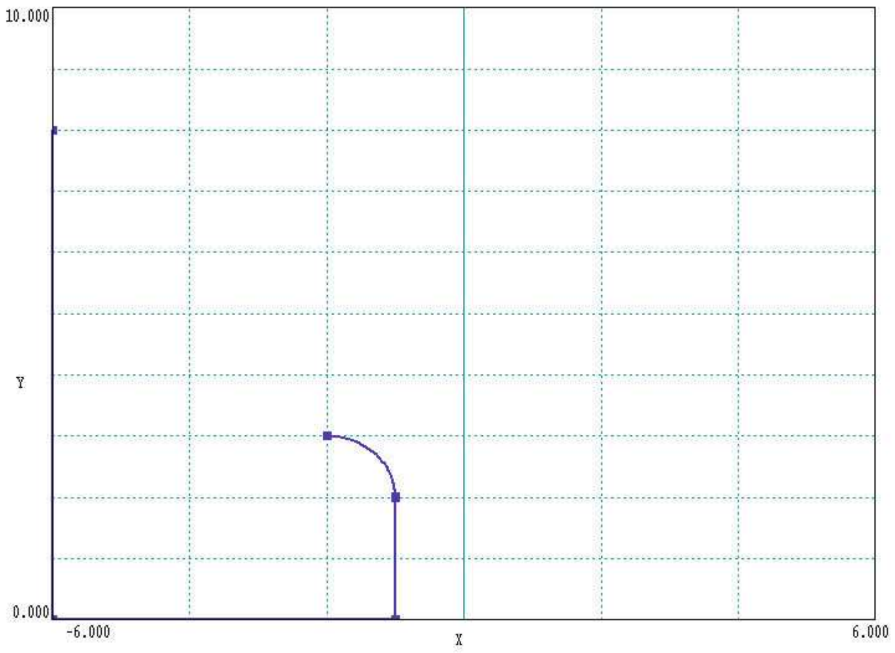


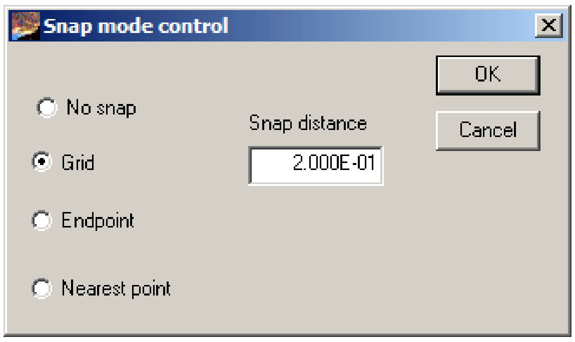
Figure 13: Partial outline of the positive electrode – Regions 1, 2 and 3 are not displayed.

return the main menu without saving the current drawing. Then choose *Create script/DXF* input and pick `SPARKGAP.DXF`.

Click on the *Insert/Line* command or tool. The program enters the *Insert/Coordinate* mode. Here, coordinates are displayed at the bottom of the screen and you can enter several line vectors in one operation. We will draw a portion of the electrode outline. Move the mouse cursor and left-click at (-6.0,8.0) and (-6.0,0.0) to create the first line. Then move the mouse cursor and left-click at points (-6.0,0.0) and (-1.0,0.0) to add a second line. Finally, the third line extends from (-1.0,0.0) to (-1.0, 2.0). Right-click the mouse to exit *Insert/Coordinate mode*.

We will now draw the arc that defines the radius on the edge of the electrode. Click the command *Insert/Arc/Start-End-Center*. **Mesh** again enters *Insert/Coordinate* mode. The arc start point is (-1.0,2.0), the end point is (-2.0,3.0) and the center point is (-2.0,2.0). The shape should look like Fig. 13 (note that the display of Regions 1, 2 and 3 has been suppressed for clarity). Use the command *Insert/Line* to add lines from (-2.0,3.0) to (-4.0,3.0) and from (-4.0,3.0) to (-4.0, 7.0).

The endpoints of the remaining two lines to complete the outline do not fall exactly on snap grid points. To draw them we shall use keyboard entry and endpoint snap. Click the command *Insert/Line*. For the first point, left-click the mouse at the grid point (-6.0,8.0). Instead of entering the second point with the mouse, press the *F1* key to display the keyboard entry dialog. Enter the coordinate values $x = -4.866$ and $y = 8.000$. For the final line, we shall set a local snap mode. Press the *F2* key to bring up the local-snap-mode dialog of Fig. 14. Change the mode by checking the *EndPoint* box and click *OK*. This snap mode will remain in effect until you exit the insertion mode or change the local mode by calling the dialog with the *F2* key. Finally, move the mouse cursor close to the endpoint of the vector at (-6.0,8.0) and click the left mouse button to complete the outline. Click the right mouse button to exit insertion mode. Note that the snap mode has been restored to grid snap.To complete the electrode definition,



No snap [✓ Grid] Endpoint Nearest point

Snap distance: 2.000E-01

Figure 14: Snap control dialog.

call up the region properties dialog. Set Region 4 to *Filled* and assign the name *PosElect*. If you switch to the *Display fill* mode, the program plots the electrode as a closed shape that over-writes portions of the three dielectric regions (Fig. 8).

4.5 Editing commands

In this section, we will use some editing options to help define the negative electrode on the right-hand side. Start a new region and then pick the rectangle tool. Make a box with corners at (1.0,0.0) and (6.0,3.0). Make another rectangle from (5.0,3.0) to (6.0,10.0). The result is a set of two adjacent rectangles that do not form a closed outline. First, we will change the shape of the lower rectangle, adding a radius on the upper-left corner. Pick the command *Edit/Fillet-Chamfer width*. In the dialog, enter the value 1.375 for the fillet radius. Click *OK* to exit the dialog and pick the command *Edit/Fillet*. Move the mouse cursor close to the midpoint of the vertical line from (1.0,0.0) to (1.0,3.0) and click the left button to highlight the vector. Then move the cursor close to the midpoint of the horizontal line from (1.0,3.0) to (6.0,3.0) and click the left button. The fillet operation automatically shortens the vectors and enters a connecting arc of the specified radius tangent to the vectors. Note that fillet operation may be applied to any intersecting line vectors, even if they are not at right angles.

To complete the figure, we need to remove two line segments: the lower side of the top rectangle and the a section of the top side of the bottom rectangle between (5.0,3.0) and (6.0,3.0). Click on the *Select vector* command, move the mouse cursor close to the midpoint of the line that is to be removed and click the left button.Mesh highlights the bottom side of the top rectangle. Click the right mouse button to exit the *Selection* mode, reactivating the other menu commands. Then, click the *Delete selection* command to remove the vector. Although the vector is gone, you will not see a difference in the plot because the top of the lower rectangle is still present. You may wonder why **Mesh** picked the shorter vector even though the two unwanted lines shared the same space. The answer is that the upper rectangle was the last object entered. When there is an ambiguity in identification by proximity, **Mesh** always picks the most recent entity.

To complete the shape, we need to shorten the vector along the top of the lower object. Pick the command *Edit/Trim vector*. To set the trimming vector, move the mouse cursor close to the left side of the top rectangle that extends from (5.0,3.0) to (5.0,10.0). The program highlights the trim line. Then move the mouse cursor close to the portion of the vector from (2.375,3.0) to (6.0,3.0) that should be removed. Figure 15 shows the completed outlines for the

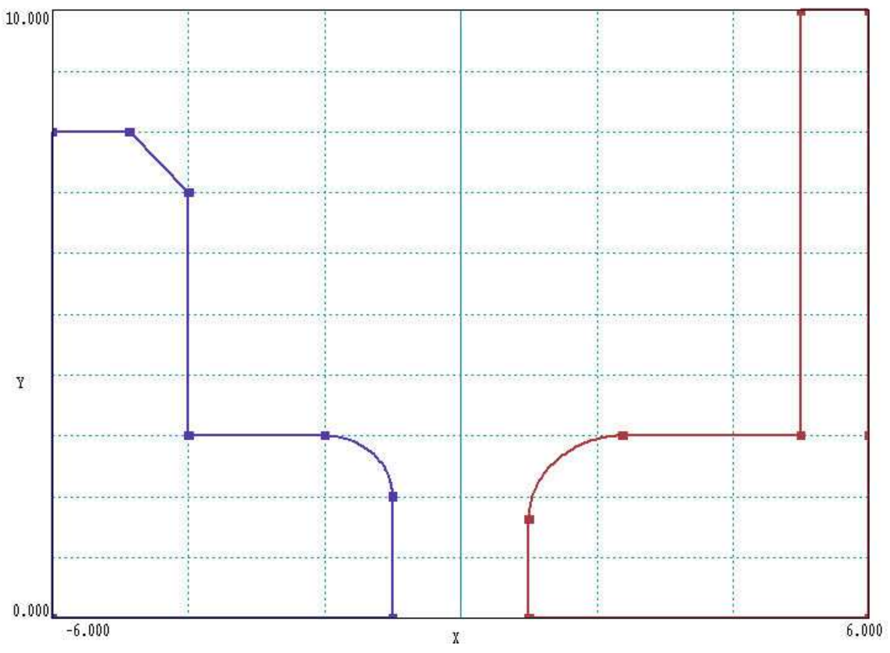


Figure 15: Completed outlines of the positive and negative electrodes.

two electrodes. Set the status of Region 5 to *Filled* and assign the name *NegElect*. Be sure to save your work before proceeding to the next stage.

4.6 Changing the view and snap settings

The final filled part is the trigger electrode. Until now, we have viewed the entire solution area. We will narrow the view and change snap settings to help create the small part. Choose *View/Zoom window* or use the tool. Enter two corners at approximately (-3.0,2.0) and (3.0,8.0) to define a view box. Note that the displayed grid intervals automatically adjust to 1.0 in z and r. Next pick *Settings/snap control*. Be sure that the mode is set to *Grid snap* and change the snap distance to 0.125. It would be helpful if the displayed grid points were the same as the snap grid points. Pick the command *Settings/Grid control*. Uncheck the *Automatic intervals* box and enter 0.125 in the boxes for *XGrid* and *YGrid*. Pick *Insert/Start next region* to advance to Region 6.

Create the shape shown in Fig. 16 using a series of lines. For the curved edge, you can use either *Insert/Arc/Start-End-Center* or the *Edit/Fillet* command. There are several options if you make a mistake:

- Remove a single vector or a series created in *Insert/Coordinate* mode with *Edit/Undo last operation*.
- Select several vector and remove them with the *Delete selected command* command. In this case, it is a good precaution to lock the nearby vectors of Region 2 using the region properties dialog.
- Remove all vectors of Region 6 using the *Select region* and *Delete selected* commands.

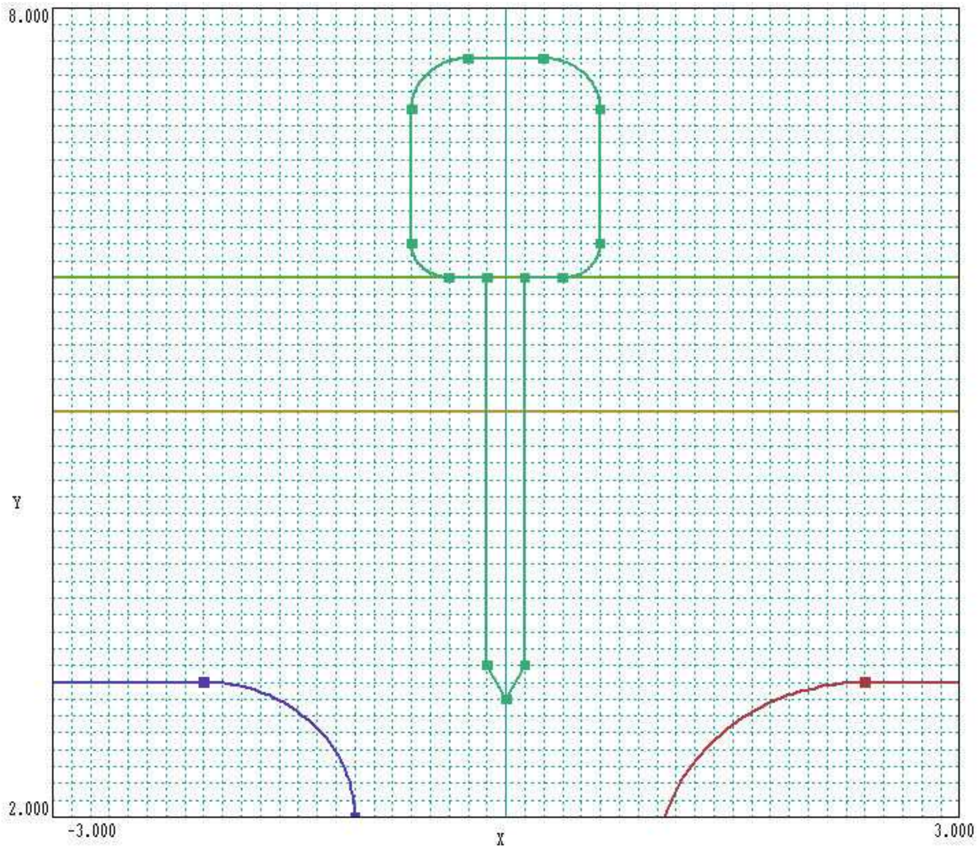


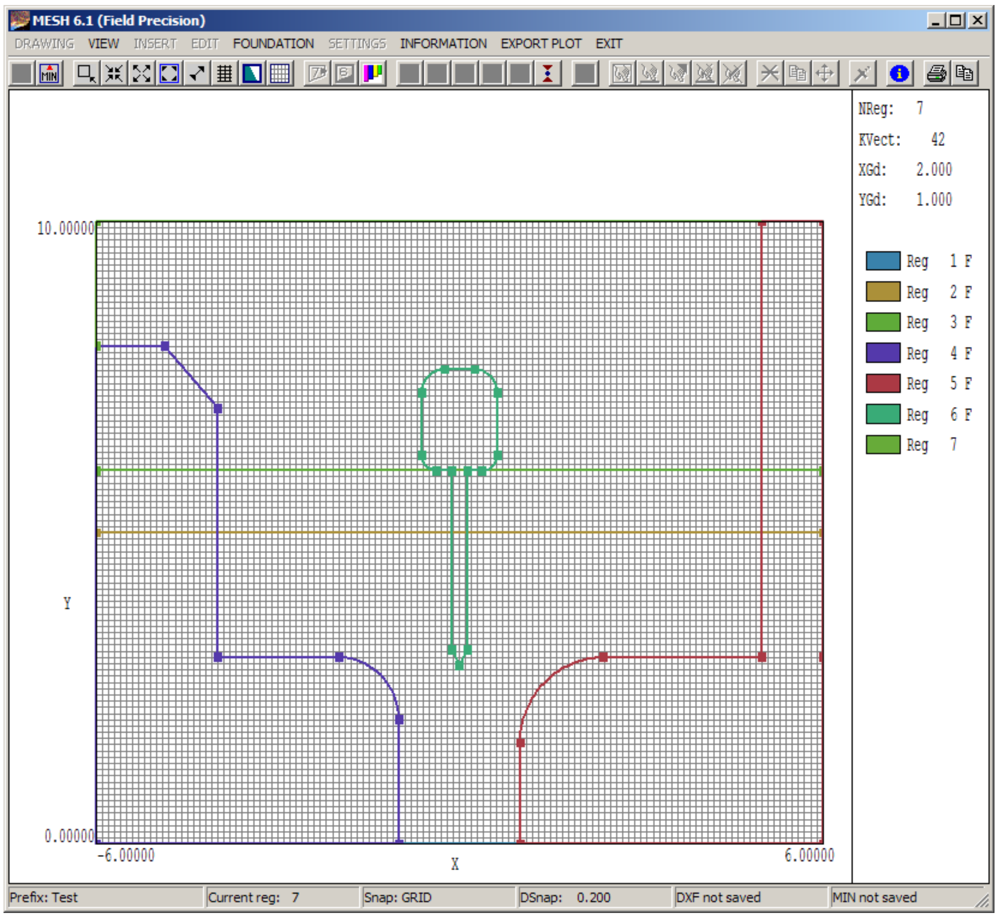
Figure 16: Drawing the trigger electrode with optimized view, snap and grid settings.

Use the *Settings/Region* properties dialog to activate the *Filled* attribute for Region 6 and to set the name as *Trigger*.

The final drawing entity is a line region (Region 7) on the left-hand boundary from (-6.0,8.0) to (-6.0,10.0). The nodes of this region will be assigned the same fixed potential as the positive electrode. Use *Settings/Grid control* to return to automatic intervals for the display grid. Use the *View/Global* view command to return the display to the full solution area. Advance to Region 7 with *Insert/Start next region* and use *Insert/Line* to create the vector. Assign the name *LeftBound* and do not check the *Filled* box. Save the completed drawing.

