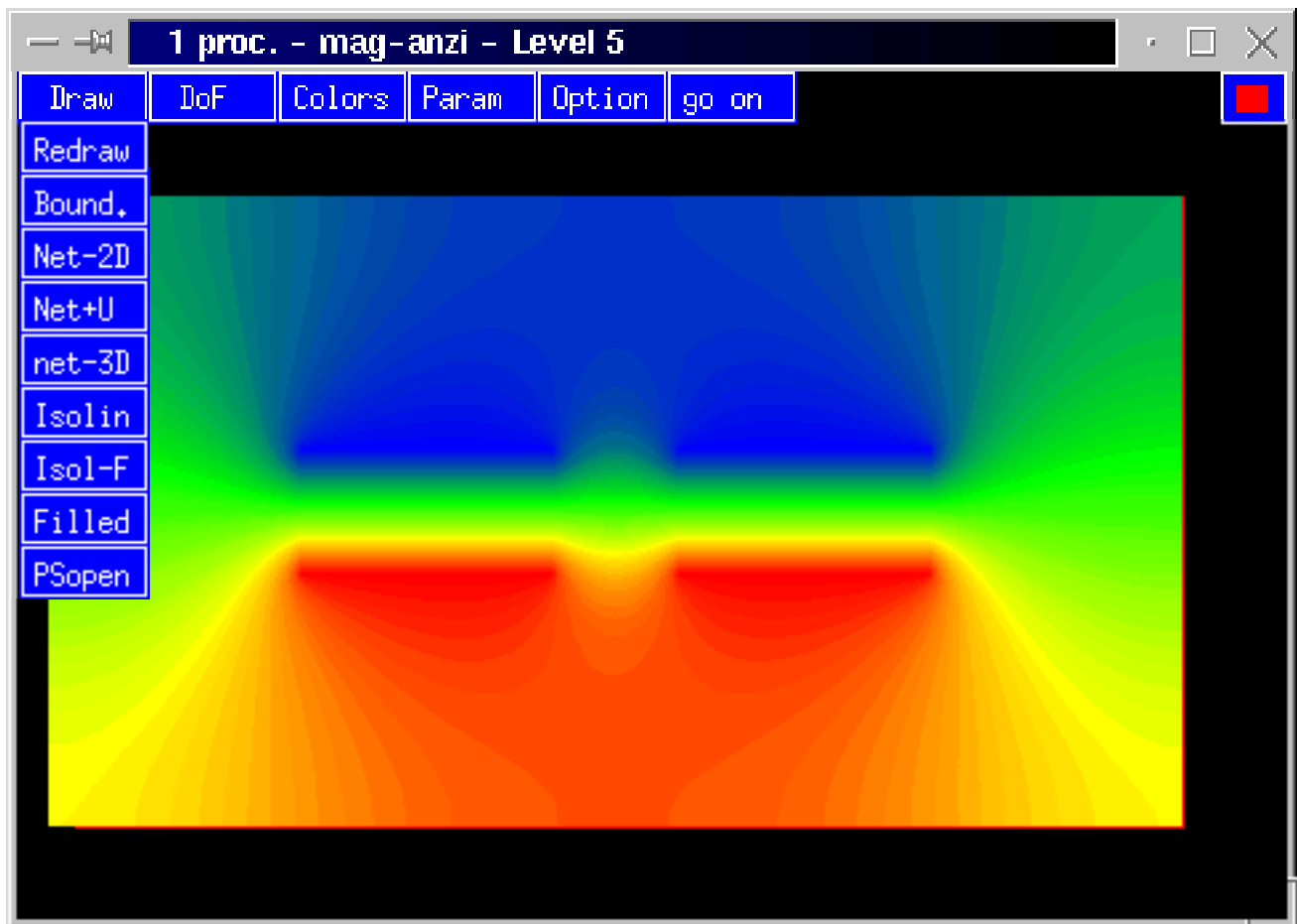


## Visualization of 2D Domains

This part of the visualization package is intended to supply a simple graphical interface for 2-dimensional finite element data structures. Furthermore, it is used as the low level interface for certain 3D graphics.

This graphics package uses the interface of the X11 library, no further extensions. Hence, it has been portable to any unix-like machine, including special parallel computers as GCPowerPlus. No third-party libraries are required. The look and feel of the user interface was designed to support most of the daily requirements in testing parallel algorithms. The left site shows a typical view for one example and most of the menu items that are available in the graphics window.

