CHAPTER 9

Decomposition Editors

OVERVIEW

//ELLPACK uses the geometry splitting technique to parallelize the computations in the numerical solution process. The Domain Decomposition Editors partition a discrete domain into subdomains, so that the computations for the subdomain may be mapped to the processors of a parallel machine.The problem is then solved in parallel by multiple processors solving the problem on different parts of the domain. Decompositions are node-wise, that is, each node of the discrete domain belongs to exactly one subdomain, and hence, one processor.

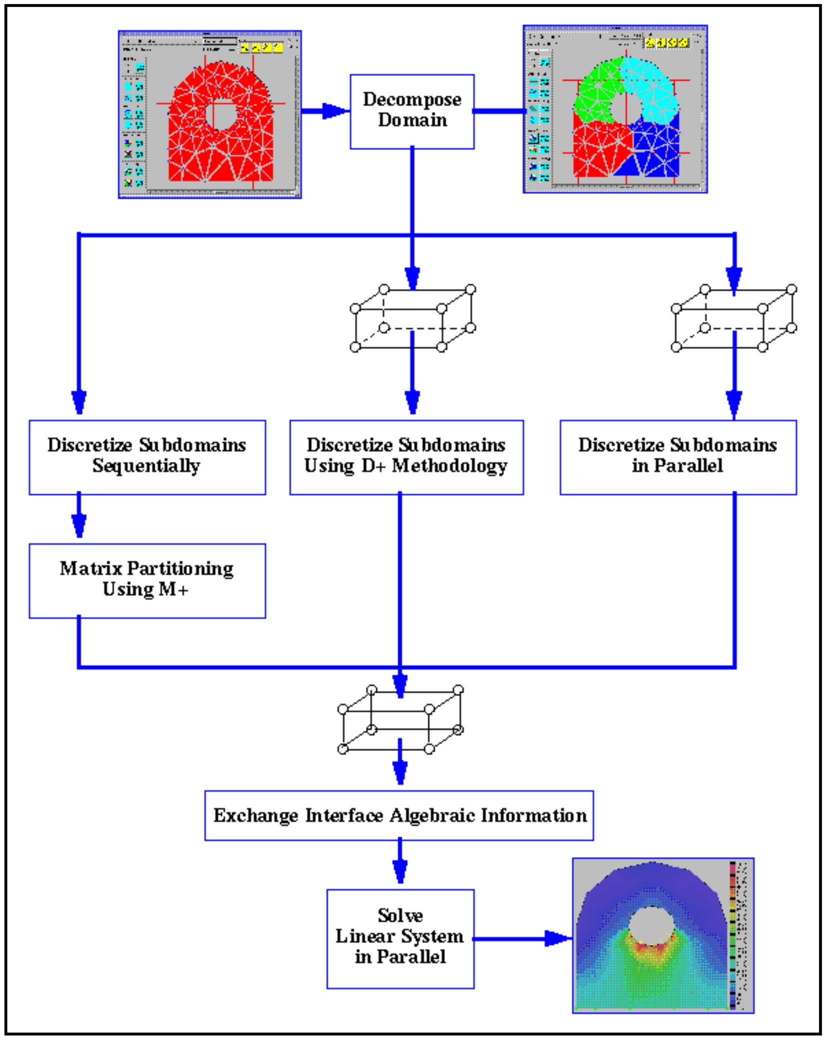
Communication between the processors occurs on the interface nodes. The definition of an interface node depends on the discretization method used in the solution scheme. For finite element meshes, interface elements are those which have member nodes destined for different processors. The interface elements contain those nodes which must communicate information from one processor to another, and nodes that communicate belong to the interface. In the case of finite difference points, interface nodes depend upon //ELLPACK’s 5-point star computational requirements.

The algorithms used to generate the partitions attempt to produce subdomains that balance the load among processors and minimize communication between processors.

The data produced by the Decomposition Editor may be used for parallel execution in one of three ways. The entire solution process may take place in parallel, including the operator discretization phase and the linear system solution phase. In this case, customized discretizers and solvers are available from the //ELLPACK library. The 2D discretizers (5-Point Star and Bi-linear FEM) and the entire ITPACK library of iterative solvers have been parallelized for various platforms (including the nCUBE2, Intel Paragon, and a network of workstations) using various libraries for communication (native, MPI, PICL, PARMACS).

The second method for using the decomposition to parallelize the PDE solution process is called D+. In this case, the discretization phase is done sequentially on the subdomains, while the solution of the linear system is processed in parallel. For D+, a //ELLPACK sequential discretizer is used, followed by any of the parallel //ITPACK solvers. D+ is available for finite element methods only.

An off-line method of executing the linear system solution phase in parallel (after first executing the discretization phase sequentially) is also available. The M+ system takes as input the geometry decomposition data produced by the Decomposition Editor, along with the linear system generated by the sequential discretizer. M+ then solves the linear system in parallel using one of the parallel solvers. This allows part of the numerical solution process to be parallelized, while taking advantage of discretizers which are not parallel. M+ has been used successfully with the sequential Vecfem discretizer and parallel solvers from the ITPACK library.