4.7 Refining the foundation mesh

Now that we have created all the outline vectors, we must consider the element sizes necessary to resolve the region shapes. To review, the *foundation mesh* consists of a baseline set of similar triangles that will be stetched to conform to the boundaries of the regions. The drawing editor contains a popup menu to set properties of the foundation mesh elements. Choose the *Foundation/Display foundation* menu command or tool to activate the display of Fig. 17. The gray lines show the approximate dimensions of the bases and heights of the triangular elements. Choosing the foundation display mode activates the foundation mesh commands and deactivates commands to create and to edit vectors. The easiest way to pick commands is to right-click the mouse to show the foundation-mesh floating menu.


MESH 6.1 (Field Precision)
DRAWING VIEW INSERT EDIT FOUNDATION SETTINGS INFORMATION EXPORT PLOT EXIT
10.00000
0.00000
-6.00000
6.00000
X
Y

```
NReg:   7
KVect:   42
XGd:    2.000
YGd:    1.000
```

🟦	Reg 1 F
🟨	Reg 2 F
🟩	Reg 3 F
🟪	Reg 4 F
🟥	Reg 5 F
🟩	Reg 6 F
🟩	Reg 7

```
Prefix: Test   Current reg: 7   Snap: GRID   DSnap: 0.200   DXF not saved   MIN not saved
```

Figure 17: Foundation mesh display with default resolution.

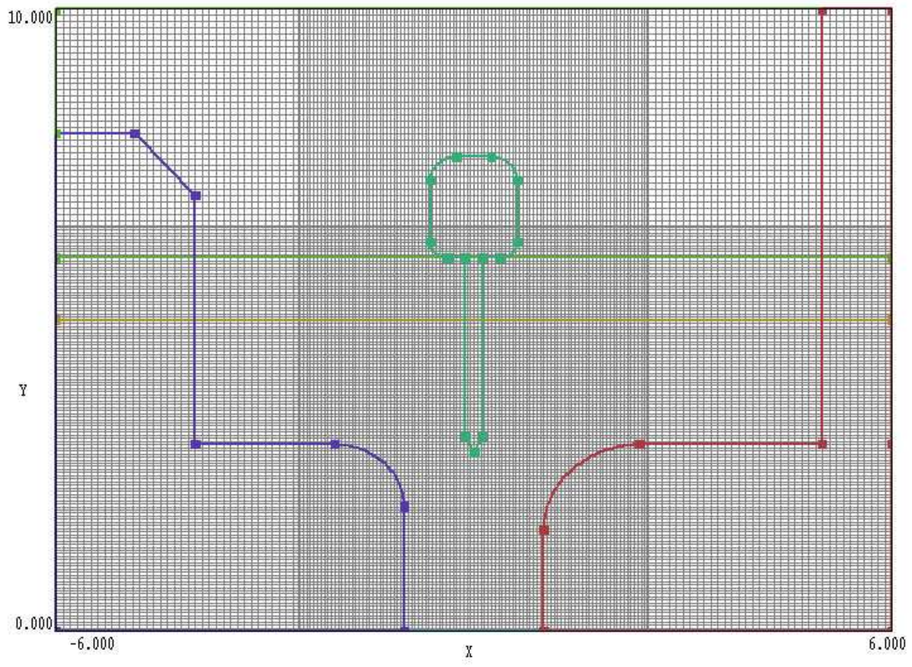


Figure 18: Modified foundation with multiple resolution zones.

Mesh has chosen a default resolution where the triagle base and height are approximately equal to 0.10. This element size is sufficient for most shapes in the drawing, but we want to use smaller elements to resolve details of the trigger electrode. We shall start by changing element sizes along the horizontal axis. The goal is to set three zones of resolution: the outer two zones have element base dimension 0.10, while the inner zone near the trigger electrode has base dimension 0.05. Click the right mouse button to bring up the foundation mesh floating menu and chose the command *XAxis/Divide zone*. Move the mouse cursor into the single horizontal zone that will be divided and click the left button. The entire zone is highlighted. Now move the cursor to the position $x = -2.5$ and click the left button. **Mesh** divides the horizontal space into two zones: $x > -2.5$ and $x < -2.5$. Again pick the *XAxis/Divide zone* command. Move the cursor into the right-hand zone and click the left button to identify it. Then move the cursor to $x = 2.5$ and click the left button. The horizontal space is divided into three zones with equal element base dimensions. To complete the modification, we need to change the element size in the central zone. Pick the command *XAxis/Element size*, move the cursor into the central zone, and click the left button. In the dialog, enter 0.05. Follow a similar procedure to divide the vertical space into two zones with element size 0.05 for $y < 6.5$ and 0.10 for $y > 6.5$. Figure 18 shows the final result.

4.8 Creating a script

At several times in this chapter, we have backed up drawing entities to the file `SPARKGAP.DXF`. To finish the project, we must save our work to a mesh script, `SPARKGAP.MIN`. It is important to recognize the functions and differences of the two file types:

- `DXF` files have a standardized but arcane text format recognized by all CAD programs. The `DXF` file created by the drawing editor does not contain information about the foundation mesh which is specific to the **Mesh** program.
- The `MIN` file has a text format recognized by **Mesh**. The format is simple and succinct comapared to `DXF` files – it is relatively easy to edit `MIN` files directly to make small changes. The `MIN` file contains all information necessary to create a mesh, including the foundation mesh specifications.

After setting details for the foundation mesh, we must generate a **Mesh** script to avoid loosing the work. You can use the *Drawing/Export MIN* command to record information on the present state of the drawing vectors and foundation mesh specifications. Use the default file prefix or enter a different prefix to save the file under a different name. **Mesh** inquires whether you want to save information when you exit the drawing editor. If you pick the *Exit/Save* option, the program records information in the file `SPARKGAP.MIN`.

When you return to the main menu, pick the *Edit script/Text* command. The program loads `SPARKGAP.MIN` into the internal text editor. Chapter 7 describes the file syntax. Even without a detailed knowledge, you will be able to recognize significant features in the file. The initial *Global* section contains the foundation mesh information including the three horizontal zones and two vertical zones of element resolution. Seven region sections follow. Each section contains the set of line or arc vectors that constitute the boundary outline. A data entry to define a line has the form

```
L   XStart  YStart  XEnd  YEnd
```

The data entry for an arc contains six values,

```
A   XStart  YStart  XEnd  YEnd  XCenter  YCenter
```

5 Drawing editor reference

5.1 Starting the drawing editor from the main menu

The example in the previous chapter introduced many features of the **Mesh** drawing editor. This section describes the full set of capabilities. We shall begin with commands in the main **Mesh** menu to call up the drawing editor, either for creating new scripts or editing existing ones. To review, a *drawing* contains a set of vectors (points, lines, arcs) that define the shapes of regions. A drawing can be saved in `DXF` format and exported to other CAD software. A *script* contains the drawing vectors in a simplified format recognized by **Mesh** as well as advanced information such as characteristics of the foundation mesh or advanced commands to load images. There are several options for conversions between the drawing and script formats.

FILE/CREATE SCRIPT/GRAPHICS

Use this command to start a new drawing. In the dialog, supply a file prefix (1-46 characters in length). The prefix will be applied by default to saved drawings (`FPrefix.DXF`) and scripts (`FPrefix.MIN`). The spatial limits in the dialog define the dimensions of the solution rectangle. The solution volume may or may not fill the rectangle.

FILE/EDIT SCRIPT GRAPHICS

This command is useful to make changes in an existing **Mesh** script or to create a modified script. The command is active if a script has been loaded into **Mesh** with the *File/Load script* command (Sect. 6.2). **Mesh** translates the script information (including the foundation mesh parameters) and opens the drawing editor.

FILE/CREATE SCRIPT/DXF IMPORT

The command serves two functions: 1) reload a **Mesh** drawing that has been saved in `DXF` format or 2) import a drawing from another CAD program. The program loads the information and starts the drawing editor. Specify whether coordinates should be displayed as x-y or r-z.

With regard to DXF import from other programs, **Mesh** recognizes entities of the types *Point*, *Line*, *Arc*, *Circle* and *Polyline*. It ignores all other entities such as text. The program splits circles into four arc vectors and splits polylines into individual line vectors. The program recognizes only the line portions of polylines and ignores arcs, splines and other complex shapes. The dimensions of the solution rectangle are determined from the maximum and minimum x-y (or z-r) values of valid entities. To ensure that entities are assigned to regions, you must create layers named '1', 2', '3', ... in your drawing. The maximum numbered layer name is '250'. **Mesh** assigns all valid entities in Layer 1 of the drawing to Region 1, entities in Layer 2 to Region 2, and so forth. Unrecognized entity types are ignored and all valid entities in other layers are assigned to the layout region (Region 0). (Note: In the past, AutoDesk and other companies have changed their `DXF` formats without notice. Therefore, we cannot guarantee that translations will be successful with future varions of CAD programs. **QCAD**, a good

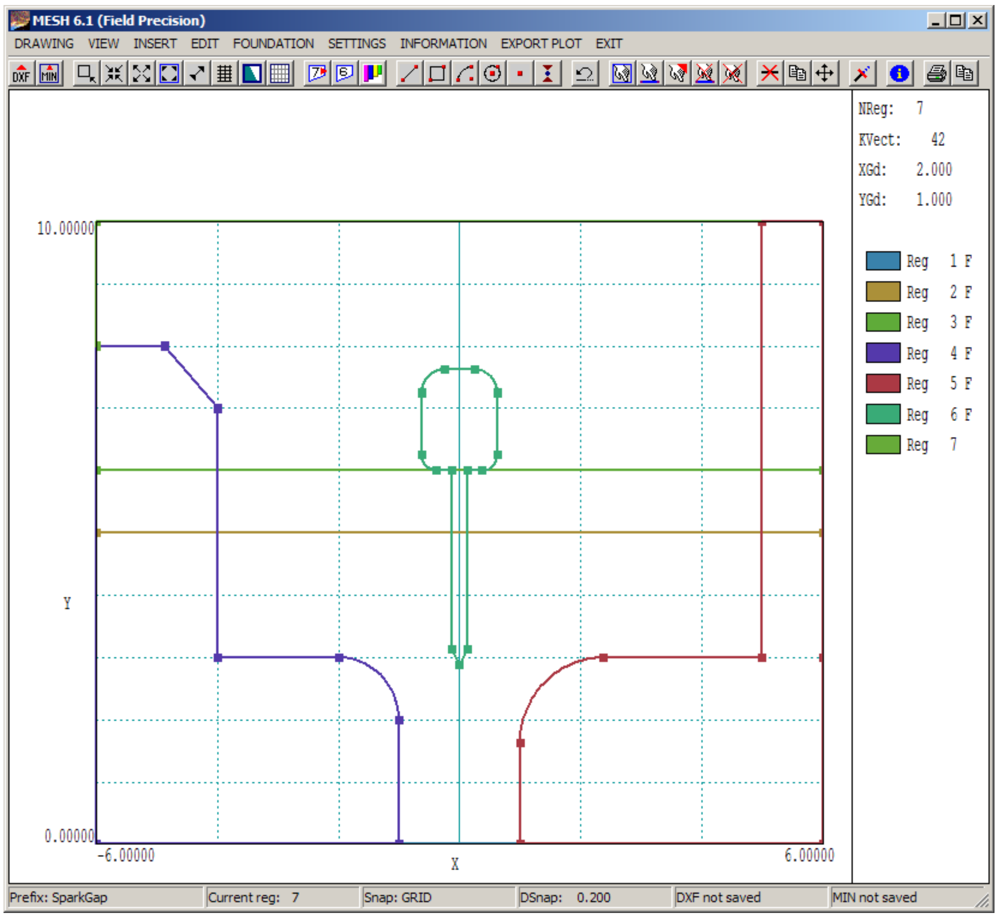


Figure 19: Working environment of the **Mesh** drawing editor.

two-dimensional freeware CAD package which is compatible with **Mesh** is available on our resource download site.)

5.2 Working environment

This section covers some general features of the drawing editor environment shown in Figure 19. All commands are accessible from the menu at the top. The toolbar contains entries for frequently-used commands. The function is displayed if you hold the mouse cursor over a tool. You can also bring up selected commands in a floating menu with the right mouse button. The main display is divided into two areas: the drawing area and the information area. The information area on the right-hand side lists the total number of regions, the total number of vectors and display-grid intervals in x and y (or z and r). The area also shows region color coding and the fill status of regions.

The status bar at the bottom of the screen operates in two modes. In the standard mode, the bar displays the prefix that will be assigned to saved `DXF` and `MIN` files, the current region for vector entry, the snap mode, the snap-grid spacing and the save status for `DXF` and `MIN` export. The status bar operates in the coordinate mode when the program expects mouse position information. Here, the bar displays brief instructions, x-y (or z-r) coordinates and the current snap mode. When the program is in coordinate mode, press the *F1* key to display a dialog where you can enter coordinate values directly from the keyboard. Press the *F2* key to change the snap mode.

Two types of grids apply to the drawing area: the *view grid* and the *snap grid*. The view grid is a series of horizontal and vertical lines superimposed on the drawing display. In the default mode, the program picks intervals in x and y that depend on the shape of the drawing and the zoomed view area. Values of the intervals are displayed in the information area. Use the *Settings/Grid control* command to set specific intervals. In this case, the view grid does not change with the zoom factor. The snap grid applies to the mouse *Grid snap* mode. The drawing is divided into an invisible set of lines with the same spacing (*DSnap*) in the horizontal and vertical directions. In coordinate mode, the magnetic snap point shows the grid location closest to the mouse cursor. Note that the view and snap grids are not necessarily the same unless you set parameters manually.

There are four snap modes that may apply in commands that require coordinate input: *No snap*, *Grid snap*, *Endpoint snap* and *Nearest snap*. The *No snap* mode is automatically invoked for operations that involve changing the view window or locating the object closest to the mouse cursor (*Settings/Region information*, *Edit/Delete vector*,...). You should avoid this mode when you are defining line and arc vectors that constitute the boundary of a filled region because the endpoints must match exactly. In the *Grid snap* mode, coordinates returned by the mouse are tied to points of the snap grid. In *Endpoint snap* mode, the program returns coordinates of the vector endpoint closest to the mouse cursor. This mode is useful to connect to a vector that does not terminate at a snap grid position. In the *Nearest snap* mode, the returned coordinates are on the nearest point on the closest line or arc vector. This mode is useful to define a vector that connects exactly to an existing one. In the three snap modes, **Mesh** displays a magnetic snap point (red square) in addition to the present mouse cursor position (black cross). The magnetic snap point shows the location of the coordinates that will be returned if you click the left mouse button.

Drawings may be displayed in two modes: *vector* and *fill*. In the vector mode **Mesh** displays the lines, arc and points that constitute region boundaries. It is the required display mode for inserting and editing drawing entities. In the fill mode, **Mesh** colors the areas outlined by filled regions. Most commands for entering and editing vectors are not active in this mode. The fill mode serves the following functions:

- Creating illustrations for presentations and papers.
- Checking that the region order correctly implements overwriting. For example, in Fig. 12 note how the electrodes over-write the space shared with the gas, ceramic and oil regions.
- Confirming that the vectors of an outline are connected and to form a closed boundary.

You can enter vectors in any order when defining a filled region. **Mesh** sorts and orders vectors as they are added so that matched endpoints connect in the vector list. In fact, you can enter part of a region, work on other entities, and then return to complete the region later. The one limitation is that the region display in the fill mode will not be correct until the complete outline has been defined.

Several vector operations (such as *Insert/Line*, *Edit/Delete/Vector*,...) allow multiple entries without retyping the command. When these commands are issued, **Mesh** enters the *Insert/Coordinate* mode where most of the menu commands are deactivated. You can enter defining points for several vectors. When you want to use a different command, press the right mouse button to exit the mode. As an example, suppose that *DSnap* = 0.250 and that we want to create the following two line vectors:

```
L  1.500  0.750  1.600  0.893
L  1.600  0.893  2.000  1.000
```

Choose the command *Insert/Line*. Then move the mouse to a point near (1.500,0.750) and press the left button. Press the *F1* key, type in the values $x = 1.600$ and $y = 0.893$, and click *OK* to complete the first vector. The program is ready for the start point of the second vector. Press the *F2* key, activate the *Endpoint snap* radio button and click *OK*. Move the mouse cursor near the second point of the first vector and click the left button. Again press the *F2* key and return to *Grid snap* mode. Move the mouse cursor close to the point (2.000,1.000) and click the left button. Finally, click the right button to exit *Insert/Coordinate* mode. Note that when you change the snap mode using the *F2* key during any insertion operation, the change applies only during the operation. The snap mode returns to the previous setting when the operation is complete.

There are three options when you exit the drawing editor:

- **Save**: create or overwrite a **Mesh** script with the name `FPrefix.MIN`, where *FPrefix* is the current file prefix.
- **Save as**: enter a new file prefix to name the script. This option is useful if you make changes but do not want to loose the original MIN file.
- **Abandon**: leave the drawing editor without creating or over-writing a script. If you have saved a DXF file, you can recover the drawing entities and create a **Mesh** script later. Note that any special settings of the foundation mesh will be lost.

Before performing any insert or edit operations, the drawing editor copies present entities to a file `TEMP.DXF`. This file is used for the *Edit/Undo last operation* command. You can load it with the *Create script/DXF* import command to recover lost work.

5.3 Drawing menu

The commands in the *Drawing* popup menu export and import information. Important commands may also be invoked by an item in the toolbar or from the floating menu (right click).