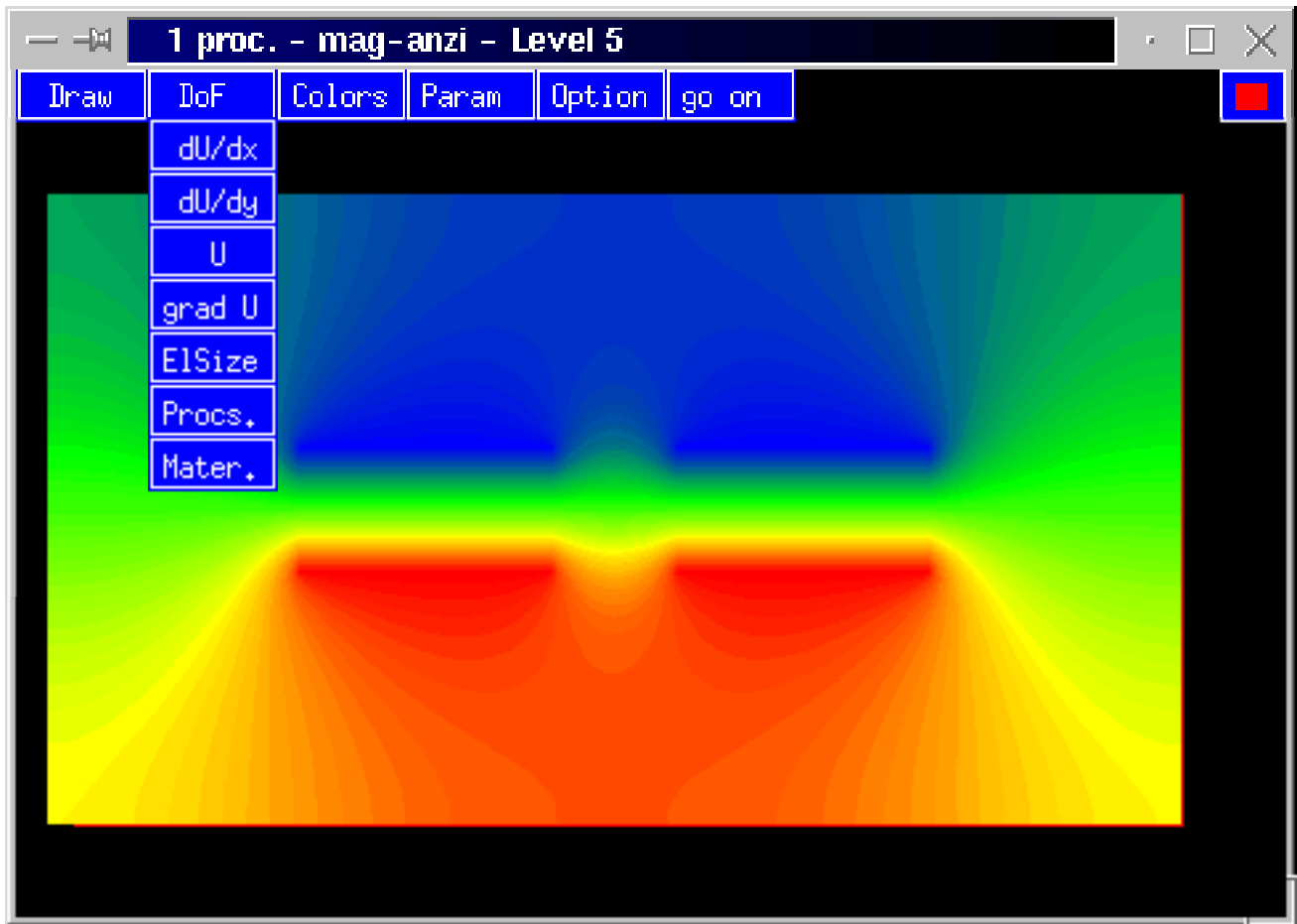
Note: For detailed information and helping hints click on the menu to the left.