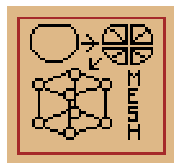


In each of the above three cases, the Decomposition Editor determines how to partition the domain into subdomains, and computations for each subdomain are handled by different processors of a parallel machine.

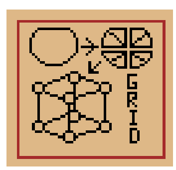
2D DECOMPOSITION EDITOR

The 2D Decomposition Editor is used to define a decomposition on the currently defined discrete 2D domain. The 2D Decomposition Editor produces a file containing a description the domain decomposition, and generates a //ELLPACK language description of the decomposition that is saved to the //ELLPACK session.

In the FEM session, the 2D Decomposition Editor is invoked by clicking the left mouse button on the following icon. To enter the 2D Decomposition Editor, the Mesh Editor must be open.

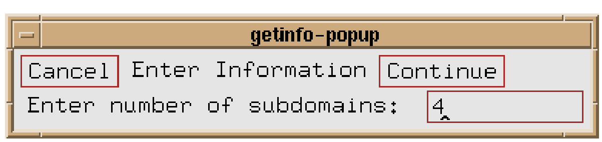


In the FDM session, the 2D Decomposition Editor can be invoked by clicking the left mouse button on the following icon. To enter the 2D Decomposition Editor, the Grid Editor must be open.

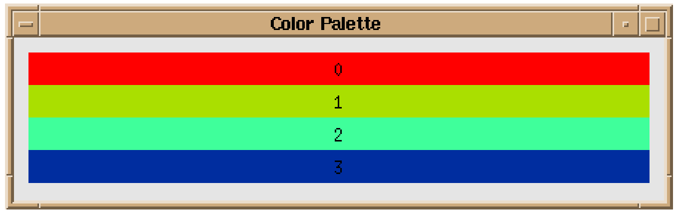


2D Decomposition Editor Windows

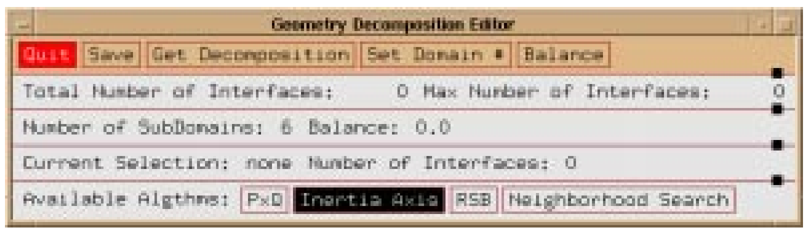
When the Decomposition Editor is invoked, the user is asked to specify the number of subdomains.



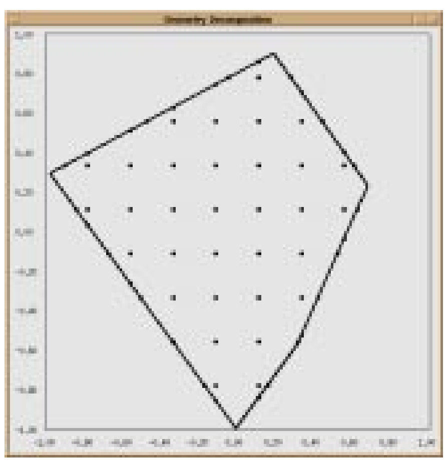
This number is equal to the number of processors that will be used to solve the //ELLPACK problem. After the number of subdomains is specified, three windows are displayed. One is the Color Palette, containing a number of color boxes equal to the number of subdomains specified.

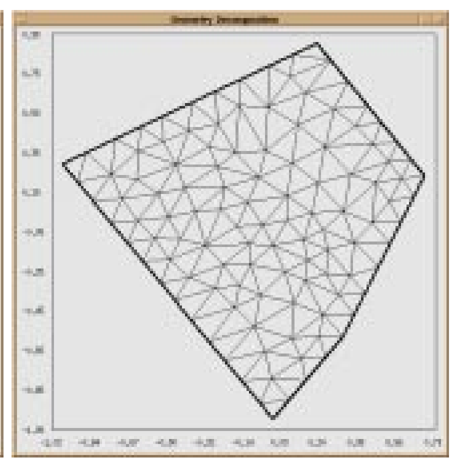


The second window is the Command Panel,



and the third window is the Drawing Window.





In the FD mode, the Drawing Window will contain the finite difference points resulting from the currently defined domain and grid. In the FEM mode, the Drawing Window shows the mesh defined on the current domain.