EXPORT DXF

Record the drawing entities in a file `FPrefix.DXF`. Use the current file prefix or supply an alternate in the dialog. The file follows a standard format that can be read by most CAD programs. Note that information about the foundation mesh is not included in the file.

IMPORT DXF ENTITIES

Add valid drawing entities (lines, polylines, circles,...) contained in a DXF file to the current drawing. In contrast to the command *File/Create script/DXF input* of the main menu, this command does not change the limits of the solution rectangle. You must ensure that all objects fit within the limits – the program does not perform automatic clippping. To avoid corrupting existing regions, all entities are added to the layout region (Region 0). Use the *Select* and *Transfer selection to current region* commands to assign entities to appropriate regions.

EXPORT MIN

Write a **Mesh** script with complete information on drawing entities and the foundation mesh. You also have the option to save the script when you exit the drawing editor.

CHANGE DRAWING LIMITS

Expand or contract the limits of the solution volume. In the latter case, all existing vector must fit inside the modified area. **Mesh** erases all foundation mesh specifications and sets default values.

DIMENSION LIST

Create a list of drawing-vector dimensions to help in fabricating a design. The file is saved under the same `FPrefix.LST`.

5.4 View menu

The commands in this menu change the limits of screen views and hardcopy plots.

ZOOM WINDOW

Zoom in on an area by defining two points of a box with mouse or keyboard entries. When the command is issued, the coordinate mode becomes active and the snap mode is temporarily set to *No snap*.

ZOOM IN

Narrow the view around the current center of the plot.

EXPAND VIEW

Expand the view around the current center of the plot.

GLOBAL VIEW

Expand the view to show the full solution volume rectangle.

PAN

Move the current center of the plot by defining two points of a displacement vector with the mouse or keyboard entries.

FILL DISPLAY
Switch between vector and fill display modes. In the fill mode, the program uses an algorithm that sets pixels inside the closed polygons of regions that you have designated as *Filled*. Open regions appear only as outlines. Filled regions will exhibit distortions if their vectors do not form a connected, closed set. If a filled region does not appear, it may have been over-written by another region. In this case, change the order of the region list using the *Settings/Region properties* command.

5.5 Insert menu

The commands in this group are used to create drawing vectors and to organize them into regions. The following two commands determine the current region for vector entry.

START NEXT REGION
Drawing entities are entered in the current region. The program sets the current region to Region 1 when starting a new script, or to the highest region when editing an existing script. This command adds a region to the drawing and sets it as the current region. If you forgot to use this command before adding vectors, you can change the region associations of vectors later.

SET CURRENT REGION
Go to an existing region to add vectors. If the specified region is undefined, Mesh sets the current region set to the maximum region number.

The next set of commands is used to add drawing entities: lines, arcs and points.

ADD LINE(S)
Create one or more line vectors in the current region. The program enters *Insert/Coordinate mode* where you can enter the start and end points of any number of lines. Press the right button to exit the mode. Press *F1* to enter coordinates from the keyboard, press *F2* to make a temporary change in the snap mode.

ADD RECTANGLE
Create a rectangle (four connected line vectors). Use the mouse or keyboard to set two corners of a box to define the rectangle.

ARC/START-END-CENTER
Add one or more arcs by using the mouse or keyboard to set the start, end and center points. The program gives an error message if the points are inconsistent. Arcs may be defined with a sense of positive or negative rotation – **Mesh** orders arcs in a filled region to produce a connected set of vectors. Note that you can change the snap mode at any time in the entry of the three points.

ARC/START-END-RADIUS
Define the start and end points with the mouse or keyboard and then type the radius value in the dialog. The second entry in the dialog gives the relative position of the arc center (up/right or down/left) to determine the sense of rotation.

ARC/START-CENTER-ANGLE
Use the mouse or keyboard to define the start and center points of an arc. Enter a value for the angle in the dialog in degrees. A positive value for the angle gives an arc that points in the counterclockwise direction.

CIRCLE/CENTER-POINT
Use the mouse or keyboard to define the center point and a point on the circle. Mesh creates four 90^o arc vectors.

CIRCLE/CENTER-RADIUS
Use the mouse or keyboard to define the center point of a circle and type a value for the radius in the dialog. Mesh creates four 90^o arc vectors.

POINT
Add one or more points using the mouse or keyboard. Points may be included only in open regions. Typically, points are used to represent arrays of thin wires.

5.6 Edit menu

The commands of the *Edit* menu are used to modify the dimensions or characteristics of drawing vectors.

UNDO LAST OPERATION
Reverse the last operation by reloading the file `TEMP.DXF`. Note that multiple vectors added in the *Insert/Coordinate* mode will be removed.

Many edit operations are performed on a selected vector or set of vectors. You can use multiple select or deselect operations to define the set before performing an edit operation. Note that vectors in locked regions or in regions that are not visible will not be selected.

SELECT WINDOW
Use this command to select all vectors whose endpoints lie within a rectangular region. Using the mouse, left-click to define one corner of the rectangle and then left-click again to define the other corner. Selected vectors are highlighted.

SELECT VECTOR(S)
Select one or more vectors by moving the mouse close to the vector and left-clicking. Click the right mouse button to exit insert mode.

SELECT REGION
Select all vectors in a region by moving the mouse close to any vector in the region and left-clicking.

DESELECT VECTOR(S)
Remove one or more vectors from the selection set by moving the mouse close to the vector and left-clicking. Click the right mouse button to exit the operation.

DESELECT ALL
Remove all vectors from the selection set.

DELETE SELECTION
Erase all vectors in the selection set. To prevent erasing coincident vectors in a region, use the *Settings/Region properties* command and lock the region before performing selection operations. As in all edit operations, **Mesh** sorts the vector set in order of regions and arranges the vectors of each region to match endpoints.

COPY SELECTION
Reproduce the selected vectors. The new vectors have the same position and region number. The new vectors remain selected, so you can move them or transfer them to a different region.

MOVE SELECTION
Move the selected vectors. **Mesh** prompts for a start and end point to define a displacement. As in any coordinate mode operation, you can enter values from the keyboard by pressing the *F1* key or change the snap mode by pressing the *F2* key.

ROTATE SELECTION
Use the mouse or keyboard to supply an origin point for the rotation. Then rnter the rotation angle in degrees (positive for a counterclockwise rotation).

With regard to the *Move selection* and *Rotate selection* commands, it is important to remember that the drawing editor does not perform automatic clipping. You must ensure that vectors of shifted objects are contained within the solution rectangle. If a script contains invalid vectors, **Mesh** will issue an error during execution of the *Process* command in the main menu.

TRANSFER SELECTION TO CURRENT REGION
Change the identity of vectors in the selected set to the current region. As an example, suppose you want to copy an object from one layer to the current. You would first use select and deselect commands to identify the object, then use the Copy selection command to replicate them and then this command to move them to the current region.

SPLIT VECTOR(S): MIDPOINT
Move the mouse close to a line or arc vector and click the left button. The vector will be split into two vectors at the midpoint.

SPLIT VECTOR(S): CLOSEST
Move the mouse close to a line or arc vector and click the left button. The vector will be split into two vectors at a point closest to the cursor location.

TRIM VECTORS
Terminate a line or arc vector at a trim line. You can add a trim line to the layout region or use an existing line vector. Identify the trim line by moving the mouse cursor nearby and clicking the left button. Then pick one or more vectors to trim. Move the mouse cursor close to endpoint of the vector **on the side that should be eliminated** and then click the left button.

FILLET
Smooth the edge between two connected line vectors by joining them with an arc of a given radius. Use the *Fillet/chamfer width* command to set the radius. Pick the two vectors with the mouse. The resulting arc has the specified radius and is tangent to the two lines. The program reports on error if it is impossible to satisfy this condition.

CHAMFER
Smooth the edge between two connected lines by joining them with a bevel of specified length. Use the *Fillet/chamfer width* command to set the bevel length. Pick the two vectors with the mouse.

FILLET/CHAMFER WIDTH
Set the current fillet radius or chamfer width by typing values in the dialog.

5.7 Foundation menu

The commands of this popup menu are used to modify the foundation mesh. The information is included in a mesh script (`MIN`) but is not saved in a drawing file (`DXF`). Most of the commands are inactive if the foundation mesh is not displayed.

FOUNDATION DISPLAY
Toggle the display of the foundation mesh. Vertical gray lines give the approximate base width of the triangular elements, while horizontal lines give the approximate height. When the foundation mesh is visible, foundation commands are active but vector editing commands are inactive.

The following commands may be applied to either the x/z axis (horizontal) or y/r axis (vertical).

DIVIDE ZONE

When you start a new script, **Mesh** creates a default foundation mesh. It contains single resolution zones along x and y with about 100-150 elements along each axis. It is often useful to divide the axes into zones with different element sizes to resolve small details. Use this command to divide an existing zone into two zones. Move the mouse cursor into the target zone and click left botton. Elements of the chosen zone will be highlighted. Then move the cursor to the division location and again click the left button. **Mesh** updates the plot with a thick line to show the boundary between the two zones.

SHIFT ZONE BOUNDARY

Adjust the position of an existing boundary between zones. Move the mouse cursor close to the target boundary and click the left botton. Then move the mouse to a new location within the adjacent zones and again click the left button.

CHANGE ELEMENT SIZE

Change the element size within an existing zone. Move the mouse cursor into the target zone and click left button. The dialog shows the present element base or height in the zone. Type in a new value and click *OK*.

UNDO LAST CHANGE

Reverse the last foundation mesh operation.

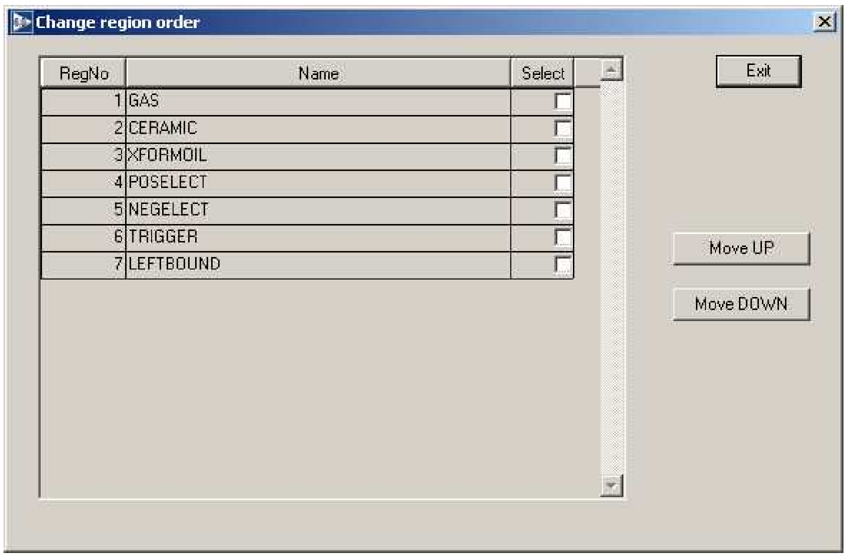
5.8 Settings menu

REGION PROPERTIES

This command brings up dialog shown in Fig. 11 to set the properties of regions. There are three columns with check boxes for each region: *Visible*, *Locked* and *Filled*. The *Visible* attribute determines if the vectors of the region are displayed in the drawing editor. Regions that are not visible will still be recorded in MIN and DXF files. Set the *Locked* attribute to prevent changes in the vectors of a layer and also to ensure that the vectors are not picked in any mouse selection operation. Finally, the *Filled* attribute indicates that a region will be marked with the designator *Fill* in the MIN file. The condition corresponds to a closed boundary with internal elements set to the region number. By default, Region 1 is marked as *Filled* and all other regions are initially *Open*. The next column shows the number of line, arc and point vectors in the region. The final column contains a descriptive name. Click inside the box and supply text. The name may not contain any of the **Mesh** standard delimiters including spaces. For example, the string `TRIGGER_PIN` is valid but `TRIGGER PIN` is not.

REGION ORDER

As described in Sect. 3.3, the order of regions in the **Mesh** script determines how shared areas are overwritten. The order may have important effects on physical solutions derived from the mesh. This command brings up dialog of Fig. 20 to modify the order in which regions will be



RegNo	Name	Select
1	GAS	☐
2	CERAMIC	☐
3	XFORMOIL	☐
4	POSELECT	☐
5	NEGELECT	☐
6	TRIGGER	☐
7	LEFTBOUND	☐

Figure 20: Dialog to modify the order of regions in the script.

recorded in the script. Check the box next to the region that should be shifted. Then click the *Move Up* or *Move down* buttons one or more times to move the selected region relative to the others. To delete a region, check its box and click on the *Delete region* button.

GRID CONTROL

This command brings up a dialog to change the settings of the view grid (these settings do not affect the snap grid). Check or uncheck the box to toggle the grid display. When the *Automatic intervals* box is active, **Mesh** picks convenient intervals between grid lines, depending on the size of the solution volume and the zoom factor. When deactivated, you can type in specific values for the horiztonal and vertical intervals. In this case, the intervals do not change with the display zoom factor. Setting $XGrid = YGrid = DSnap$ may be help to define entities in *Grid snap* mode.

SNAP CONTROL

The command brings up the dialog shown in Fig. 14 where you can choose the mouse snap type. Under the *No snap* option, the program returns coordinate values closest to the current mouse position. Under *Grid snap* option the program returns the closest position to a square mesh with nodes separated by distance *DSnap*. When this mode is active, you can type a values in the box to set *DSnap*. Finally, in the *Endpoint snap* mode the program returns the position of the vector endpoint closest to the cursor position.

TOGGLE MAGNETIC SNAP DISPLAY

Hide or display the magnetic snap point.

TOGGLE NUMBER FORMAT

Mesh determines whether to use scientific notation in the display of coordinates from the dimensions of the solution volume. Use this command to toggle the number display format.

5.9 Information menu

VECTOR INFORMATION
Show the properties of a drawing vector, including type, region number and endpoint coordinates. Move the mouse cursor close to the target vector and click the left button.

REGION INFORMATION
Show the parameters of a region, including type, number of vectors and fill status. Move the mouse cursor close to a vector of the target region and click the left button.

5.10 Export plot menu

All screen displays created by **Mesh** and the **TriComp** solution programs may be exported to hardcopy devices and plot files using the following commands:

DEFAULT PRINTER
This command sends a copy of the current plot to the default Windows printer. If you have several printers, use the *Settings/Printers* option on the Windows Start Menu to make changes in the default before running **Mesh**.

SAVE PLOT FILE
Use this command to create a graphics file of the current plot in either Windows Bitmap (BMP) or Portable Network Graphics (PNG) formats. In the dialog, specify the format, the size in pixels and the file prefix. The graphics file is created in the current directory.

COPY TO CLIPBOARD
Copy the current plot to the Windows clipboard (in Windows Metafile format) where you can paste it into other applications.

6 Creating meshes from scripts

6.1 Processing the mesh

Previous chapters showed how to create **Mesh** *scripts* with the drawing editor. A script is a text file that contains sets of vectors that define the outlines of objects in the solution space. This chapter describes how to convert the script to a mesh. The term *mesh* refers to the list of node coordinates and element identifies that will be used for the finite-element solution. They are recorded in a text file with name of the form `FPrefix.MOU`.

To begin, we shall review the procedure using the `SparkGap` example we created with the drawing editor (Chap.4). If you have not closed the program and have exited the drawing editor with the *Exit/Save* command, the script will already be loaded and ready for processing. Otherwise, run **Mesh** from **TC** and choose the command *File/Load/Script* in the main menu. In the dialog, pick `SparkGap.MIN`. Next, click the *Process* command or tool. The program automatically analyzes the contents of the script and creates a mesh. The status bar gives the processing status. If there are no errors, the *Plot/repair* command and tool are active. If an error occurs, temporarily use the copy of `SparkGap.MIN` supplied in the example library. We shall discuss methods to correct errors in latter chapters.

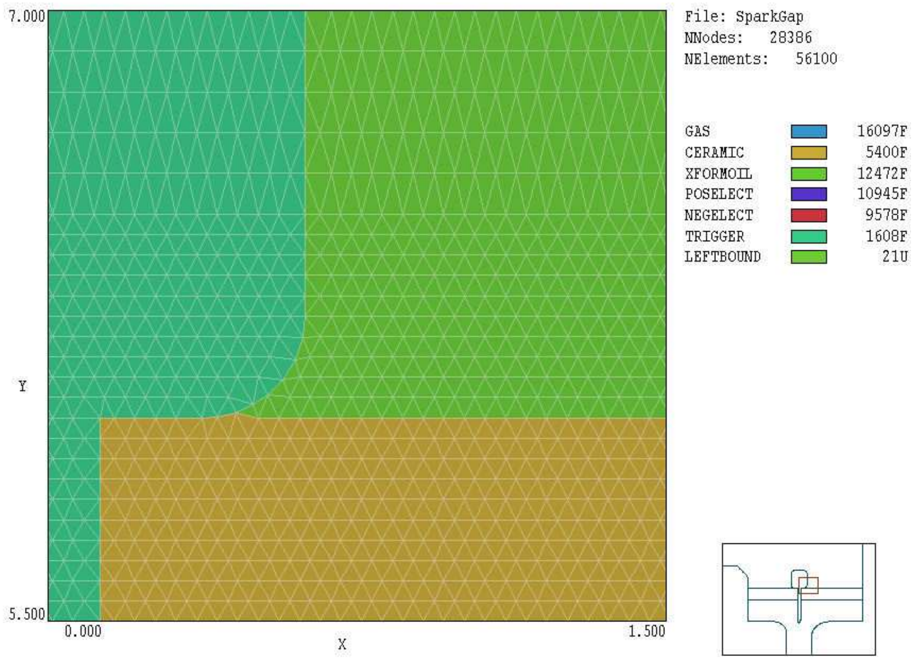


Figure 21: *Plot/Repair* menu – zoomed view of the `SparkGap` example.