Draw

**Pull down menu to select the display mode.  
The selected mode is performed immediately.**

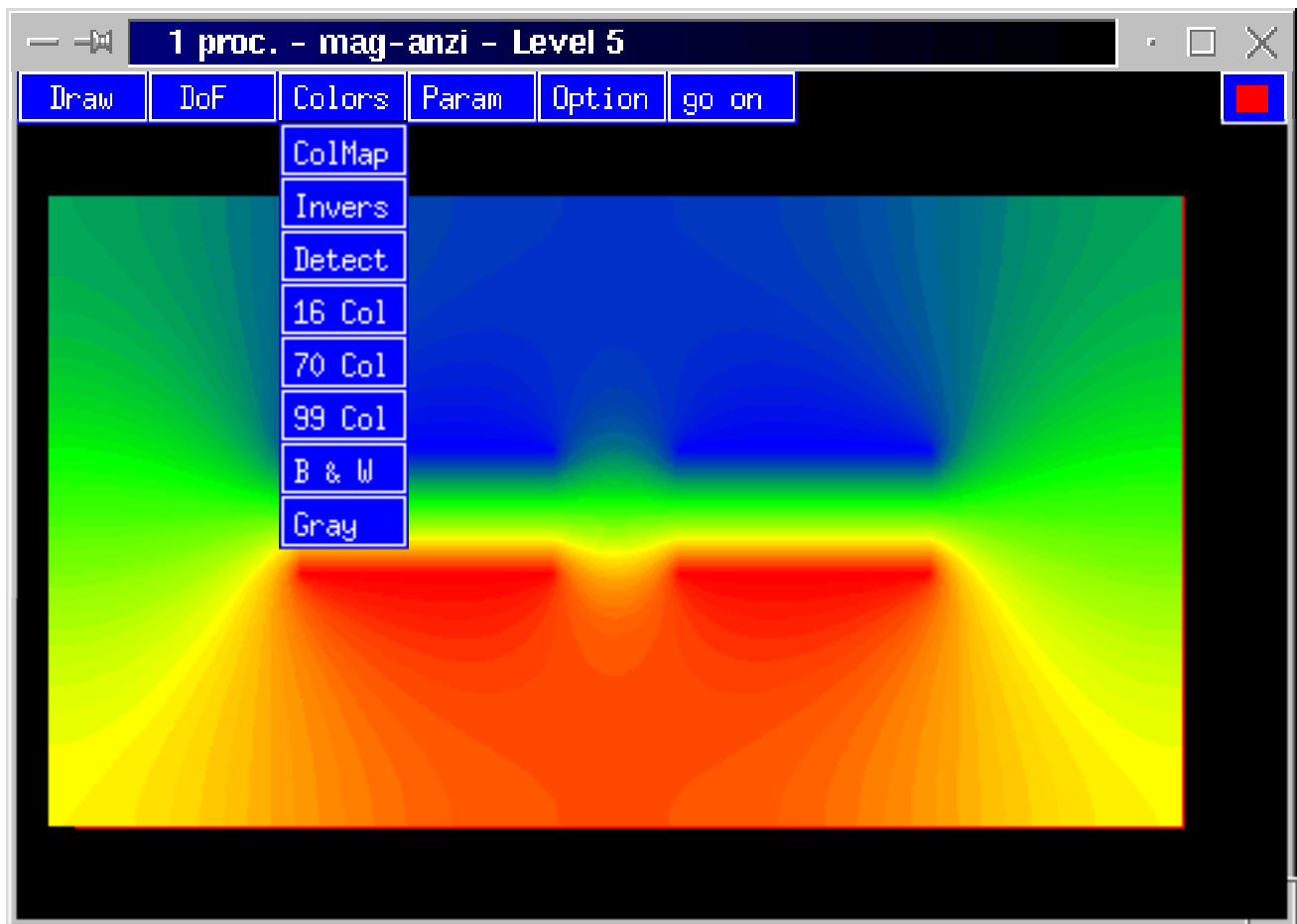
**Home**



DoF

**Pull down menu to select the component of the solution which is to be displayed by the next draw command.**

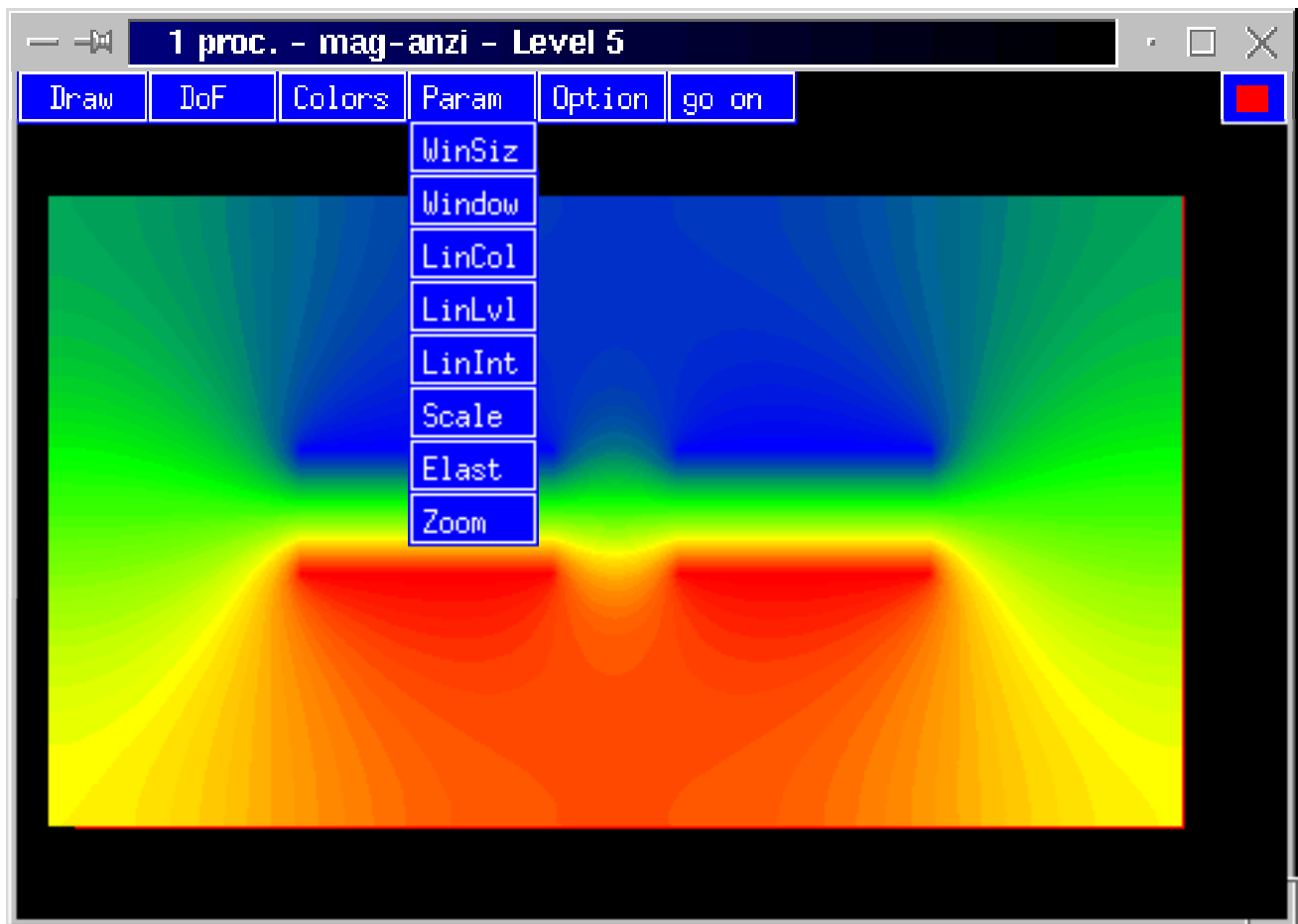
**Home**



Colors

Pull down menu to select one of various predefined color palettes.