2D Decomposition Editor Command Panel

The 2D Decomposition Editor Command Panel contains command buttons that can be used to specify a decomposition, edit the current decomposition and save the current decomposition to the //ELLPACK session. There are also some additional fields in the Command Panel that display information about the current decomposition.

Command Panel Buttons

The buttons on the command panel are selected by clicking the left mouse button on them.

Quit

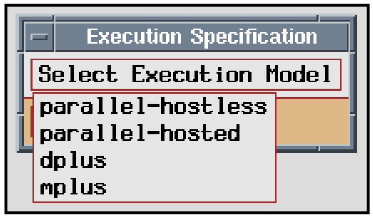
Exits the 2D Decomposition Editor. The current decomposition will not be saved to the session unless the Save button is clicked.

Save

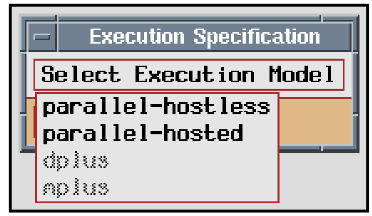
Saves the current decomposition for use in the problem solving process. Before the decomposition is saved, the user must first specify the parallel execution model which will be used to execute the problem. If the problem will be executed in a parallel hosted environment or by using M+, then the decomposition will be saved to a single file. If the problem will be executed in a parallel hostless environment or by using D+, then the decomposition will be saved to multiple files, one file per processor (ie, one file per subdomain).

All models are available for FEM sessions (hostless, hosted, M+, D+).

For FDM sessions, hosted is available for all decomposition algorithms, and hostless and hosted are available when the RuntimeGrid algorithm is used.



FEM session for all algorithms



FDM session for RuntimeGrid algorithm

Saving the decomposition involves saving the actual data in the decomposition file(s), as well as saving control information in the //ELLPACK session that can be used to access the decomposition file(s). The filename of the decomposition file is specified through a File Dialog.

Get Decomposition

Generates a decomposition according to the selected algorithm and its parameters, and then displays the domain decomposition in the Drawing Window. Note that users *must* click on this button to generate the decomposition and display the results. The selection of an algorithm and the specification of its parameters merely identifies the algorithm choice but a decomposition is not generated until the user clicks on the Get Decomposition button.

Set Domain #

Resets the number of subdomains. The dialog window displayed upon entry to the Decomposition Editor, is displayed again so that users can modify their selection for the number of subdomains. After a new number of subdomains is specified, the color palette will be updated to show the correct number of colors for the new number of subdomains, and the Drawing Window will display the original discretized domain without a decomposition.

Balance

Not implemented. In a future version, this value will be used to show the workload balance among all processors for the current decomposition.

Algorithm buttons

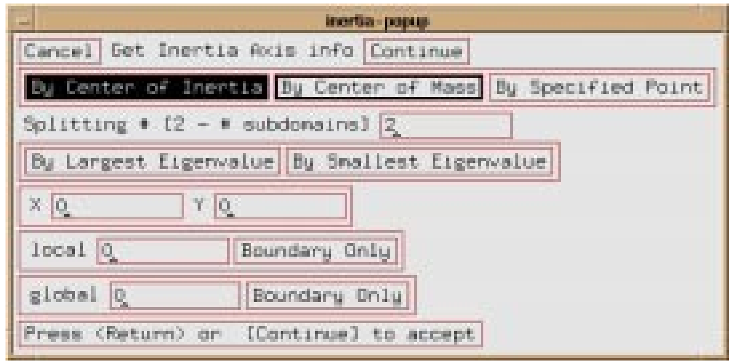
The Available Algthms buttons are used to select a new decomposition algorithm or modify the parameters for the current algorithm.

PxQ

The PxQ algorithm splits the domain along the main axis (Cartesian or Polar) after sorting the coordinates of the nodes (FD mode) or the center of mass of the elements (FEM mode). Below are the user-specified parameters for this algorithm.

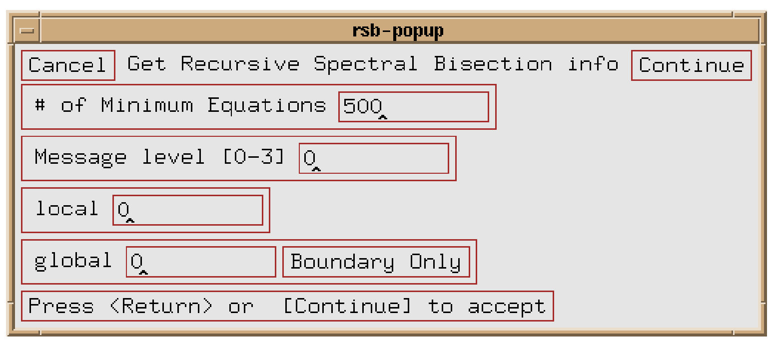
Inertia Axis

The Inertia Axis algorithm keeps splitting the domain into subdomains along the symmetry axis defined by the coordinates of the nodes (FD) or the center of mass of the elements (FEM), until the specified number of subdomains is reached. This is the default algorithm. Below are the user-specified parameters for this algorithm.