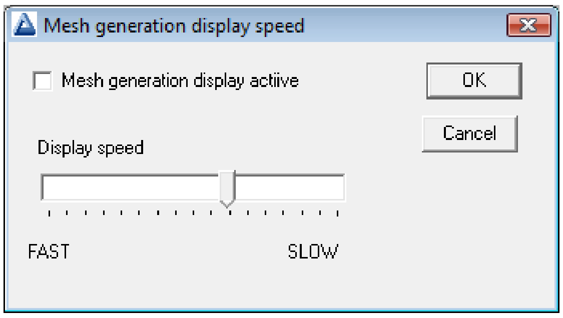

Mesh generation display speed
Mesh generation display actiive
OK
Cancel
Display speed
FAST
SLOW

Figure 22: Dialog to activate solution display and to set the refresh speed.

You can view operations performed by **Mesh** as they occur. Choose the command File/Mesh generation display to bring up the dialog of Fig. 22. Check the box to activate the display. You can also adjust the refresh speed to match your computer. Note that these settings are restored the next time you run **Mesh**. As the program proceeds through each region in turn, the display shows the nodes and element facets that are shifted to conformalize the boundary in red. After **Mesh** has identified enclosed elements of the region, the program shows the final boundary and element assignment and then proceeds to the next region. The active display is useful to check the ordering of regions and to locate the problem area when a vector is encountered.

Choose the command *File/View listing file*. **Mesh** loads `SparkGap.MLS` into the internal editor. This file contains detailed processing information that may be useful to diagnose problems. Exit the editor and click on the *Plot/Repair* command or tool. Figure 21 shows the working environment of the **Mesh** plot menu with a magnified view of a portion of the trigger electrode. The region names assigned in the drawing have been preserved. Note how the conformal elements created by **Mesh** follow the boundaries accurately.

6.2 File and help commands

There are two ways to run **Mesh**: 1) as an interactive program in a window and 2) as an autonomous program operating in the background. Initially, we shall concentrate on the interactive mode – Sect. 6.5 describes the autonomous mode which is used to process large amounts of data under batch file control.

If you run **Mesh** from the **TC** program launcher, the program starts in the same location and with the same screen size as in the previous session. The screen is blank and several commands are inactive. The following commands of the *File* popup menu are active in the initial state:

CREATE SCRIPT/GRAPHICS

Start the drawing editor to create a script. The process was described in Sect. 4.1.

CREATE SCRIPT/DXF IMPORT

Read a DXF file created with the drawing editor or another CAD program to define drawing entities and start the drawing editor. Section 5.1 described allowed entities and the conventions for assigning them to regions. The boundaries of the solution rectangle are determined by the maximum extents of the drawing entities.

CREATE SCRIPT/TEXT

Create a mesh script directly in text format. Chapter 7 gives a detailed description of the script format. The program displays the same dialog used for the graphics option. Enter a file prefix and the limits of the solution rectangle. The internal text editor starts with template content to remind you of the options. The material includes baseline *Global* and *Region* sections with available commands listed as comment lines.

LOAD/SCRIPT (MIN)

Pick an existing input script for processing and plotting. The dialog displays a list of files with names of the form `FPrefix.MIN`. Changing directories in the dialog will also change the working directory of the program. In this case output files will be written to the new directory. The status bar lists the name of the current script.

LOAD/MESH (MOU)

Load an existing mesh file for plotting and minor repairs (Sect. 6.4). Note that the shapes of objects in a processed mesh are fixed and cannot be changed within **Mesh**.

EDIT FILE

Use the internal editor to view or to modify any text file. Changing directories in the dialog does not change the working directory of the program.

The following commands are active when a script has been loaded.

EDIT SCRIPT/GRAPHICS

Translate the `MIN` file to a drawing and start the drawing editor. You can make changes within the editor and save the script with the same or a different name.

EDIT SCRIPT/TEXT

Load the script into the internal editor where you can make direct changes or additions. Be sure to save your work under the same or different file name before exiting the editor.

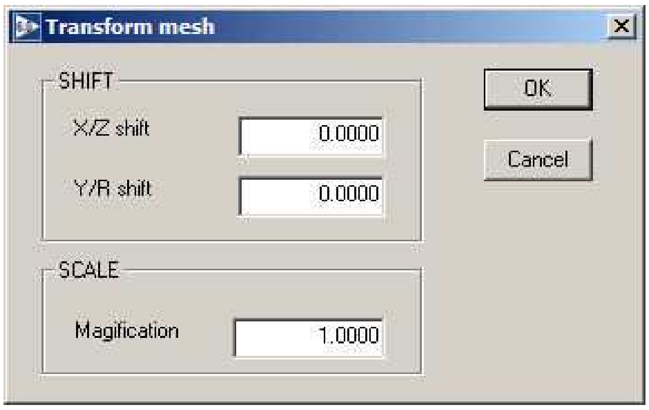
TRANSFORM MESH

Use this command to modify the current script to move the mesh in space or to change dimensions (rescale the mesh). **Mesh** displays the dialog shown in Fig. 23. Enter values x_{shift} and y_{shift} (or z_{shift} and r_{shift}) to move the mesh and/or a value M to scale the mesh. Coordinates are changed according to:

$$x_{new} = M x_{old} + x_{shift}, \tag{3}$$

$$y_{new} = M y_{old} + y_{shift}. \tag{4}$$

When you exit the dialog, **Mesh** prompts for a file prefix for the saved file.


Transform mesh

SHIFT

X/Z shift 0.0000

Y/R shift 0.0000

SCALE

Magification 1.0000

Figure 23: Transform mesh dialog.

MESH GENERATION DISPLAY

In the display mode, you can watch mesh generation operations are they are performed. The command calls up the dialog of Fig. 22. Check the box for an active display and use the slider to set the refresh speed. Note that mesh generation will be considerably slower. Also, no display is created when **Mesh** is called directly from the command prompt (Sect. 6.5).

PROCESS

The command has its own entry on the main menu. It initiates the mesh generation procedure. **Mesh** analyzes the input script, generates a foundation mesh and fits region boundaries. Information on the operation is recorded in the current listing file (`FPrefix.MLS`). The status bar reports if the operation has been successful.

The next two commands are active if a mesh script has been processed.

SAVE MESH (MOU)

Save the current mesh in the format described in Sect. 8.6. The output file has a name of the form `FPrefix.MOU`. The command is active only if processing has been successful. The program will prompt to save the current mesh if you leave the program or load a new script.

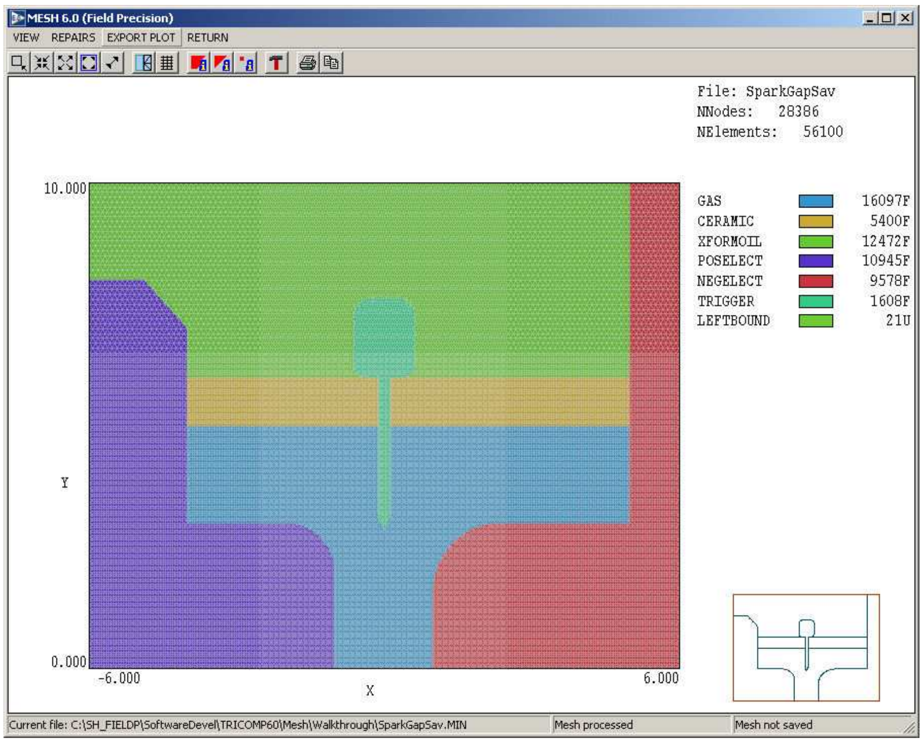
VIEW LISTING FILE (MLS)

Open an editor to view the current listing file. If an error condition exists, the cursor is located at the corresponding position in the file.

Finally, the *Help* menu contains two commands.

MESH MANUAL

view the chapters in this manual that pertain to Mesh in your default PDF browser. The file `mesh.pdf` must be in the same directory as `mesh.exe`.



```
File: SparkGapSav
NNodes:   28386
NElements:   56100
```

GAS	🟦	16097F
CERAMIC	🟨	5400F
XFORMOIL	🟩	12472F
POSELECT	🟪	10945F
NEGELECT	🟥	9578F
TRIGGER	🟩	1608F
LEFTBOUND	🟩	21U

```
Current file: C:\SH_FIELDP\SoftwareDevel\TRICOMP60\Mesh\Walkthrough\SparkGapSav.MIN    Mesh processed    Mesh not saved
```

Figure 24: Working environment of the *Plot/Repair menu.*

CLEAN DIRECTORY/PARTIAL
CLEAN DIRECTORY/FULL

A large number of data files may accumulate in an extended session. This command deletes unneccessary files from the working directory. Under the *Partial* option, **Mesh** erases `TEMP.DXF` and listing files from all programs (`*.?LS`). Under the *Full* option, the program deletes the above files as well as output files from **Mesh** and **TriComp** solution programs (`*.?OU`). The operation does not delete input files that are necessary to reconstruct the solution.

6.3 Plot/repair menu – mesh display

The *Plot/repair* command in the main menu is active when a mesh has been successfully processed. This command brings up the menu, toolbar and plot area shown in Fig. 24. The plot area is divided into three sections:

- **Main plot**. This section shows a scaled plot of the mesh.
- **Legend**. The section on the right-hand side of screen shows the number of nodes and elements in the mesh and a color-coded list of regions. Numerical information on far right side shows the number of elements for filled regions (F) and the number of nodes for unfilled regions (U).
- **Orientation**. The thumbnail sketch on the bottom right shows the boundaries of regions and the current zoomed view window.

There are three popup menus: *View*, *Repairs* and *Export plot*. Commands in the *Export plot* menu have the same function as in the drawing editor (Sect. 5.10). Similarly, the functions of the following commands to control the display area were described in Sect. 5.4:

ZOOM WINDOW
ZOOM IN
EXPAND VIEW
GLOBAL VIEW
PAN
GRID CONTROL
TOGGLE NUMBER FORMAT

The mouse can be used to set the view window and plays an important role in commands of the *Repair* menu. In contrast to the drawing editor, the mouse operates in either the *No snap* or *Grid snap* mode. Mouse operations are controlled by the following commands:

MOUSE CONTROL/TOGGLE MOUSE/KEYBOARD
MOUSE CONTROL/TOGGLE SNAP MODE
MOUSE CONTROL/SET DSNAP
The *Toggle mouse/keyboard* option determines whether coordinate input is from the mouse or keyboard. In the mouse mode, you can enter keyboard values for a point by pressing the *F1* key. *Toggle snap mode* switches between *No snap* and *Grid snap*. You can also toggle between the modes by pressing the *F2* key during coordinate input. As in the drawing editor, the quantity *DSnap* is the interval for the square snap grid.

PLOT TYPE/ELEMENTS
PLOT TYPE/NODE
PLOT TYPE/FACETS
The *Element* plot type shows elements color-coded by region number with the option to display boundary facets. The plot is useful for checking whether numbers have been correctly assigned to *Filled* regions. The *Node* plot shows nodes color-coded by region number. This plot is useful for checking whether region numbers have been correctly assigned along shared boundaries. Finally, the *Facet* plot is useful to check element shapes.

DISPLAY FACETS
The command determines whether element boundary facets are included in the *Element* type plot.

6.4 Plot/repair menu – repairs

There are four types of errors that may occur during mesh processing:

1. Script syntax errors, including non-matching intervals in the *XMesh* and *YMesh* commands and unconnected boundaries of *Filled* regions.
2. Failure to find a boundary path between the start and end points of a line or arc vector.
3. Inverted elements.
4. Incorrect region number assignment to nodes or regions.

The first two types of errors are fatal and halt mesh processing. Usually, errors of the first type can be corrected quickly using the *Edit script/Text* command. The editor initializes with the cursor on the line where the error condition was detected. The second error type occurs when the foundation mesh is ill-suited to the lines or arcs that constitute one or more region boundaries. The third or fourth types of errors can be corrected by changing the input script or by directly modifying nodes and elements in the processed mesh using commands of the *Repairs* menu. The first option is preferable because the changes will be implemented in any regeneration of the mesh. Direct mesh modification should be used sparingly in difficult situations.

The following commands display information on the processed mesh:

INFORMATION/NODES

Move the mouse cursor close to a node and click the left button. The identified node is highlighted and the following information appears in a dialog: coordinates (x, y), indices (k, l), region number and region name. Click *OK* to continue.

INFORMATION/ELEMENT

Move the mouse cursor into an element and click the left button. The identified element is highlighted and the following information appears in a dialog: center-of-mass coordinates (x, y), area, region number and region number.

INFORMATION/REGION

Move the mouse cursor into a region and click the left button. The identified region is highlighted and the following information appears in a dialog: region number, status (filled or open), total area, and region name.

The following commands are useful when there are a few stubborn errors. Be sure to save the mesh after correction and to preserve the file `FPrefix.MOU` for future solutions. The first three commands correct inverted elements.

RELAX BAD NODES

Mesh displays an error message if there are inverted elements and marks the associated nodes in red in the *Element* and *Facet* plot modes (Fig. 25, top). Use the *Element* mode and zoom

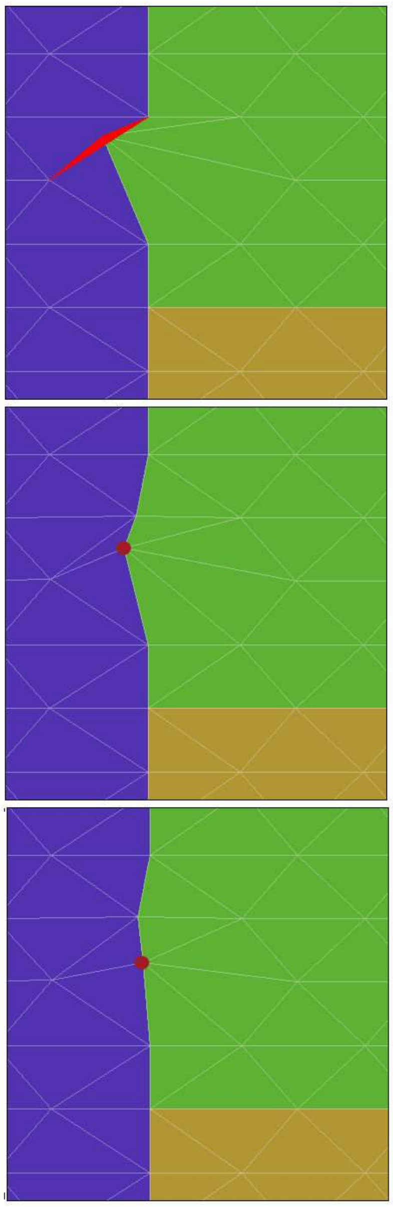


Figure 25: Element repairs. Top: inverted elements are displayed in red. Center: state of the mesh after application of the *Relax bad nodes* command (target node highlighted for clarity). Bottom: Further correction using the *Move node* command.

the view to the affected area. Click the *Relax bad nodes* command one or more times. The operation usually untangles the bad elements (Fig. 25, center). You can then complete repairs with the *Move node* command.

RELAX NODES IN BOX

Occasionally, you may have to relax additional nodes surrounding bad elements to untangle the mesh. After selecting this command, use the mouse to create a box around the affected region by defining two corner points. The program identifies the enclosed nodes and relaxes their positions. After using the command one or more times, proceed to the *Move node* command to complete the repair.