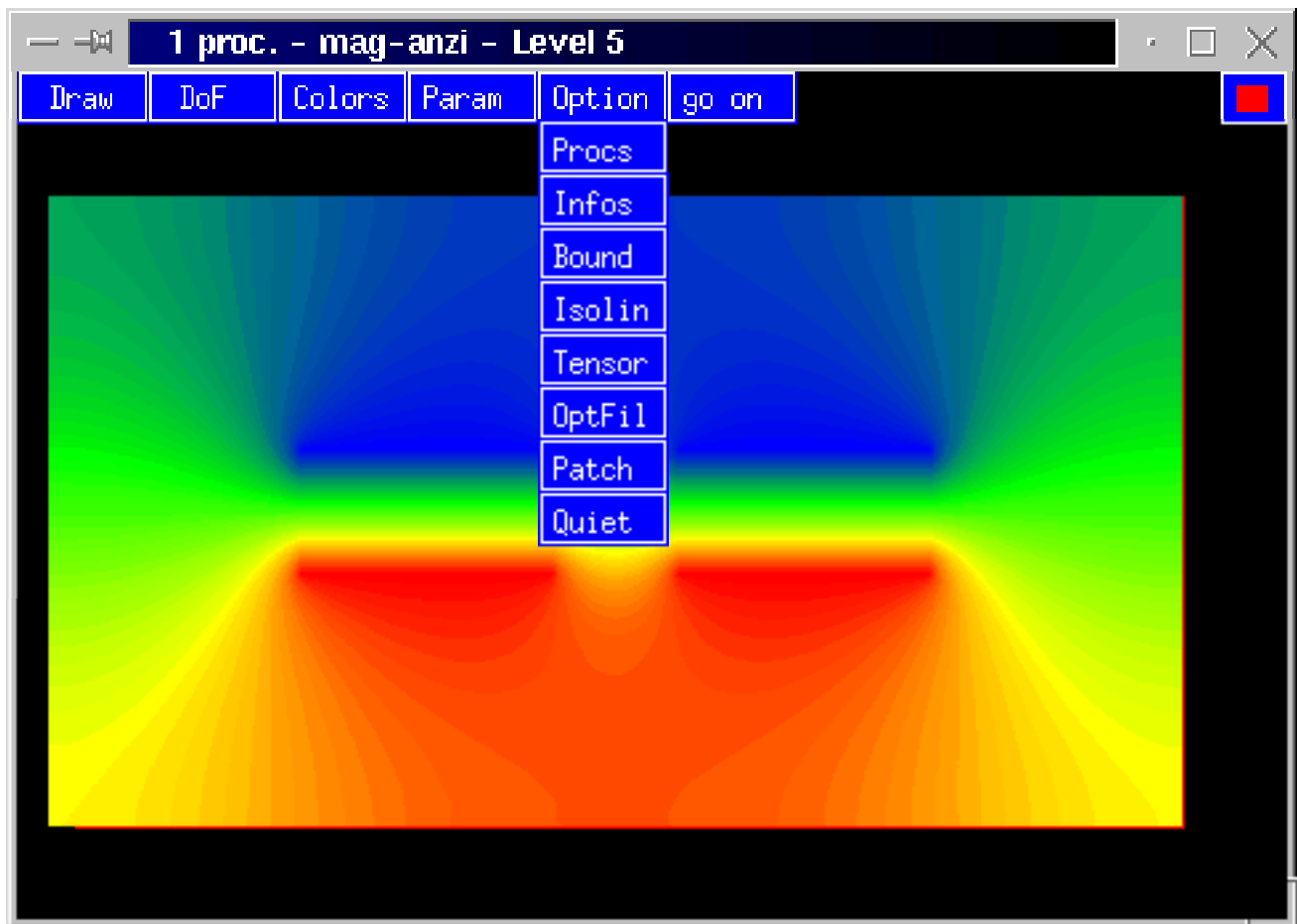
**Home**



Param

**Pull down menu to set various parameters for drawing modules.**

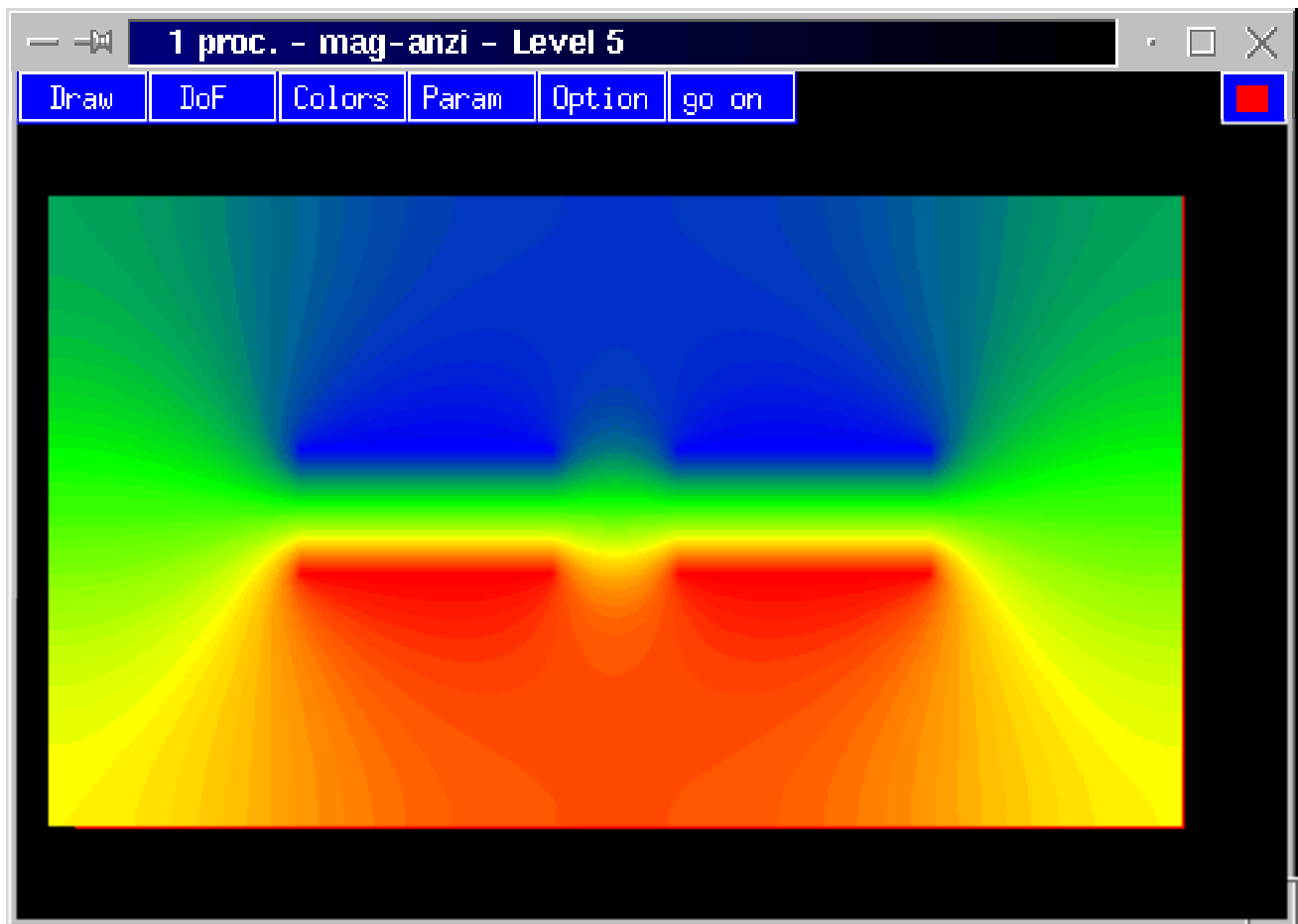
**Home**



Option

**Pull down menu to set or switch some additional options.**

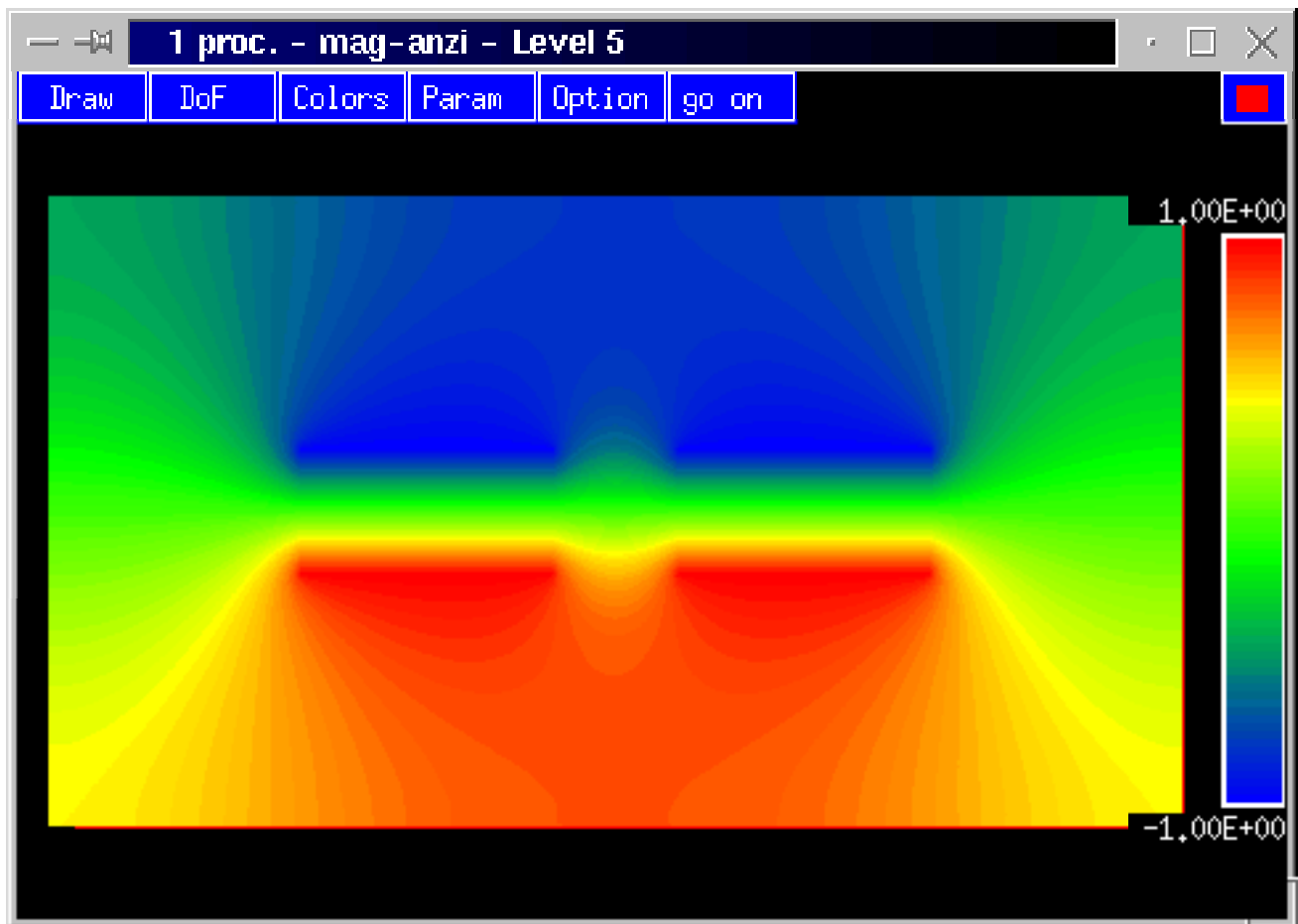
**Home**



go on

Leave the interaction and return to the program.

**Home**



[Home](#)



Redraw

Clear the graphics window and repeat the previous drawing, with possibly changed other options (e.g. line color, colormap, scaling, selected component).

**Back**

**Home**



Bound.

Draw only the boundary of the subdomains. There is an option to switch between drawing all boundaries of the coarse grid or only the boundaries of the current distribution of subdomains to the processors. The color of the lines is green in the case of Dirichlet boundary conditions and red for Neumann boundary conditions.

**Note:**

We assume the information on boundary conditions to be stored in the node field as a bit mask indicator in the third entry after the coordinates  $x$  and  $y$ . If this information is not stored this display mode has no output.

[Back](#)

[Home](#)



Net-2D

Tensor

Vector