RSB

The Recursive Spectral Bisection (RSB) algorithm uses eigenvector spectral search. Nodes are visited in order of increasing eigenvector values of the Laplacian matrix of the graph. Below are the user-specified parameters for this algorithm.



Neighborhood Search

The Neighborhood Search algorithm splits the initial mesh based on the neighborhood traversal scheme. That is, the subdomains are gathered on the basis of the searching order defined. Below are the user-specified parameters for this algorithm.

By clicking on Continue in the parameter window, the user has defined the instance of the algorithm that will be used to generate the new decomposition. To apply the selected algorithm and view the new decomposition in the Display Window, the user must click on the Get Decomposition button.

2D Decomposition Editor Information Fields

These fields display information about the subdomains in the current decomposition.

Total Number of Interfaces

Identifies the total number of interface nodes, over all the subdomains, for the current domain decomposition.

Max Number of Interfaces

Lists the maximum number of interface nodes per subdomain, over all the subdomains.

Number of Subdomains

Shows the current number of subdomains.

Current Selection

Displays the color of the currently selected subdomain. To select a different subdomain, click with the left mouse button on the corresponding color box in the color palette.

Number of Interfaces

Lists the number of interfaces for the selected subdomain.

Display Window

The Display Window shows the current domain decomposition. It can also be used to manually modify the decomposition, by allowing users to interactively change the distribution of nodes per subdomain. To edit the decomposition, the user first must select a target subdomain, by selecting the corresponding color in the color palette, and then choosing the nodes that should be added to that subdomain. The new decomposition can be saved into the //ELLPACK session by selecting Save on the Command Panel.

Process for Defining a 2D Domain Decomposition

To define a decomposition for a 2D domain using this editor, the user must first specify the number of subdomains. A color palette is displayed with a number of colors equal to the number of subdomains that the user specified. The user can now either generate a decomposition by specifying one of the algorithms, or specify the decomposition by hand. The following examples show how to use the built-in algorithms to generate or modify a 2D domain decomposition.

If the user selects Get Decomposition upon entry, the default algorithm, Inertia Axis, will be applied to the user's domain. To select a decomposition, other than the default, the user can either

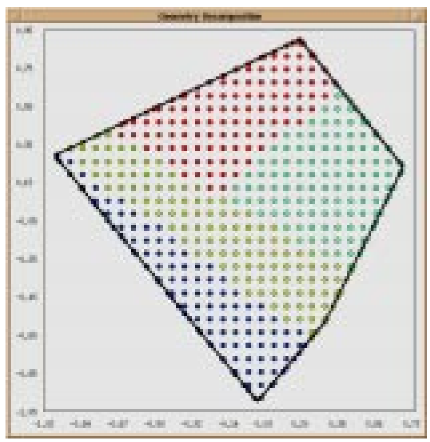
- change the default parameters to the default algorithm
- specify a different algorithm altogether

- manually edit the generated decomposition

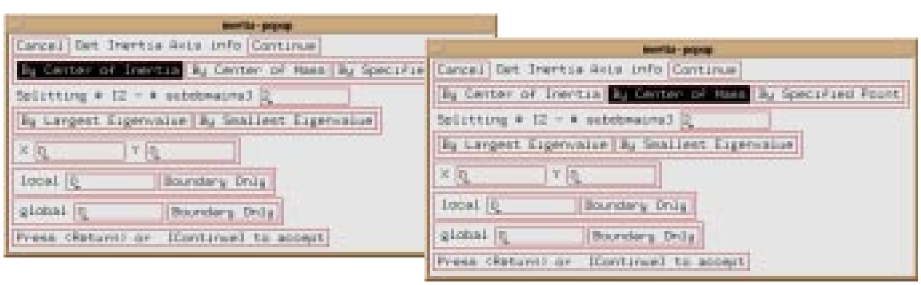
Specifying an algorithm

The following example uses the FDM session, but the discussion applies in the FEM session as well.

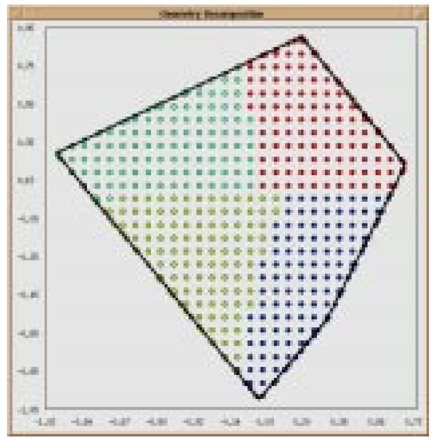
Upon entering the 2D Domain Decomposition Editor, users must specify the desired number of subdomains. If Get Decomposition is now selected, users will see the result of the default decomposition algorithm. The default parameters were used with the Inertia Axis algorithm, to define this decomposition.