MOVE NODE

With this command you can move an individual node of the mesh to correct a boundary (Fig. 25, bottom). Move the mouse cursor close to the target node and click the left button. Move the mouse to the desired new position and click the left button. Click the right button to abort the operation.

The following commands correct errors in the assignment of region numbers. Commands are grouped under the *Change region number* command:

SINGLE NODE

You can spot nodes with incorrect region numbers using the *Node* plot type. This command changes the identity of an individual node. Use the mouse to identify the target node. A dialog appears with the present region number. Type in the new number and click *OK*.

NODES IN BOX

This command is similar to the *Single node* command except that you outline a region of the mesh by clicking on the two corners of a box with the mouse. The change effects all nodes within the box.

SINGLE ELEMENT

Use this command to change the region number of a single element. Move the mouse pointer inside the element and click the left button.

ELEMENTS IN BOX

This command is similar to the *Single element* command except that you define a region of the mesh by clicking on the two corners of a box with the mouse. The change effects all elements with center-of-mass inside the box.

6.5 Running Mesh autonomously

You can run **Mesh** from the command prompt under batch file control. In this mode you can set up your computer to perform an extended series of operations autonomously. **Mesh** operates in the background mode if a file prefix is supplied when the program is called. For example, suppose you type

```
\TRICOMP\MESH \DATA\IFACE <Enter>
```

from the command prompt. **Mesh** starts and searches for the file `IFACE.MIN` in the current or specified directory. If successful, the program runs in the background. A batch file to create meshes and find electrostatic solutions for a series of geometries may look like this:

```
REM Variation of focus electrode postion
REM Processing FELEC01
START \TRICOMP\MESH \GUNDESIGN\FELEC01\FELEC01
START \TRICOMP\ESTAT \GUNDESIGN\FELEC01\FELEC01
REM Processing FELEC02
START\TRICOMP\MESH \GUNDESIGN\FELEC02\FELEC02
START\TRICOMP\ESTAT \GUNDESIGN\FELEC02\FELEC02
REM Processing FELEC03
START \TRICOMP\MESH  \GUNDESIGN\FELEC03\FELEC03
START \TRICOMP\ESTAT \GUNDESIGN\FELEC03\FELEC03
REM Job completed
```

7 Mesh script format

7.1 Script file structure

The **Mesh** script is a text file that you can create and modify with an editor. You can also generate the file graphically with the drawing editor (Chaps. 4 and 5). The file must have a name of the form `FPrefix.MIN`. The string *FPrefix* is the run name, a descriptive title from 1 to 32 characters in length.

The **Mesh** script has the following structure:

```
Global
 (Global commands)
End

Region [RegName1]
 (Boundary vectors)
End

Region [RegName2]
 (Boundary vectors)
End

...

Region [RegNameN]
 (Boundary vectors)
End

EndFile
```

There is one *Global* section with commands that affect the entire solution and there are up to 250 Region sections with geometric information on each physical object in the problem. The *Global* section must appear first in the file. Its function is to set program parameters and characteristics of the foundation mesh. The *End* command marks the end of global input. The global commands can be entered in any order. **Mesh** reads commands until encountering the *End* command and then carries out the operations. The *Region* command marks the beginning of region boundary information that continues to the *End* command. Finally, *EndFile* signifies that all regions have been entered.

Mesh reads the commands with a free-form parser. A line consists of a command and one or more parameters separated by any number of delimiter characters. The following delimeters are recognized:

- Space [ ]
- Comma [,]
- Tab
- Colon [:]
- Left parenthesis [(]
- Right parenthesis [)]
- Equal sign [=]

You may use any of these characters to give your input files a distinctive style. You can also include indentations for readability. Commands and keywords can be entered in upper or lower case. **Mesh** ignores blank lines and comment lines. A comment line starts with the character [*] (asterisk) followed by any set of characters. Real number parameters can be entered in any valid format:

```
 2.3456
2.63E12
-1.95E+02
5
```

The last number is interpreted as 5.0. You can also include any amount of text after the *EndFile* command. In this case, no asterisk is necessary. As an illustration, Table 3 shows an example of a complete script:

7.2 Basic foundation mesh definition

The *XMesh* and *YMesh* (or *ZMesh* and *RMesh*) commands in the *Global* section specify the foundation mesh geometry. They define the limits of the solution rectangle and give the triangular element sizes along each axis. Use the commands *XMesh* and *YMesh* for a planar geometry and *ZMesh* and *RMesh* for a cylindrical system. The *XMesh* and *ZMesh* commands signify that a list of resolution zone properties for the horizontal axis will follow. The *End* command signals the end of the list. In this section we shall consider the simplest form with a single data line:

```
XMesh
 -6.00 10.00 0.25
End
```

The parameters in the data line are

```
Xmin  Xmax  Dx
```

for rectangular problems. In a cylindrical problem, the quantities in data lines of the *ZMesh* structure are

Table 3: **Mesh** script example

```
* File DTQUAD.MIN
GLOBAL
   XMesh
     0.000E+00   5.000E+00   9.999999E-02
   End
   YMesh
     0.000E+00   5.000E+00   9.999999E-02
   End
END
REGION  FILL Vacuum
* Vacuum
  L  0.00E+00  0.00E+00  5.00E+00  0.00E+00
  L  5.00E+00  0.00E+00  5.00E+00  5.00E+00
  L  5.00E+00  5.00E+00  0.00E+00  5.00E+00
  L  0.00E+00  5.00E+00  0.00E+00  0.00E+00
  END
REGION  FILL Dielectric
* Dielectric
  L  0.00E+00  5.00E+00  1.00E+00  5.00E+00
  L  1.00E+00  5.00E+00  1.00E+00  2.00E+00
  L  1.00E+00  2.00E+00  0.00E+00  2.00E+00
  L  0.00E+00  2.00E+00  0.00E+00  5.00E+00
 END
REGION  FILL Electrode_Up
* Upper electrode
  L  0.00E+00  2.00E+00  1.75E+00  2.00E+00
  L  0.00E+00  1.00E+00  1.75E+00  1.00E+00
  A  1.75E+00  1.00E+00  2.25E+00  1.50E+00  1.75E+00  1.50E+00
  A  2.25E+00  1.50E+00  1.75E+00  2.00E+00  1.75E+00  1.50E+00
  L  0.00E+00  1.00E+00  0.00E+00  2.00E+00
 END
REGION Gnd_Bound
* Grounded boundary
     L  0.00E+00  0.00E+00  5.00E+00  0.00E+00
     L  5.00E+00  0.00E+00  5.00E+00  5.00E+00
     L  5.00E+00  5.00E+00  0.00E+00  5.00E+00
   END
ENDFILE
```

```
Zmin  Zmax  Dz
```

The data in the example specify that the solution rectangle extends from -6.00 to 10.00 along the x axis with an approximate triangle base size of 0.25 units. For geometries of moderate complexity, pick Dx to give about 100-200 elements along the axis. The values may cover any range as long as $x_{max} > x_{min}$.

Similarly, the *YMesh* and *RMesh* statements define mesh properties along the vertical axis:

```
YMesh
 3.25  10.95 0.10
End
```

Here, the data line parameters are

```
Ymin  Ymax  Dy  (rectangular)
Rmin  Rmax  Dr  (cylindrical)
```

For cylindrical problems, values of $r_{min} < 0.0$ give non-physical results. **Mesh** stops with an error message if it detects negative values of r.

Spatial dimensions are flexible in **TriComp** programs. You may enter spatial quantities like x_{max} and y_{min} in any convenient set of consistent units. When you run a solution program, you can enter a conversion factor that changes coordinates from **Mesh** into SI units (meters). When preparing **Mesh** input, pick units that result in dimensions in the unity range for easy interpretation of the file and the best numerical accuracy. Examples are centimeters, microns, or miles.

Note. The data in the **Mesh** output file does not depend on whether you use *XMesh*-*YMesh* or *ZMesh*-*RMesh* commands. The option is included for your convenience. The choice affects axis labels in the plot menu and drawing editor within **Mesh**. If a script contains the *ZMesh* and *RMesh* commands, **Mesh** stops with an error message if it detects vertical coordinate values less than 0.0.

7.3 Defining regions

After **Mesh** processes the *Global* commands and sets up a foundation mesh, the program is ready for sequential processing of region sections that define the problem geometry. The *Region* command that marks the beginning of a region definition has two forms.

```
Region
Region Fill
```

You may also include a trailing region name of up to 24 characters to help document runs.

```
Region Fill UpperPlate
```

Note that the allowed delimiters (including spaces) may not be included as part of the region name. Therefore, *Focus_electrode* is valid but *Focus electrode* is not.

Mesh assigns numbers sequentially to regions as they appear in the script. Nodes and enclosed elements are marked with the region number as they are conformalized. The program creates a list of region numbers and names in the listing file in the form of comment lines. You can paste this information into the input script of the solution file to document the associated of physical properties with region numbers:

```
Assignment of region numbers
   Number of regions in the file:   6
   Region    Region
   number    name
   =================
*       1    VACUUM
*       2    ANODE
*       3    CATH_SUPPORT
*       4    FOCUS_ELEC
*       5    CHAMBER_WALL
*       6    CATHODE_SURF
```

The *Fill* keyword designates that the region should be filled. After processing the boundary vectors, sorting them, and checking that they define a closed curve, **Mesh** marks all internal nodes and elements with the current region number.

A complete region section may look like this

```
Region Fill Upper_Pad
 XShift: 2.5
 YShift: 0.2
* Upper pad, 1.0 microamp source
     A -5.0 30.0  0.0  35.0  0.0  30.0
     A  0.0 35.0  5.0  30.0  0.0  30.0
     A  5.0 30.0  0.0  25.0  0.0  30.0
     A  0.0 25.0 -5.0  30.0  0.0  30.0
End
```

It consists of 1) the *Region* command, 2) any number of comment lines, 3) the optional *XShift*, *YShift* and *Rotate* commands, 4) one or more arc, line or point vectors that define the boundary, and 5) an *End* statement. The following sections describe how to enter points, lines and arcs.

The *XShift* and *YShift* (or *ZShift* and *RShift*) commands have the following syntax:

```
XShift  xs
YShift  ys

ZShift  zs
RShift  rs
```

They cause a transformation of coordinates for all point, line and arc data of the region according to the formulas:

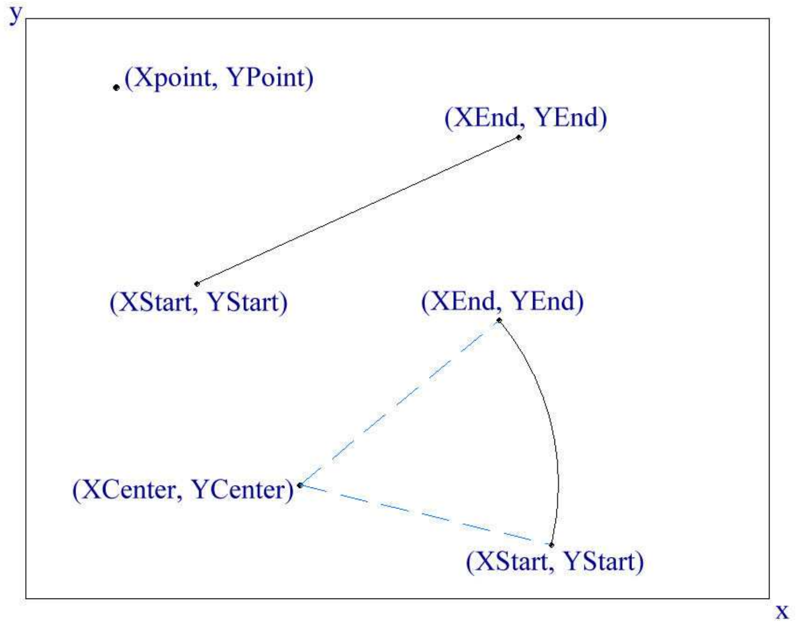


Figure 26: Parameters to define points, lines and arcs.

$$x(program) = x(file) + xs, \tag{5}$$
$$y(program) = y(file) + ys. \tag{6}$$

The commands are useful for geometries with repeated structures. You can simply copy and paste the vector data lines for one or more regions and use the *XShift* and *YShift* commands to position them. Note that **Mesh** does not provide automatic clipping. A region shifted with points outside the solution volume will generate an error message.

The *Rotate* command has the form:

```
Rotate  Ang [xc  yc]
Rotate  Ang [zc  rc]
```

The quantity *Ang* is the rotation angle in degrees. A positive rotation is in the counter-clockwise direction. The optional parameters x_c and y_c define the center point for the rotation. The default values are $x_c = y_c = 0.0$. As with the shift commands, the *Rotate* command acts on all vectors in the region section and the program does not automatically implement clipping. Note that the rotation is performed before shift operations.

7.4 Points

A point data line moves and marks a single node. The line has the format

```
P  XPoint  YPoint
P  0.1678 10.45
```

The letter *P* designates the vector type. The two real-number parameters (*XPoint* and *YPoint*) are the (x, y) or (z, r) coordinates of the desired node location (Fig. 26). The point position must be inside the solution rectangle. Be sure to use the same distance units as those in the *XMesh* and *YMesh* statements. Because zero-dimension points are inherently disconnected, they are not allowed in the definition of a *Filled* region boundary. You can include up to 2000 individual points in a region. Points can be combined with lines and arcs in an *Open* region. In electrostatic solutions, arrays of points are useful to simulate grids.

7.5 Lines

The following command defines a straight line:

```
L  XStart YStart XEnd YEnd
L  0.571 0.986 1.756 -0.234
```

The four real-number parameters are the coordinates of the starting and ending points (Fig. 26). The line must fit inside the solution rectangle. If the solution program handles cylindrical problems, then the quantities are interpreted as

$$XStart \to ZStart, \tag{7}$$

$$XEnd \to ZEnd, \tag{8}$$

$$YStart \to RStart, \tag{9}$$

$$YEnd \to REnd. \tag{10}$$

7.6 Arcs

The statement

```
A  XStart YStart XEnd YEnd XCenter YCenter
A  0.500 1.000 1.000 0.500 0.500 0.500
```

defines an arc of a circle. The six real-number values are the coordinates of the starting, ending and center points. Figure 26 shows the definition of the points. **Mesh** gives an error message if the start and end points are outside the solution rectangle but does not check whether all points on the arc are valid.

For a given set of start-end-center points there are two possible arcs with positive and negative angles. **Mesh** always picks the arc that spans less than 180^o. Use two or more lines to define arcs that span large angles.

7.7 Filled region boundaries

The vectors of *Open* regions need not be connected. In contrast, the boundary vectors of a *Filled* region must define a closed, continuous curve. The endpoints must meet to within a small tolerance distance. You can enter the vectors in any order. **Mesh** has a powerful sorting feature that reorders vectors so that the start point of a vector matches the end point of the previous one. When the program detects the keyword *Fill*, it picks a starting point and then attempts to walk around the boundary, looking for matching vectors and reversing start and end points when necessary. The resulting set of vectors with corrected order is recorded in the listing file, `FPrefix.MLS`. **Mesh** issues an error message if it does not return to the starting point of a filled region.

8 Mesh – advanced techniques

8.1 Variable resolution

Meshes with uniform element size are suitable for most applications. On the other hand, variations of element size (*variable resolution*) may sometimes improve solution accuracy and reduce computational time. Variable resolution is particularly useful in two instances:

- Accurate determination of fields near small structures in a large solution space.
- Simulation of an infinite-space boundary condition by surrounding a volume with an extended region of coarse elements.

In contrast to many finite-element programs, the **TriComp** codes use *structured* meshes. The term means that the elements have an ordered topology, making it possible to identify neighboring elements and nodes through index operations. Structure gives the advantage of high speed in programs that involve extensive search operations. On the other hand, the logical connections impose some limits on variable resolution.

You can introduce variable element size along one or both axes by adding multiple data lines to the *XMesh*, *YMesh*, *ZMesh* and *RMesh* statements that appear in the *Global* section. Consider the following example:

```
XMesh
  0.000  1.000 0.100
  1.000  2.550 0.200
  2.550  4.000 0.300
End
```

The statements convey the information that $x_{min} = 0.00$ and $x_{max} = 4.00$. Furthermore, the triangle base size along x is approximately 0.10 between 0.00 and 1.00, increases to 0.20 over the interval 1.00 to 2.55, and equals 0.30 from 2.55 to 4.00. Note that the intervals along the x axis in each data line must constitute a continuous range and be in order from x_{min} to x_{max}. The *YMesh*, *ZMesh* and *RMesh* data lines have the same format.

```
YMesh
  0.000  2.000 0.100
  2.000  4.000 0.250
End
```

Figure 27 shows the foundation mesh resulting from the two commands sets listed above.