Draw the current mesh. In general, the grid lines are drawn in a standard color (white or black). However, if **[Procs]** is selected from the **[DoF]**-submenu, the grid lines are colored according to the processor number, and for **[Mater]** the grid line colors correspond to the material numbers.

The **[Net-2D]**-submenu will change either to **[Tensor]** (draw small crosses) or **[Vector]** (draw small arrows) as described in the **[Tensor]**-submenu of **[Options]**.

[Back](#)

[Home](#)



Net+U

Show the deformation of a mesh by adding the displacement vectors to the node coordinates and displaying the grid. The line color is the same as for isolines (see below) and may be selected from the **[Param]**-submenu entry **[LinCol]**. The scaling for the deformation may be changed by the **[Param]**-submenu entry **[Elast]**.

**Note:**

A large (scaled) deformation may exceed window bounds and will be clipped. It is intentionally not scaled to window size in order to have a real comparison with the undeformed mesh.

[Back](#)

[Home](#)



net-3D

Draw a 3-dimensional view of the grid with the solution selected from the **[DoF]**-submenu giving the "height" of the grid points. If this drawing mode is selected there appears a little coordinate system in the upper left corner of the window. Clicking this coordinate system will allow to change the viewpoint by rotating via  $x$ -,  $y$ - or  $z$ -axis (press the corresponding key, and shift-key for backwards). Other keys are available for manipulating the viewpoint: **u,v,w,p,0,1,2,3,\*h** - try it. **<ESC>** will reset to the previous view, **<RETURN>** will accept. You must redraw your current solution explicitly to apply the new viewpoint.

**Back**

**Home**



Isolin

Draw isolines of the current component of the solution. The background is not changed (so you can draw different things in the same picture). Before drawing, the number of isolines can be changed in the **[Param]**-submenu ( **[LinLvl]** ). The color of isolines is blue by default and may be changed in the **[Param]**-submenu { **[LinCol]** }.

**Back**

**Home**



Isol-F

Draw contours of the domain, filled with a standard color (white or black) and isolines as in the previous menu point ( **[Option]** => **[Isolin]** ).

**Back**

**Home**



Filled

Fill the domain with colors of the current palette corresponding to the current component of the solution. Coloring is done by linear interpolation between the grid points.

**[Back](#)**

**[Home](#)**



**PSopen**

Open a new file for postscript output. The file name has to be entered in the console window where the program was started from. After opening this file you have to redraw everything you want to see in your postscript picture.