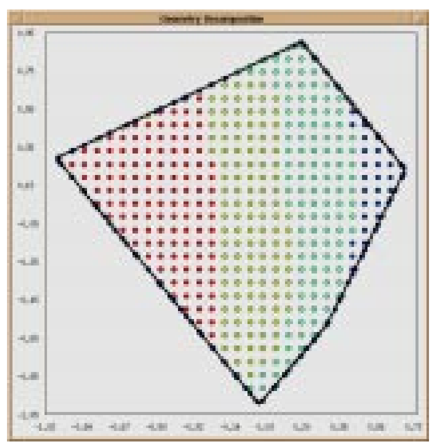
To change the paramenters for the Inertia Axis algorithm, click the left mouse button on Inertia Axis. A window displaying the input to the Inertia Axis algorithm will be displayed. Change one of the fields to



see how it affects the current decomposition. For example, the Center of Inertia field is highlighted to show that this was the property used by default to define the symmetry axis along which the subdomains were split. Click with the left mouse button on the Center of Mass field to select it, and then click on Continue. Now select Get Decomposition to see the decomposition resulting from the Initial Axis algorithm using the nodes Center of Mass to define the axis of symmetry.



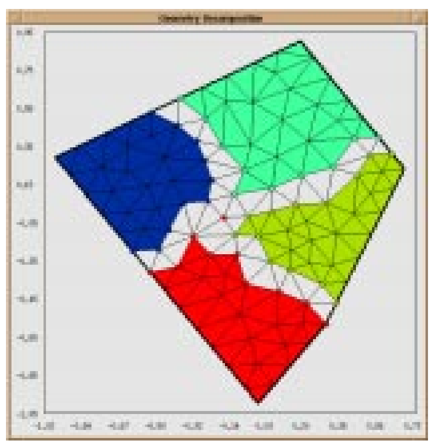
To change algorithms, click on a different algorithm button, for example PxQ. Using the default parameters for PxQ, the following decomposition will be generated with Get Decomposition.



Editing the current decomposition by hand

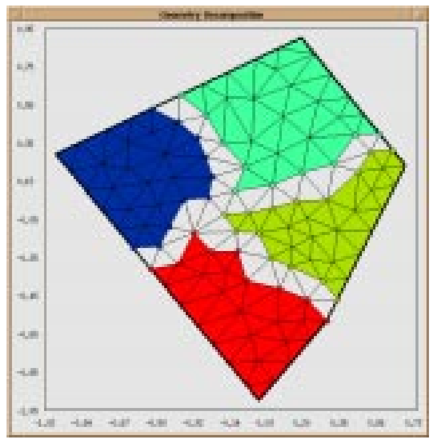
The following example uses the FEM session, but the discussion applies to the FDM session as well.

Often, the user may want to modify the current decomposition slightly, for load balancing purposes. In that case it may be easier to edit the decomposition by hand, than to trigger the input paramenters of the algorithm to achieve the same results. In this case, the user can use the drawing window to manually edit the decomposition. For example, in the following decomposition the user may want to group the node that seems disjoint from the four subgroups, into one of the four subdomains.



The four colored groups are the four different subdomains assigned to four processors. The white elements are the interface elements: elements whose nodes do not all belong to the same subdomain. Currently, there is one red node in the center of the domain that belongs to the red processor's subdomain.

To include this node in a different subdomain, first decide in which subdomain this node belongs. Select the color that corresponds to this domain, by clicking on the appropriate rectangle in the color palette. Then, notice how the node belongs to multiple elements, and select that element that is closest to the subdomain you have chosen for this node, by clicking with the left mouse button on that element. Selecting an element equates to "painting" that element the color of the selected subdomain. The node is now part of that group.

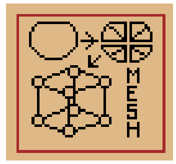


You can follow the same technique to change the assignment of elements to subdomains. Make sure you save the decomposition, by selecting Save on the Command Panel, when you are satisfied with the results.

3D DECOMPOSITION EDITOR

The 3D Decomposition Editor (FEM session only) is used to define a decomposition on the currently defined discrete 3D domain. The 3D Decomposition Editor produces a file containing a description of the domain decomposition, and generates a //ELLPACK language description of the decomposition that is saved to the //ELLPACK session.

The 3D Decomposition Editor can be invoked by clicking the left mouse button on the following icon.

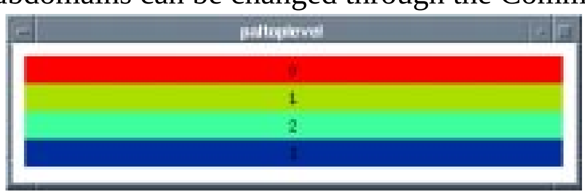


3D Decomposition Editor Areas

The 3D Decomposition Editor consists of three windows: a Command Panel, a Display Window, and a Palette. The Command Panel contains buttons and scrollbars for loading, saving, and generating a decomposition. The Display Window is used as a canvas for viewing the decomposition, and the Palette displays the colors associated with the subdomains which are currently defined for the decomposition.