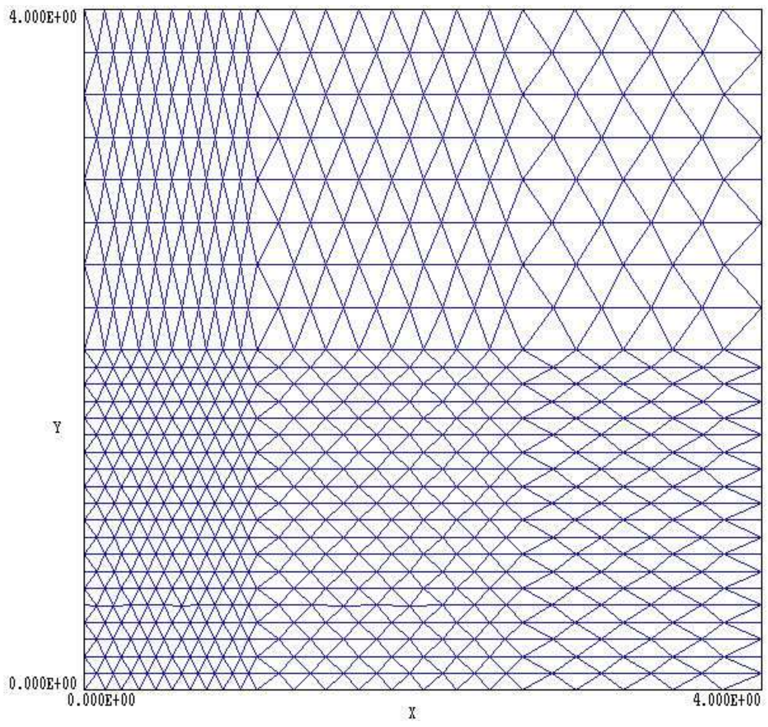


Figure 27: Variable resolution example.

8.2 Mesh smoothing

In the default setting, the *XMesh* and *YMesh* commands produce discontinuous changes in triangle size along each axis as in Fig. 27. Sometimes continuous variations in size may lead to more reliable fitting and better element shapes. The *PreSmooth* command relaxes the variations in element size in the foundation mesh. The command appears in the *Global* section and has the following form:

PRESMOOTH 6

The integer parameter is the number of smoothing cycles (default: 0). Higher values give more smoothing. Figure 28 shows the foundation mesh of Fig. 27 with 4 cycles of pre-smoothing.

The **TriComp** solution programs achieve the best accuracy when all elements have about the same shape (equilateral triangles). Acute triangles with very small angles may increase roundoff errors in the numerical calculations. After shifting nodes to boundaries, **Mesh** attempts to improve the shapes of elements by relaxing the positions of unclamped nodes. The process of mesh smoothing is controlled by the *Global* command:

SMOOTH NSmooth

The integer parameter is the number of relaxation cycles. Use $NSmooth = 0$ if you want to preserve the discontinuities in triangle size defined by the *XMesh* and *YMesh* statements. The default value is $NSmooth = 15$.

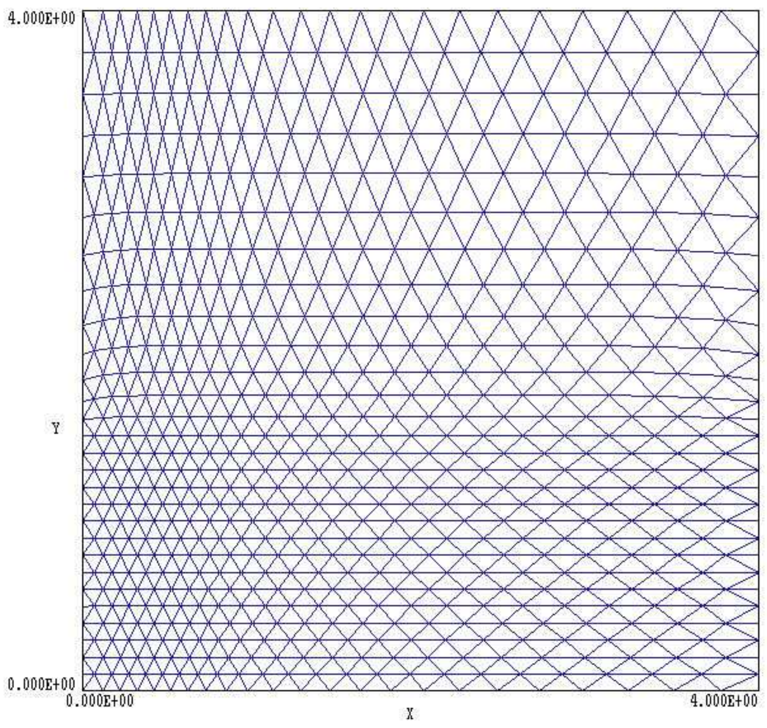


Figure 28: Variable resolution mesh with presmoothing.

Figure 29 shows a mesh with and without smoothing. Note that clamped nodes along the sides of the foundation mesh volume (at x_{min}, x_{max}, y_{min} and y_{max}) are allowed to slide along the sides during the process.

8.3 Setting foundation element shapes

By default, **Mesh** initially fills the foundation mesh with isosceles triangles. The triangles are almost equilateral when the element sizes are equal along the x and y directions. Equilateral triangles give the best performance in boundary fitting operations on slanted or curved surfaces. There are other element shapes that may be useful in special circumstances. The *TriType* command appears in the *Global* section and controls the triangle shape:

TRITYPE ISO
Fill the foundation mesh with isosceles triangles (Fig. 30, top). This is the default choice.

TRITYPE RIGHT
Fill the foundation mesh with right triangles (Fig. 30, center). This option may give better element shapes in systems where all boundaries are parallel to the x and y axes. Here, the term *better* means that all elements have about the same shape and that there are few triangles with acute angles. Do not use this option when there are slanted or curved boundaries. Note that right-angle triangles will be converted to isosceles triangles in the final mesh if $NSmooth > 0$.

TRITYPE GLASS [GlassAdjust]
Creates a foundation mesh of amorphous elements (Fig. 30, bottom). Under this option, **Mesh**

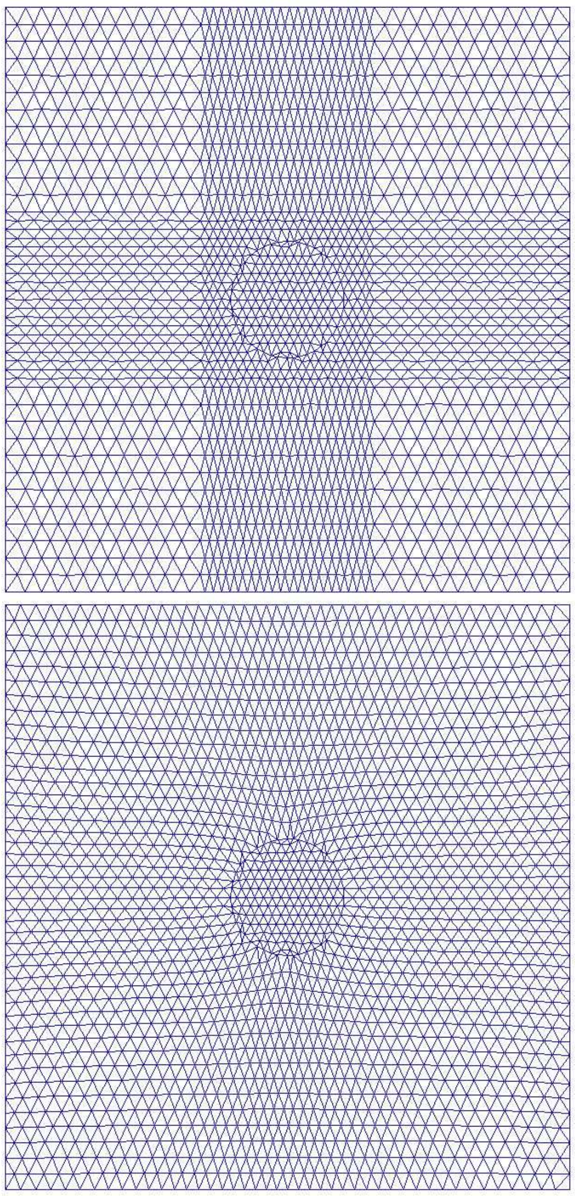


Figure 29: Variable resolution mesh without (top) and with (bottom) smoothing.

fills the foundation mesh with isosceles triangles and then adds random displacements to create a glass-like distribution. This option has been included to support the Field Precision **KB2** hydrodynamics code. In a shock simulation, ordered element boundaries may act as material dislocations giving false element displacements. The optional parameter *GlassAdjust* determines the degree of disorder. Generally the value should be between 0.0 and 0.5. The example of Fig. 30 uses *GlassAdjust* = 0.25. The default value is 0.2. Note that this mode gives no advantages for electromagnetic field or thermal simulations.

8.4 Autocorrection and boundary fitting parameters

Mesh has powerful capabilities for correcting element errors introduced by the conformalization process. The program automatically shifts positions of nodes connected to any inverted elements detected after the fitting process. In most cases, the autocorrection process eliminates the problem of inverted elements. In the rare case where the procedure fails to fix all elements, you can change properties of the foundation mesh or use the repair tools described in Sect. 6.4. By default, autocorrection is active. Use the following command if you want to deactivate the process and make corrections manually:

AUTOCORRECT OFF

There are two specialized *Global* commands that may be helpful in the event of an error while fitting boundary vectors. **Mesh** moves nodes during the fitting process. The *Relax* parameter controls whether the six neighbors of the shifted node are also shifted in position. If a target node is displaced by a vector **d**, then the neighbors are displaced by *Relax* × **d**. This process is beneficial, so it is usually not necessary to change *Relax* from its default value of 0.2. On the other hand, displacing neighboring vertices may cause problems in tight corners or when tracing the same boundary several times. You can adjust the value of *Relax* with the command

RELAX 0.25
The parameter should be a positive number smaller than 1.00. A value 0.00 turns off the adjustment process.

The *Tolerance* command sets the maximum distance such that two coordinate locations are taken as the same point. In other words, two points are identical if their separation is less than *Tolerance*. An approximate criterion of equality is necessary when dealing with inexact floating point inputs.

TOLERANCE 1.0E-5
Set a criterion to decide whether two real-number coordinates are identical. By default, **Mesh** sets a tolerance based on the dimensions of the solution rectangle.

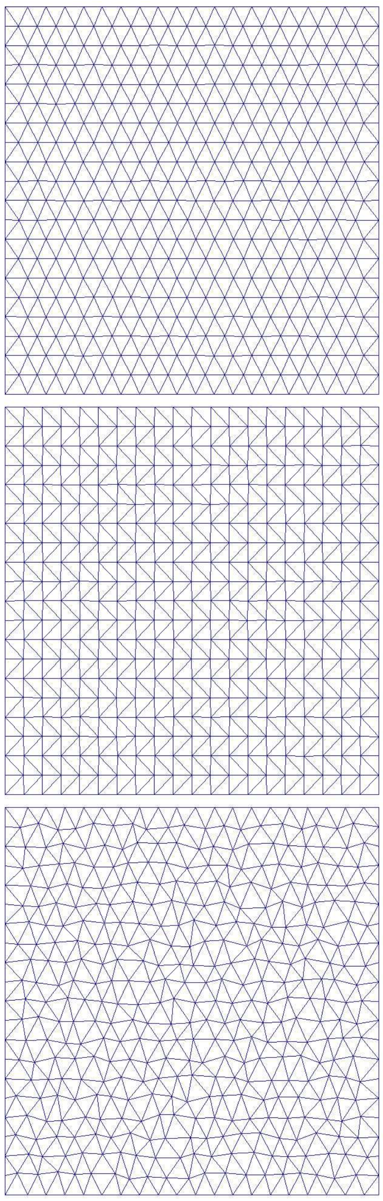


Figure 30: Options for foundation elements shapes. Top: Iso. Center: Right. Bottom: Glass.

8.5 Mesh generation problems

We have made extensive improvements to the reliability of **Mesh** over a ten-year period. Nonetheless, it is impossible to create a mesh generator that will accept any user input and perform flawlessly. There is always the possibility that the triangles of the foundation mesh will be inappropriate for the geometry of one or more regions. With some care in specification of the foundation mesh, all reasonable physical geometries can be modeled. This section reviews ways to avoid problems and methods to diagnose them.

If the boundaries do not look at all like your conception of the system, the most likely cause is a typographical error in the script. At the next level, **Mesh** may recognize the correct boundaries and process all the vectors but give a *Bad triangle* message. This means that the program had to displace nodes so far to match a boundary that some elements have been turned inside-out. On rare occasions, the autocorrection process may not be able to handle severe distortions. In this case, you should modify the foundation mesh to conform more closely to the system geometry. You can also repair a limited number of inverted elements using the interactive features of **Mesh** discussed in Sect.6.4.

There are three main causes for bad elements.

- The local size of elements in the foundation mesh is too large to fit the geometry of the specified objects.
- Objects have sharply pointed edges.
- The shape of foundation-mesh elements is inappropriate for a region outline.

The first problem can usually be solved by decreasing the size of triangles, either globally or locally, using the *XMesh* and *YMesh* commands to vary the resolution. The second problem is a special case of the first. **Mesh** may get lost if it walks to a sharply pointed edge along one side and then cannot find available nodes to walk back along the other side. Regarding the third problem, try to use equilateral triangles in the foundation mesh whenever possible.

There are several possible causes of a logical path error. The most likely one is a syntax error in the script leading to an inconsistent vector definition. A second possibility is when a line or arc vector is too small to resolve with the foundation elements. In this case, decide if the small detail is physically significant. If not, remove it. If the object is critical, reduce the element size or employ variable resolution in the *XMesh* and *YMesh* statements. A third possibility is that element shapes in the foundation mesh are inappropriate for the vector. An error may occur if you try to fit an arc in an area with very tall or short elements. Again, the solution is to change the element sizes in the *XMesh* and *YMesh* statements. Finally, a logical path error will occur if two vectors of an open region cross each other or intersect at positions other than the endpoints.

A higher-level problem is when mesh generation is apparently successful but the physical solution for the fields looks incorrect. The most common cause of non-physical solutions is incorrect ordering of regions in the **Mesh** script. A common symptom is strange field line behavior at boundaries. An infrequent cause of problems is the incorrect identification of elements and nodes inside a complex closed region. You can diagnose this problem using the *Element* and *Node* plot types (Sect.6.3) to spot entities that are out of place. Finally, on rare occasions you may observe local distortions of fields with an apparently good mesh. The

Table 4: Format of the Mesh output file (`MOU`)

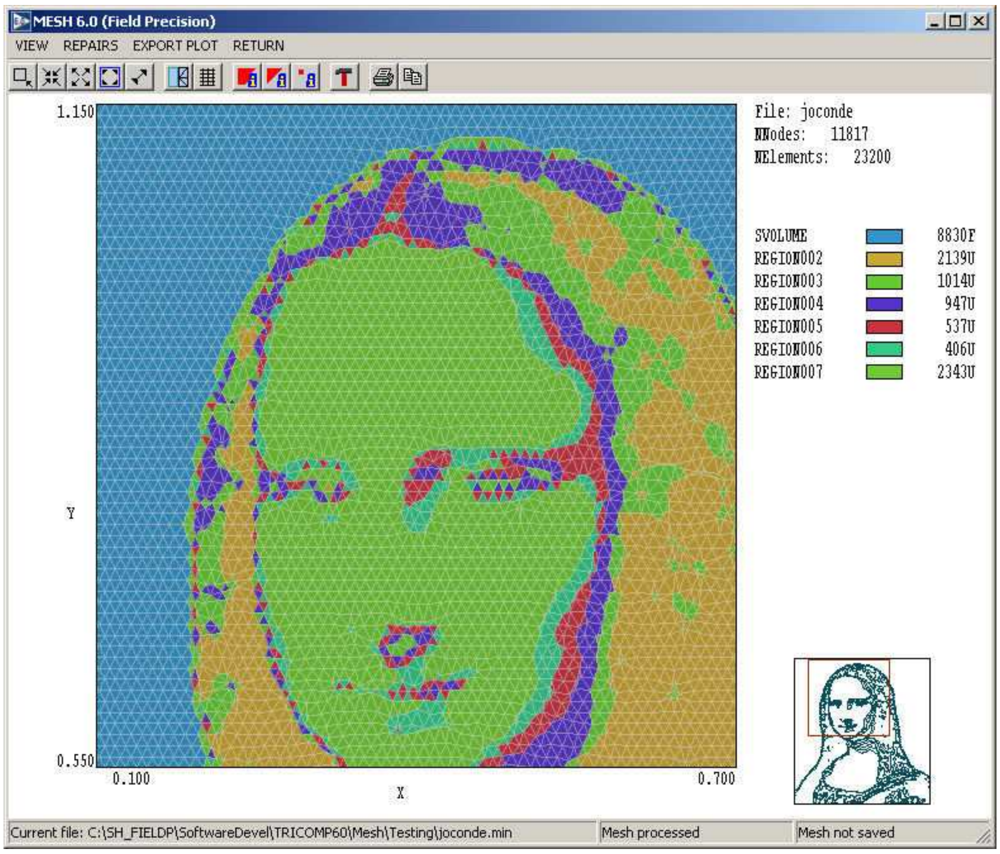
```
--- Run parameters ---
XMin: -1.00000000E+00
XMax:  1.00000000E+01
KMax:    111
YMin:  0.00000000E+00
YMax:  1.00000000E+01
LMax:    101

--- Nodes ---
   k     l   RgNo  RgUp  RgDn         x               y
 ============================================================
     1     1     4     4     0 -1.00000000E+00  0.00000000E+00
     2     1     4     4     0 -8.96963486E-01  0.00000000E+00
     3     1     4     4     0 -7.94126606E-01  0.00000000E+00
   ...
   110   101     5     0     5  9.90303742E+00  1.00000000E+01
   111   101     5     0     0  1.00000000E+01  1.00000000E+01


--- Region names ---
  NReg   Name
 ================================
     1   VACUUM
     2   WATER_LINE
     3   INSULATOR
     4   INNER_COND
     5   SHAPED_WALL
     6   XLINE_WALL
```

problem arises when elements in an area have bad shapes (very small angles). Check the mesh in the problem area by zooming in on the problem region with a *Facet* type plot. Often, you can fix the triangles by making small changes in the local resolution using the *XMesh* and *YMesh* commands. You can also move and relax selected nodes using commands of the *Repair* menu.