**Note:**

One postscript file will contain only one image, written as EPS (encapsulated postscript). If you need multiple pictures you will have to close the first file and open a new one. Once a file has been opened the menu entry will be changed to **[PSclos]**. The postscript file will be closed explicitly by selecting **[PSclos]** or automatically by selecting **[goon]** to leave the graphics window.

**Back**

**Home**



[dU/dx](#)

[dU/dy](#)

The first entries of this submenu are taken from the parameters supplied by the user-defined subroutine **getdofs**. The menu list given by **getdofs** will be terminated by an entry interpreting the first two components as vector.

[Back](#)

[Home](#)



More user supplied components for any degrees of freedom of the computed solution.

**Back**

**Home**



grad U

The last entry specified by the user describes the 2D-vector which is built by the first two components.

**Back**

**Home**



**EISize**

Display the (2D-) elements in colors corresponding to their element size. This may be useful for adaptive mesh refinement only.

**[Back](#)**

**[Home](#)**



Procs.

Display subdomains from different processors in different colors. This menu item exists if the program is running on a parallel machine.

**Back**

**Home**



Material.

Display materials in different colors. This menu item appears if the program was called with an additional vector of material indices.

**Back**

**Home**



ColMap

Switch between default and private colormap. On pseudo-color screens there may be a maximum of 256 entries to the default color map which is shared by all applications. Clicking this switch will use a private colormap for the graphics window. In this case the colors outside this window will change until the mouse pointer leaves the window - then the colors inside the window will be "wrong". On true-color screens this switch may have no effect.

[Back](#)

[Home](#)



Invers

Switch the colors black and white. The default is black background and white grid lines. This does not affect the postscript output, which is always on white background.

**[Back](#)**

**[Home](#)**



Detect

Select a default palette which is the same as **[16 Col]** for pseudo-color screens or **[70 Col]** for true-color screens.

**Back**

**Home**



16 Col

An EGA-like palette of 14 colors (black and white are not used for coloring data areas).

Useful to show better contrasts.

**[Back](#)**

**[Home](#)**



70 Col

Rainbow palette: blue - green - yellow - red.  
Useful for a more continuous coloring.

**Back**

**Home**



99 Col

Extended rainbow palette: magenta - blue - green - yellow - red - white.

**Back**

**Home**



B & W

Black and white coloring, i.e. alternating black and white areas.

**Back**

**Home**



Gray

Grayscale with up to 100 levels. The user is prompted for the number of gray levels between black and white. Grayscale is intended to use for printing on non-color printers, since the representation of colors (e.g. of the rainbow palette) might depend on printer drivers.

This palette is best-suited to show some lighting effects if this program is used for 3D surfaces.

**[Back](#)**

**[Home](#)**



WinSiz

This will prompt for new width and height of the graphics window (in pixels). Of course, the window may be resized in the usual way using the features of the window manager on your desktop. However, sometimes you might want to define an exact size, e.g. to make a series of snapshots, or **mpeg\_encode** needs a multiple of 8 or 16 pixels.

[Back](#)

[Home](#)



Window

Start a textual dialog to manage "virtual windows" within the original graphics window. You may split the window into 2 - 4 virtual windows of equal size (or delete them if no longer needed).

The dialog allows either horizontal or vertical arranging of those virtual windows. Each of them can be used for a different display. Selecting any mode from the **[Draw]**-menu will always use the active virtual window highlighted by a frame.

Quickly switching between virtual windows is enabled by hotkeys, i.e. pressing the corresponding key **[1]** ... **[4]**

**Back**

**Home**



LinCol

Select a color from the current palette to be used for drawing isolines. The default color is blue. A new window appears showing an array of all available colors. Click on a color field to select it. The last field shows the complete palette once more. If this is selected, the isoline color will be different corresponding to the value the isoline represents.

**Note:**

The selected color also affects the appearance of **[Net-2D]**, if the mesh has 6-point triangles or 8- or 9-point quadrilaterals, i.e. midpoints of edges. Then the (inner) lines connecting such midpoints are drawn in the same color that is selected for isolines. If the multi-color field was selected, the inner lines of the grid elements are not drawn.

**Back**

**Home**



LinLvl

This will ask in the console window for a number that indicates how many levels, i.e. how many isolines should be drawn. The default value is 25.

**[Back](#)**

**[Home](#)**



LinInt

This will ask for an interval. The default interval from 0 to 1 corresponds to the minimum and maximum of the current component of the solution. Selecting a subinterval will concentrate all the isolines to that part of the component's values. This may be useful if there is a small peak in the solution which would attract most of the isolines.

**[Back](#)**

**[Home](#)**



Scale

Define a minimum and maximum value for the currently selected component of the solution. You may enter 0, 0 for minimum and maximum to return to the default behavior, i.e. automatic scaling from the current data values.

This facility is important for the visualization of time-dependent solutions as a series of pictures. It ensures that at different time the same color means the same value.

**Back**

**Home**



Elast

This menu entry yields a textual interaction in the console window. The user may change parameters for the [Net+U] display (deformed grid). You may choose to set a relative or absolute scaling factor for deformation display.

A **relative** factor  $r$  means that the maximum displacement of all nodes is scaled to be  $r$  % of the maximum expansion of the domain, with the exception that  $r = 100$  means original deformation as supplied by the computational solution.