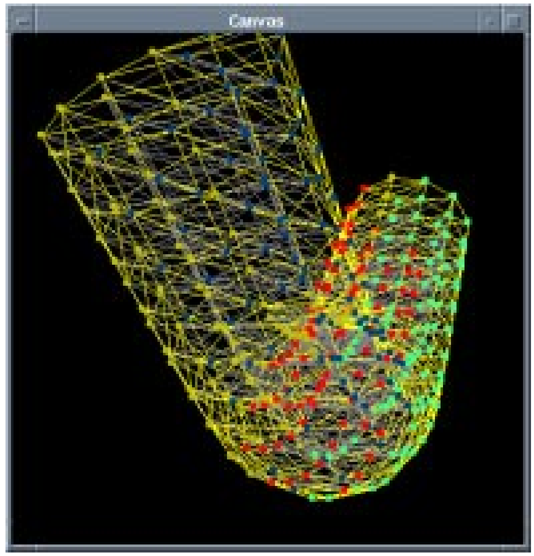
Palette Window

The Palette Window is used to identify the subdomains for the 3D mesh. Each node is assigned to a specific subdomain and the correspondence is shown on the canvas via color assignments. The number of subdomains can be changed through the Command Panel.



Display Window

The Display Window is the canvas for viewing the mesh or decomposition. Meshes and decompositions that are loaded are displayed in the canvas, and the effect of all transformations specified through the Command Panel can be viewed here.



Command Panel

The Command Panel consists of two sections: the top level buttons and the transformation scrollbars.The top level button panel contains buttons to load, edit, save, and display the 3D mesh or decomposition. The scrollbars control the view of the mesh and its decomposition in the Display Window.

Command Panel Buttons

Quit

Exits the 3D Decompostion Editor. The current decomposition will not be saved to the session unless the Save button is clicked.

Load

Loads a mesh or decomposition data file. Selecting Load causes the

File dialog to appear so that users may enter a file type and filename for loading. Users may either load a mesh file for decomposing or load a decomposition file for viewing or modifying. Two mesh file formats are supported: the //ELLPACK format and the neutral file format.

Save

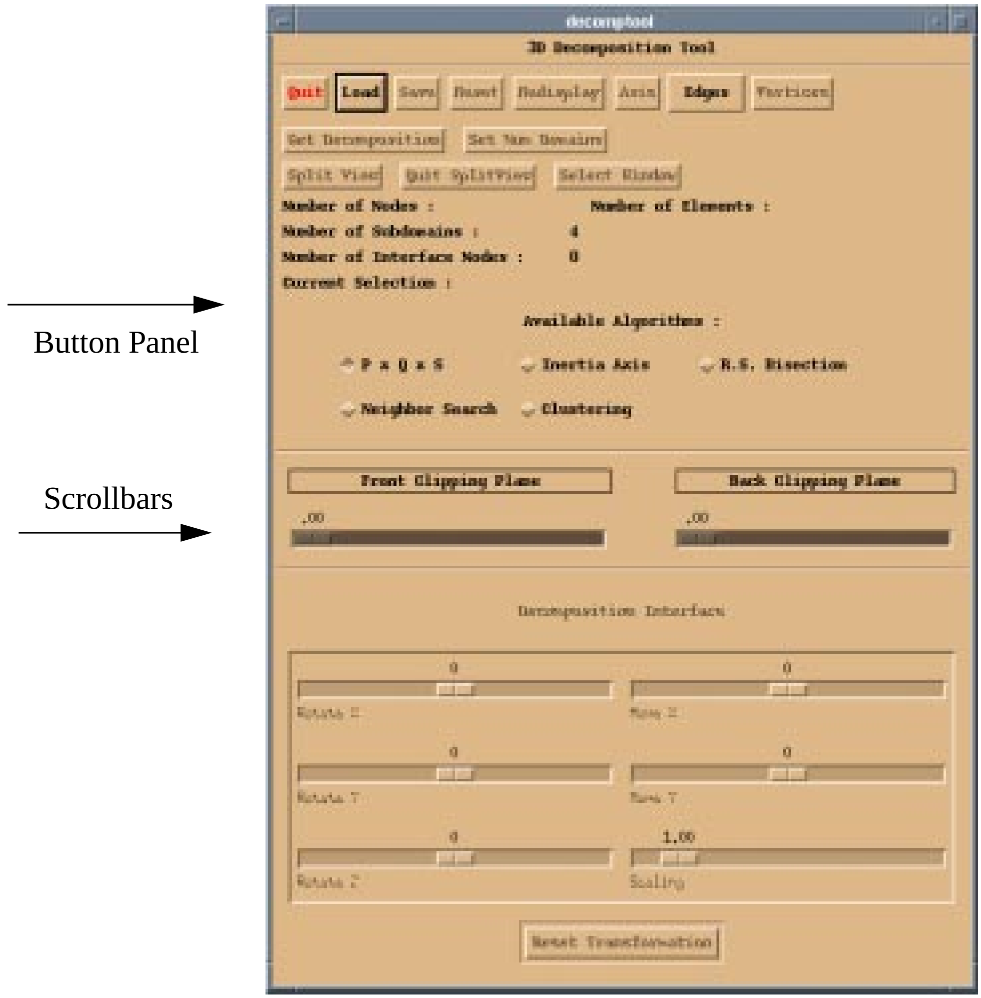
Saves the generated decomposition to the session using the //ELL-PACK language. A File dialog appears so that users may specify a filename for the saved decompostion. This dialog operates exactly as the Load File dialog.