8.6 Format of the Mesh output file

The **Mesh** output file (`FPrefix.MOU`) is in text format. This feature makes it easy to access information with editors, spreadsheet programs and user analysis programs. The format is straightforward. Table 4 shows the initial part of a typical file.

The first section (*Run parameters*) is a header that gives information about the spatial limits and indices of the foundation mesh. The FORTRAN formats used are `[1P, E15.8]` for real numbers and `[I6]` for integers. The main mesh data is contained in the second section(Nodes). There is one data line for each mesh node. The first two integer quantities in a line are the (k, l) indices (horizontal and vertical) of the node. The third quantity, $RegNo$, is the region number associated with the node. Nodes outside the solution volume have $RegNo = 0$. The structured meshes have six elements surrounding each node. On the average there two elements per node. Therefore, each node line contains the region numbers of two elements. By convention, the elements associated with a node are 1) upward and to the right ($RegUp$) and 2) downward and to the right ($RegDn$). Elements outside the solution volume have $RegUp = 0$ or $RegDn = 0$. The final two real number parameters are the (x, y) or (z, r) coordinates of the node. A data line has the following format: [5I6, 1P, 2E16.8].



File: joconde
NNodes: 11817
NElements: 23200

SVOLUME	🟦	8830F
REGION002	🟨	2139U
REGION003	🟩	1014U
REGION004	🟪	947U
REGION005	🟥	537U
REGION006	🟩	406U
REGION007	🟩	2343U

Current file: C:\SH_FIELDP\SoftwareDevel\TRICOMP60\Mesh\Testing\joconde.min | Mesh processed | Mesh not saved

Figure 31: Mesh screenshot – creation of a mesh from an image.

9 Creating meshes from images

9.1 Introduction

The mesh-generation methods described in the previous chapters were designed to represent a moderate number of objects with relatively simple shapes and precise dimensions. The approach is well suited to simulations of mechanical systems but may be inefficient for systems with indefinite boundaries. As an example, suppose we wanted to study the effect of small electric fields on the regeneration of bones in the human leg. To construct a mesh, we might start from an MRI image of a leg cross-section to identify regions of different electrical conductivities (such as bone, muscle tissue and skin). To employ the standard method, we would outline region boundaries, measure dimensions, and convert the results to a series of line and arc vectors. The process involves considerable labor because biological systems have irregular and complex boundaries. Furthermore the effort of defining exact dimensions would be largely wasted because there is considerable variability between legs.

Mesh can automatically generate conformal meshes from image data (Fig. 31). The program can convert several types of two-dimensional image data directly to conformal triangular meshes:

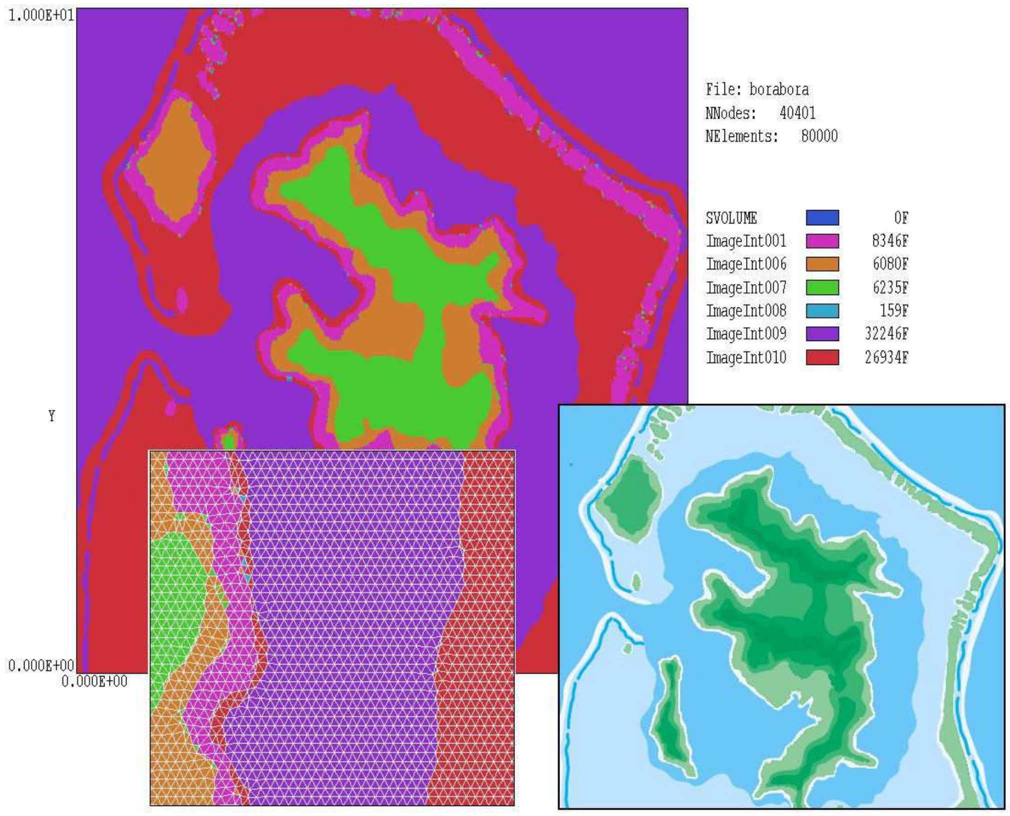


Figure 32: Generation of a conformal triangular mesh from a bitmap image. Lower right: input image. Lower left: detail of the mesh showing element boundaries.

- Medical images (MRI, X-ray, ...).
- Digital photographs of equipment.
- Scans of anatomical charts, blueprints and maps.
- Data files describing continuous variations of quantities like temperature and density in space.

Figure 32 illustrates an example, a mesh of Bora Bora resolved into regions by elevation. The original bitmap image is shown at bottom right. The inset at bottom left shows a details of the 40,401 node mesh with element boundaries displayed. **Mesh** required less than 2 seconds to perform the conversion.

The remainder of this section discusses characteristics of the two types of images recognized by **Mesh**: visual images and data images. Section 9.2 covers the structure of **Mesh** scripts to include images, while Sect. 9.3 describes commands and procedures for generating visual images. Section 9.4 illustrates techniuqes for data images. Finally, Sect. 9.5 summarizes image processing commands and rules.

In **Mesh** an image is a record of a quantity at the nodes of a two-dimensional rectangular mesh. In such a mesh we designate node positions with indices (I,J), where I gives the position of the node in the horizontal direction and J corresponds to the vertical direction as shown in Fig. 33. **Mesh** deals with two types of images.

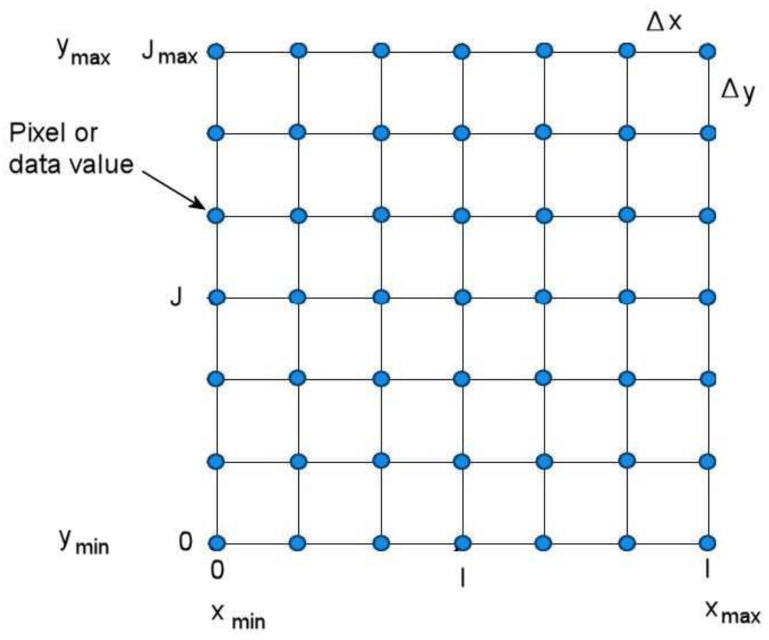


Figure 33: Definition of image quantities. Values are arranged on a two-dimensional square or rectangular mesh.

- Visual images (such as those generated by scanners and digital cameras) can be converted to bitmap format (BMP, PCX and PNG). The mesh for this image type consists of values defined at the node points of a square mesh ($\Delta x = \Delta y$). The recorded quantity is the RGB (red-green-blue) value of the pixel at the node.
- Data images are text files that consist of real number values defined at the nodes of a rectangular mesh (where Δx may not equal Δy).

9.2 Script Image section

To review, a standard **Mesh** script consists of a *Global* section followed by one or more *Region* sections. The *Global* section defines the properties of the foundation mesh (the shape of the solution volume and the approximate sizes of triangular elements that fill it). The *Region* sections contain vectors that outline objects in the solution space. To process a region, the program shifts nodes so that they lie on boundaries and marks nodes and elements contained within the boundary with the current region number.

Mesh scripts may also include one or more *Image* sections with the form:

```
IMAGE
 (Image commands)
END
```

The function of the construct is to load a file that contains two-dimensional information and to create several regions based on values in the file. *Image* sections must follow the *Global* section and at least one *Region* section. Typically, a script has the following structure:

```
GLOBAL
 (Global commands)
END

REGION FILL SolutionArea
 (Region commands)
END

IMAGE
 (Image commands)
END

REGION [FILL] Object01
  (Region commands)
END

REGION [FILL] Object02
  (Region commands)
END
...
ENDFILE
```

The first *Region* section defines the boundaries of the solution area, which may or may not fill the rectangle defined in the *Global* section. New regions created from the image are clipped to fit inside the solution area. *Region* sections that follow the *Image* section(s) may overwrite portions of the image. For example, an *Image* section could be used to define regions of different electrical conductivity in a cross-section of the liver. Subsequent *Region* sections could insert electrodes and insulators with precise dimensions.

The order in which *Region* and *Image* sections appear in **Mesh** scripts is important. The following rules apply:

- The currently-processed *Region* or *Image* over-writes the region identities of nodes and elements in the shared space.
- The function of the first *Region* is to define the solution area. **Mesh** shifts and clamps nodes on the specified boundary and sets $RegNo = 1$ for all included nodes and elements in response to the *Fill* keyword.
- An *Image* section reassigns elements (and associated nodes) to one or more new regions. An element is reassigned only if it already has a valid region number ($RegNo > 0$). In this way, the solution area region provides clipping information for the image.
- *Region* sections that follow an *Image* may change the region numbers in the shared space and may also shift nodes to conform to the specified region boundary vectors.

At this point, the rules may seem obscure. The process is actually easy to implement. The examples in the following sections will clarify image processing operations.

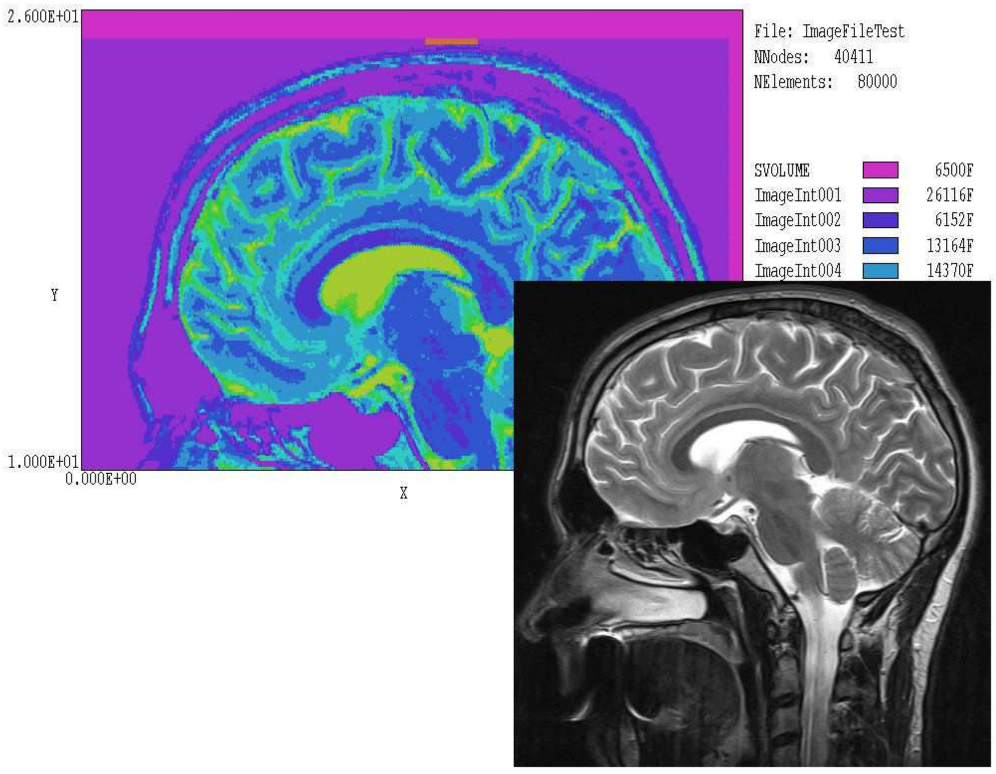


Figure 34: Example `BRAINMRI` – conversion of a bitmap image to a conformal triangular mesh.

9.3 Commands and procedures for visual images

This section introduces procedures for creating meshes from visual images using the example of Fig. 34, conversion of a gray-scale MRI bitmap image to a conformal mesh. The example illustrates many of the advanced features in **Mesh**. It also raises issues about what we can hope to accomplish with visual image conversion. Before beginning, it is important to understand some limitations and constraints.

- To perform electric field or thermal solutions, the image must be consistent with the two-dimensional symmetries handled by the codes of the **TriComp** series. The image of Fig. 34 does not have planar or cylindrical symmetry – we should consider it only as a demonstration of mesh techniques.

- In processing visual images, **Mesh** can divide the solution area into regions based on *Lightness* or *Hue* values. These values must have some correlation with the physical properties assigned to the regions in subsequent solution programs. Sometimes images may be directly useful. For example, we could photograph the cross-section of a complex iron pole piece with a distinctive color and use the picture to generate the corresponding region in a magnetic field solution. The image of Fig. 34 may not be directly useful. The lightness values depend on the concentration of nuclear species and there may be little direct correlation with quantities like thermal or electrical conductivity.

- The utility of images can be enhanced by pre-processing with programs like PaintShop Pro, PhotoShop or GIMP. For example, we could use the lasso method to select critical brain structures in Fig. 34 and then to change them to a distinctive color that could be easily identified by **Mesh**.

- Image sections are not transferred to the **Mesh** drawing editor. Use the drawing editor to create standard regions and then add images directly to the script.

With these precautions in mind, we can proceed to a discussion of the example. Two input files have been included in the *Mesh* example library:

- `BRAINMRI.BMP`. The bitmap image (shown on the lower right-hand side of Fig. 34) in Windows/OS2 bitmap format.

- `BRAINMRI.MIN`. The **Mesh** input script.

The script is listed in Table 5.

The *Global* section defines a foundation mesh with dimensions 25.0 cm in x and 26.0 cm in y. The first *Region* section outlines the complete area, aligns element nodes along the straight-line boundaries and sets all included elements and nodes to $RegNo = 1$. The *Image* section that follows includes the following four commands:

IMAGEFILE BRAINMRI.BMP 0.0 0.0 24.5 25.0
This command loads the image file. The optional real-number parameters specify that the image should be mapped into a rectangular portion of the solution volume with corners at [0.0, 0.0] and [24.5, 25.0]. The parameters can be used to specify physical dimensions and also to shift or to scale the image. If they are omitted, the program adjusts the image to fit the foundation mesh rectangle.

INTERVALS LIGHTNESS
0.0 20.0
20.0 30.0
30.0 40.0
40.0 50.0
50.0 60.0
60.0 70.0
70.0 80.0
80.0 90.0
90.0 100.0
End
This construct specifies that elements will be assigned to regions according to their *Lightness* value. *Mesh* converts RGB pixel values to HLS (hue-lightness-saturation). Hue values range from 0.0° to 360.0° and lightness values from 0.0 to 100.0. The nine intervals initiate the creation of nine new regions ($RegNo = 2$ to 10). Elements with lightness values in the range 0.0 to 40.0 are assigned $RegNo = 2$ and so forth.