An **absolute** factor  $a$  means that the deformation is displayed  $a$ -times the real value, i.e.  $a = 1.0$  is the true deformation.

In this menu you may also select any two of the components of the solution (**DoFs**) to be interpreted as horizontal and vertical components of a displacement vector (components 1 and 2 are used by default).

**Back**

**Home**



Zoom

Zoom into the picture. Left-click on a first point in the window. The mouse pointer will change its shape and show a rectangle {Sometimes this rectangle badly visible because of low-contrast}. Move the mouse to a second point and left-click again to select the rectangle for zooming in. Right-clicking will terminate the interactive zoom selection and switch to the textual mode, where you can enter the (world) coordinates of a center point  $x,y$  and a radius  $r$  for the zoom area. You can enter  $x = y = r = 0$  to leave the zoomed view and return to the default.

**Note:**

The coordinate input mode is available by a hotkey: press **z** on the keyboard (while the graphics window is focused).

**Back**

**Home**



Procs

You may enter a range of processors (between 0 and **nProc** - 1). The next draw will only show the data of those processors.

**Back**

**Home**



Infos

Display a few internal statistics, e.g. the total amount of pixel data that had to be transferred.

**Back**

**Home**



Bound

Switch between two modes of displaying boundaries. The default is to draw only the boundaries (shapes) of subdomains (1 processor = 1 subdomain). The alternate is to draw all initial bounds, i.e. the coarse grid.

**[Back](#)**

**[Home](#)**



## Isolin

Switch between two modes of displaying isolines, affecting **[Isol-F]** in the **[Draw]**-submenu. The major difference is how to draw the domain's background shape before drawing the isolines.

The default is a "quick" mode where the program tries to draw and fill each subdomain as a single polygon.

Since this may be incorrect if a subdomain had a hole inside (it would be filled as the background of the domain), you may switch to a "slow" mode where each small element of the domain is filled one by one.

[Back](#)

[Home](#)



Tensor

Vector

Switch to the specified drawing mode for vector solutions. This will change the corresponding menu entry in the **[Draw]**-Submenu, where in the place of **[Net-2D]** will appear either **[Tensor]** (draw small crosses) or **[Vector]** (draw small arrows) if the vectorial solution is selected from the **[DoF]** menu.

**Back**

**Home**



**OptFil**

Select an "optimized" way to draw the coloring of the domain ( **[Filled]** ). By default, for each small triangle (or quadrilateral) a set of polygons - one for each color - is generated. If all those very small polygons have to be drawn this may take a while. There are implemented three different optimization algorithms (one *worse* than the other). These algorithms are to reduce the number of polygons by connecting them to larger ones. This may considerably reduce the pixel data which has to be transferred (by 30 ... 90 percent).

However, this will need a lot of computational time, but at least, this can be done completely parallel on each processor. It is sure that the drawing to the window will be faster if the polygons are optimized before. But it is generally not clear if the total time can be reduced. This depends on the number of nodes in the subdomains, the number of colors and the current data values.

The different modes are:

1. a simple and slow sorting algorithm that produces correct output;
2. a quick and dirty algorithm that works well in most cases, however, incorrect if the solution in one subdomain contains a "ring" of one color with another color inside; it may happen that the inner field is overwritten by the color of an outer field;
3. the same as 2. using **XOR**-mode drawing to the screen; this "repairs" the errors mentioned above. (But it is useless for postscript output!)
0. this selects explicitly **no** optimization and no re-ordering of polygons (normally they are sorted by colors). This is equivalent to the next menu item ( **[Patch]** ).

Clicking **[OptFil]** the first time will ask for the optimization mode; clicking the second time switches off the optimization.

The default case (no optimization) would lead to very large (and hence slowly loadable) postscript files if **[PSopen]** is active. Therefore, postscript output is always optimized using the last mode selected with **[OptFil]** or mode **2**, if none was selected before. Thus, to avoid time-consuming optimization and accept large postscript files instead, you must select **[OptFil]** mode **0** before.

[Back](#)

[Home](#)



Patch

Switch on or off a so-called "patch" mode. In patch mode, all elements are drawn one by one, otherwise all colors are drawn one by one.

This patch mode is helpful to draw 3D surfaces with elements sorted from back to front (the simplest hidden surface method for convex bodies). For this purpose there is also a difference in drawing grids ( **[Net-2D]** or **[Net-3D]** from the **[Draw]** menu) - in patch mode, the polygons are filled with the background color on screen or with a light gray in postscript output.

**Back**

**Home**



Quiet

Leave the interactive graphics mode immediately and switch to a batch graphics mode. You may decide, if the visualization should be executed or not in future calls to the graphics subroutines. The program repeats up to two different draws (e.g. **[Filled]** for one component and **[IsoLin]** for another one) each time. When multiple virtual windows ( **[Window]** ) are active each of them repeats its current view.

The batch mode can be finished by pressing <RETURN> or any other key while the graphics window has the focus (move the mouse into the window or click on the top bar of the window). Then the next call to **gebgraf** or **firegraf** enables the interaction menu again.

This mode is useful if the program runs in a loop where you want to see the differences in subsequent steps (time-dependency or adaptive meshes). Together with the **xxgrab**-utility you may produce a series of image files (in GIF format) in order to generate animations.

[Back](#)

[Home](#)