Reset

Resets to the original decomposition.

Redisplay

Redisplays the generated decomposition on the current mesh.



Axis

Displays the coordinate axes in the Display Window in the correct orientation.

Edges

all edges : displays the entire mesh, including the edges of all elements, both interior and on the surface.

boundary edges : shows edges of only the boundary edges of the elements on the surface of the mesh.

Vertices

Displays only the vertices of the mesh.

Get Decomposition

Applies the selected decomposition algorithm with its current parameter settings and currently specified number of subdomains to the mesh. The result is shown in the Display Window.

Set New Domains

Brings up a dialog box so that a new number of subdomains can be specified.

The next three buttons apply to the subdomain display windows which can be used to view each subdomain separately. These “views” are useful for assessing the results of the decomposition algorithm.

Split View

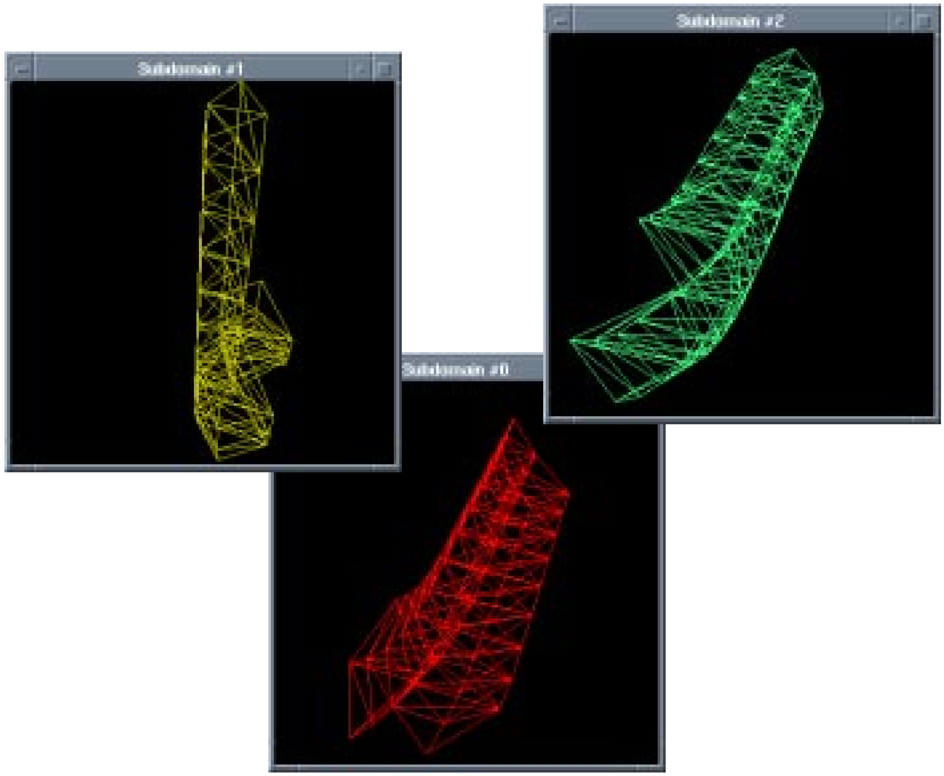
Displays each subdomain in a separate window.

Quit Split View

Closes the subdomain display windows.

Select Window

Allows the user to select a subdomain display window so that subsequent object transformation input applies to that window.



The decomposition algorithms which are available for decomposing 3D domains are the idential to those used in the 2D Decomposition Editor. The parameters required for each algorithm are exactly the same as in the 2D case. The Available Algorithms are listed below.

PxQxS

The PxQxS algorithm splits the domain along the main axis after sorting the coordinates of the center of mass of the elements.

Intertia Axis

The Inertia Axis algorithm splits the domain repeatedly along the chosen axis of symmetry.

R.S.Bisection

The Recursive Spectral Bisection algorithm uses eigenvector spectral search.

Neighbor Search

The Neighborhood Search algorithm splits the initial mesh based on the neighborhood traversal scheme.