CORRECT NCorrect
CORRECT = 5

Table 5: Contents of the file BRAINMRI.MIN

```
GLOBAL
  XMesh
    0.00 25.0  0.10
  End
  YMesh
   10.00 26.0  0.10
  End
END
REGION FILL SVolume
  L  0.0  10.0  25.0  10.0
  L 25.0  10.0  25.0  26.0
  L 25.0  26.0  00.0  26.0
  L  0.0  26.0  00.0  10.0
END
IMAGE
  ImageFile Brain_MRI.bmp 0.00 0.00 24.5 25.0
  Intervals Lightness
    0.0  40.0
   40.0  50.0
   50.0  60.0
   60.0  70.0
   70.0  80.0
   80.0  90.0
   90.0 100.0
  End
  Correct 5
END
REGION FILL Electrode01
  L  24.0  13.0  24.2  13.0
  L  24.2  13.0  24.2  15.0
  L  24.2  15.0  24.0  15.0
  L  24.0  15.0  24.0  13.0
END
REGION FILL Electrode02
  L  13.0  24.8  15.0  24.8
  L  15.0  24.8  15.0  25.0
  L  15.0  25.0  13.0  25.0
  L  13.0  25.0  13.0  24.8
END
ENDFILE
```

This command directs **Mesh** to perform five cycles of smoothing after element assignment. The process reduces jagged edges on boundaries between image regions (see Sect. 9.4).

The remaining two *Region* sections place electrodes on the skull. The region designated *Electrode02* is visible as the orange rectangle near the top of the mesh in Fig. 34.

The listing file `BRAINMRI.MLS` contains useful information on the image conversion process. **Mesh** creates a table of image information when the *ImageFile* command appears in the script:

```
   Image file size    NX:  480 NY:  489
   Image number of colors:      256
   Image function limits   Min:    0.000E+00  Max:    1.000E+02
   Image analyzed by LIGHTNESS
      LightMin    LightMax      Pixels
     =================================
        0.0000      2.0000         193
        2.0000      4.0000       11698
        4.0000      6.0000       36758
        6.0000      8.0000       16646
      ....
       94.0000     96.0000         431
       96.0000     98.0000         525
       98.0000    100.0000        2764
```

The data may be helpful in determining good intervals for the division into regions. After assigning elements to intervals based on the values of *Hue* or *Lightness*, **Mesh** creates the following table:

```
Distribution of elements in intervals
 Interval  FMin         FMax        FAverage     NElem
====================================================
     1   0.000E+00   2.000E+01   7.404E+00     26275
     2   2.000E+01   3.000E+01   2.600E+01      6153
     3   3.000E+01   4.000E+01   3.622E+01     13164
     4   4.000E+01   5.000E+01   4.524E+01     14370
     5   5.000E+01   6.000E+01   5.473E+01      5835
     6   6.000E+01   7.000E+01   6.525E+01      2734
     7   7.000E+01   8.000E+01   7.538E+01      1742
     8   8.000E+01   9.000E+01   8.507E+01      1113
     9   9.000E+01   1.000E+02   9.797E+01      2114
```

The table shows the bounding values of the intervals and associated number of mesh elements. The fourth column (marked *FAverage*) is the area-weighted average of *Hue* or *Lightness* over the assigned elements. This quantity may be helpful dealing with the data images discussed in the next section. The final table shows the association of the intervals with new regions in the mesh. The program issues an error message if any of the regions have no elements.

9.4 Creating meshes from data images

A data image is a file that contains a set of real values $F(I, J)$ defined at the points of a rectangular mesh (Fig. 33). The indices correspond to positions:

$$X_I = x_{min} + I\ \Delta x, \quad Y_J = y_{min} + J\ \Delta y, \tag{11}$$

where

$$\Delta x = \frac{x_{max} - x_{min}}{I_{max}}, \quad \Delta y = \frac{y_{max} - y_{min}}{J_{max}}. \tag{12}$$

The file has the following format:

```
Line 1: IMax  JMax
Line 2: XMin YMin XMax YMax
Lines 3 through 2 + (IMax+1)(JMax+1)
  F(0,0)
  F(1,0)
   ...
  F(IMax,0)
  F(0,1)
   ...
  F(I,J)
   ...
  F(IMax,JMax)
```

The index limits I_{max} and J_{max} are integers. All other quantities are real numbers in any valid format. The file may also include blank lines and comment lines that begin with an asterisk [*]. Note the order of values: J is the outer loop and I is the inner loop.

The **Mesh** example directory includes the Perl script `testimage.pl` which illustrates how to create a data file. You can modify the script to make files for any function $F(x, y)$. As supplied, the script records values of the function:

$$F(x, y) = \sin\left(\frac{\pi x}{5.0}\right)\cos\left(\frac{\pi y}{8.0}\right), \tag{13}$$

over the range $-5.0 \leq x \leq 5.0$ and $-4.0 \leq x \leq 4.0$. The resulting file `cosine.dat` is used for the following examples.

The script `COSINEDEMO.MIN` (Table 6) illustrates the method to integrate a data image into a mesh. In this example, the first region covers a bevelled section of the full solution rectangle. Figure 35 shows the resulting mesh. Note that the image has been clipped to fit within the first region. The image section contains three commands. The *Intervals* and *Correct* commands have functions similar to those discussed in Sect. 9.3. The following new command appears.

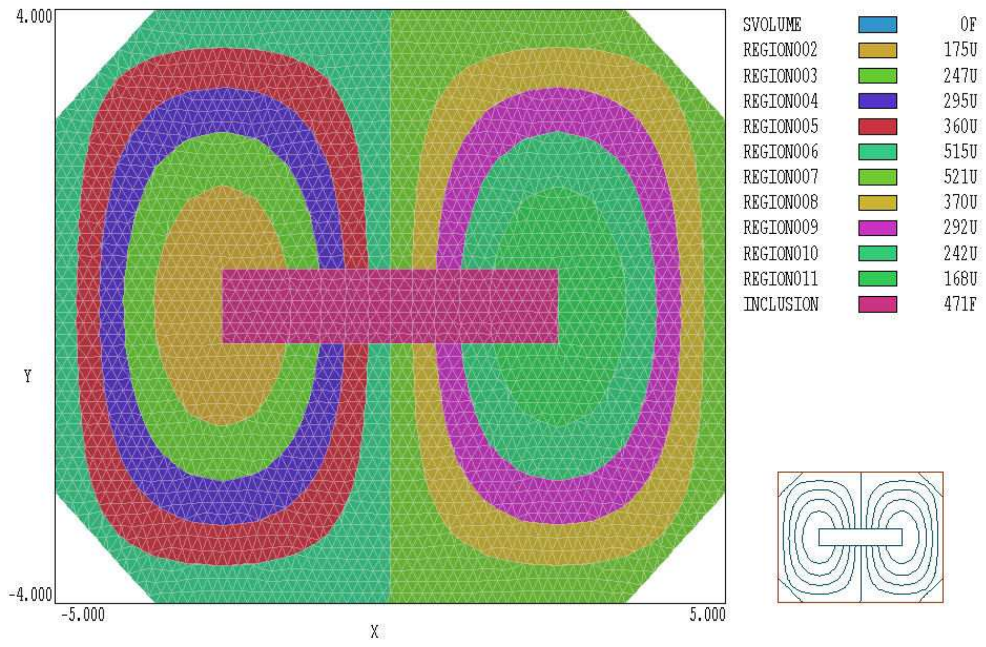


Figure 35: Mesh created with the script `COSINEMEDO.MIN` and the image file `COSINE.DAT` using the *Fit* option in the *DataFile* command and $Correct = 9$.

DATAFILE COSINE.DAT FIT

Load an image from the file and assign elements according to the data values and the intervals defined by the interval command.

The keyword *Fit* in the command instructs **Mesh** to move nodes on the boundaries of regions so that they closely correspond to the curves $F(x,y) = F_{max}$ and $F(x,y) = F_{min}$. Here, F_{max} and F_{min} are the bounding values of the region defined with the *Intervals* command. **Mesh** uses the following procedure to fit surfaces:

- Collect the set of element facets that constitute the boundaries of each region in the image.
- Divide the facets into two sets depending on whether they are closer to F_{max} or F_{min}.
- For each node at position $\mathbf{X}_0$ connected to a facet near F_{max}, find the value of the image function and its gradient at the current node position, F_0 and ∇F_0.
- Calculate the displacement vector

$$\delta\mathbf{x_0} = \frac{\nabla F_0}{|\nabla F_0|^2}\,(F_{max} - F_0).$$

- Correct the node position according to $\mathbf{X}_1 = \mathbf{X}_0 + \alpha\ \delta\mathbf{x_0}$, where $\alpha < 1.0$.
- Perform the same operations on the F_{min} nodes.
- Repeat the procedure through *NCorrect* iterations to move the nodes incrementally toward the ideal surface.

Table 6: Contents of the file COSINEDEMO.MIN

```
GLOBAL
  XMesh
    -5.0 5.0 0.15
  End
  YMesh
    -4.0 4.0 0.15
  End
  Smooth 5
END
REGION FILL SVolume
  L  -3.5 -4.0  3.5 -4.0
  L   3.5 -4.0  5.0 -2.5
  L   5.0 -2.5  5.0  2.5
  L   5.0  2.5  3.5  4.0
  L   3.5  4.0 -3.5  4.0
  L  -3.5  4.0 -5.0  2.5
  L  -5.0  2.5 -5.0 -2.5
  L  -5.0 -2.5 -3.5 -4.0
END
IMAGE
  DataFile COSINE.DAT Smooth
  Intervals
   -1.0 -0.8
   -0.8 -0.6
    ...
    0.6  0.8
    0.8  1.0
  End
  Correct 9
END
REGION FILL Inclusion
  L -2.5 -0.5   2.5 -0.5
  L  2.5 -0.5   2.5  0.5
  L  2.5  0.5  -2.5  0.5
  L -2.5  0.5  -2.5 -0.5
END
ENDFILE
```

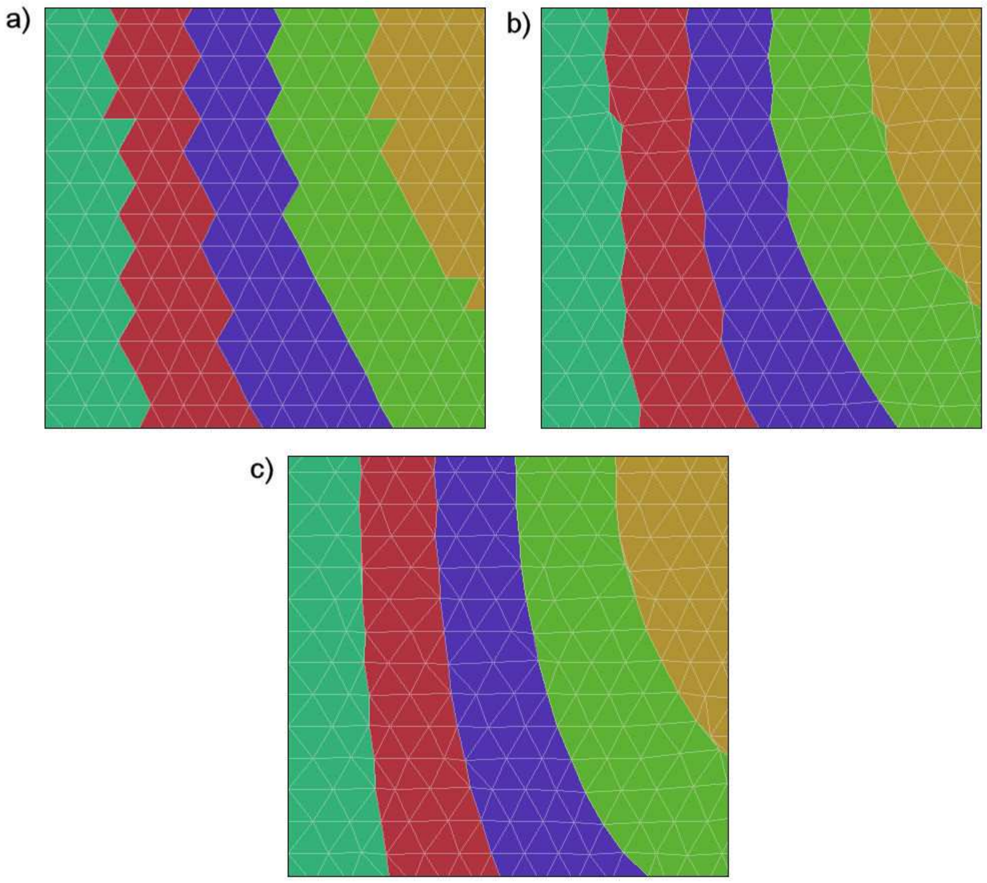


Figure 36: Closeup view of the mesh of Fig. 35. *a*) Fitting and smoothing suppressed by setting $NCorrect = 0$. *b*) Smoothing for $NCorrect = 9$ cycles. *c*) Fitting for $NCorrect = 9$ cycles.

It is important to recognize that the procedure is valid only when the gradient of the function is defined. Noisy or discontinuous data will lead to unpredictable boundaries and distorted elements. In this case, use the keyword *Smooth* in the *DataFile* command. This option invokes the smoothing routine used for visual images.

Figure 36 illustrates smoothing and fitting for the `COSINEDEMO` example. Figure 36*a* shows the state of the mesh where smoothing or fitting have been suppressed with the command:

```
CORRECT 0
```

Elements are assigned to a region depending on whether the function evaluated at the center of mass lies within the region interval. This process leaves jagged edges on the boundary. Figure 36*b* shows the modified mesh with nine cycles of *smoothing*, the equivalent of filing the edges. Although the boundaries have better continuity, their positions are not necessarily more accurate. Finally, Figure 36*c* shows the mesh with nine cycles of *fitting*. This option gives accurate boundaries that lie close to contours of the image function. The process is the preferred option, but can be applied only if the image function and its gradient are continuous functions of position.

9.5 Image command reference

One or more *Image* sections may be mixed with *Region* sections. At least one *Region* section must appear before the first *Image* section. The first region defines the boundary of the solution volume for image clipping. The over-writing rule applies to the regions defined by *Regions* and *Image* sections. The following commands may appear in any order within an *Image* section:

IMAGEFILE FileName [XIMin YIMin XIMax YIMax]
IMAGEFILE = tulsa.bmp (0.0, 0.0, 5.0, 5.0)
Load a visual image file in bitmap format (BMP, PCX or PNG). The file must be available in the working directory. If the optional real number parameters X_{imin}, Y_{imin}, X_{imax} and Y_{imax} appear, the image is mapped to a rectangle with corners at (X_{imin}, Y_{imin}) and (X_{imax}, Y_{imax}) which may or may not lie completely within the solution volume. If the parameters are omitted, the image is mapped to fill the boundaries of the solution rectangle $(X_{min}, Y_{min}, X_{max}, Y_{max})$.

DATAFILE FileName [Fit, Smooth]
DATAFILE = VARDIELECTRIC.DAT (Fit)
Load a data image file in the text format described in Sect. 9.4. The file must be available in the working directory. The optional keyword *Fit* indicates that nodes on boundaries of the image regions should be shifted by a mathematical calculation to lie on appropriate contours of the image function. This option should be employed only if the image function and its derivatives are continuous. Otherwise, use the *Smooth* option (the default).

INTERVALS [Abs, Rel] [Hue, Lightness]
FLow(1) FHigh(1)
...
FLow(NInt) FHigh(NInt)
END
This structure defines intervals for image element assignment. The command causes the addition of $NInt$ regions to the mesh. The regions are numbered consecutively and assigned default names of the form *Region003*,.... The program issues an error message if the total number of regions in the mesh exceeds 250. Each data line contains the upper and lower limits of the region (real numbers). The intervals should not overlap, but they need not be continuous or uniform. The option *Abs* (the default) indicates that values of the image function will be compared directly to $F_{low}(i)$ and $F_{high}(i)$ to see if an element lies in a region. The option *Rel* indicates that a relative comparison will be made. In this case, values of $F_{low}(i)$ and $F_{high}(i)$ should lie between 0.0 and 1.0. The comparison quantities are $F_{low}(i) \times F_{min}$ and $F_{high}(i) \times F_{max}$, where F_{min} and F_{max} are the calculated limits of the image function. Values of lightness in RGB images range from 0.0 (black) to 100.0 (white), while hue varies from 0.0° to 360.0°, where 0.0° corresponds to red, 120.0° to green, 180° to cyan and 240° to blue.

CORRECT NCorrect
CORRECT = 9
The assignment of triangular elements of the foundation mesh to regions leaves jagged boundaries. The fitting or smoothing processes (Fig. 36) average positions along region boundaries for $NCorrect$ cycles. A zero value of $NCorrect$ gives no boundary modification while a high value gives strong smoothing or fitting. The *Fit* option applies only to data images and should be applied only to continous functions. Smoothing should not be applied to speckled images where a region may consist of discontinuous single elements. Note that excessive boundary correction may lead to mesh distortions. (Default: $NCorrect = 0$.)

Index