Clustering

Not implemented.

Command Panel Information

Information about the mesh and its decomposition is contained in the Command Panel. It is updated when new meshes or decompositions are loaded, new decompositions are generated, or the subdomain selection is changed. Changes in the location of the clipping plane in the foreground/background of the mesh are displayed in the canvas.

Number of Nodes, Number of Elements

Lists the number of nodes and elements in the current mesh.

Number of Subdomains

Lists the number of subdomains currently specified.

Number of Interface Nodes

Not implemented.

Current Selection

Shows the color of the currently selected subdomain. This affects the application of transformations to the subdomain windows.

Transformation Scrollbars

The transformation scrollbars allow users to enlarge or shrink, rotate or translate the object in the canvas area.

New values in the scrollbar are set by holding the left mouse button down on the scrollbar indicator and moving it to the left or to the right.

Front Clipping Plane,Back Clipping Plane

Changes the location of the clipping plane in the foreground/background of the object displayed in the canvas.

Rotate X, Rotate Y, Rotate Z

Rotates the object in the X/Y/Z direction by the amount specified in the scroll bar. The initial rotation value is zero. Rotational amounts are in degrees.

Move X, Move Y

Translates the object in the X/Y direction by the amount specified in the scroll bar. The initial translation value is zero.

Scaling

Resizes the object in the canvas. The initial scale value is 1. Values greater than one enlarge the object; values less than one shrink the object.

Reset Transformation

Reinitializes all transformation parameters to their original values and redisplays the object in the canvas.

MORE ABOUT PARALLEL EXECUTION MODELS

	FDM	***FEM***	***save single or multiple decomposition file(s)***	
parallel-hosted	x	x	single file	See (1)
parallel-hostless	x	x	multiple files	See (2)
MPlus (M+)		x	single file	See (3)
DPlus (D+)		x	multiple files	See (4)

(1) In the ***parallel-hosted model***, two programs are generated by the //ELLPACK Execution environment, the “host” program and the “node” program. The host program reads the decomposition file and sends the decomposition data to the node processors, where the node program then runs in parallel. The nodes discretize and solve the problem according to the geometry partition received from the host, and both discretization and linear system solver phases of the solution process use parallelized code. At the conclusion of the solver phase, the nodes send the solution data back to the host which collects the output from each node and and writes a single solution output file.

(2) In the ***parallel-hostless model***, two programs are generated by the //ELLPACK Execution environment, the “node” program and the “post-processing” program. The node program is run in parallel on the node processors. Each node processor reads its own decomposition file and computes the solution for the geometry partition described in the file. Both discretization and linear system solver phases of the solution process use parallelized code, and at the conclusion of the solver phase, the nodes write their portion of the solution to an output file. A post processing program reads the node output files and collects the data into a a single solution file. Recall that although all decomposition algorithms can be used for the FEM hostless model, only the RunTime Grid algorithm can be used for the FDM hostless model.

(3) A three stage execution process is used in the ***MPlus model***. The execution of a //ELLPACK program using the MPlus model requires the //ELLPACK Execution environment, as well as a specialized MPlus execution environment. Two of the three programs are sequential, and these are created and executed in the //ELLPACK Execution environment. The middle stage is parallel, and is handled by the MPlus execution environment.

First the discretization phase is run sequentially producing the linear system matrix and writing it to a file. Second, the file containing the matrix and the (single) file containing the decomposition are read into MPlus environment, and MPlus solves the linear system in parallel. The solution is output to multiple files, one from each node processor. In the third stage, a post processing program collects the data from the node output files and generates a single solution file. This method is an "off-line" method, in that the parallel linear system solution is handled outside of the //ELLPACK Execution environment, and the required input for the MPlus solution stage is only two data files - not a //ELLPACK .e file. Some manual effort is required for the MPlus model. This is described in detail in the MPlus documentation.

(4) In the ***DPlus model***, two programs are generated by the //ELLPACK Execution environment, the "node" program and the "post-processing" program. The node program is run in parallel on the node processors. Each node processor reads its own decomposition file and computes the solution for the geometry partition described in the file. From this viewpoint, the DPlus model works in the same way as the parallel-hostless model. However, there is a difference in the *code* which is used for the discretization phase. For DPlus, the discretization code is *sequential* rather than parallel - each node performs a sequential discretization on its partition of the domain. Special interface boundary conditions are used on the nodes belonging to the subdomain interfaces. In this way, available sequential discretization codes may be used for the discretization phase, while parallel codes can still be used for the more time-consuming linear system solver phase. Fi-

nally, as in the hostless model, at the conclusion of the solver phase, the nodes write their portion of the solution to an output file. A post processing program reads the node output files and collects the data into a a single solution file.

It is essential to understand this section before using the ExecuteTool for solving parallel problems. All models are described in more detail in the chapter on the ExecuteTool environment.