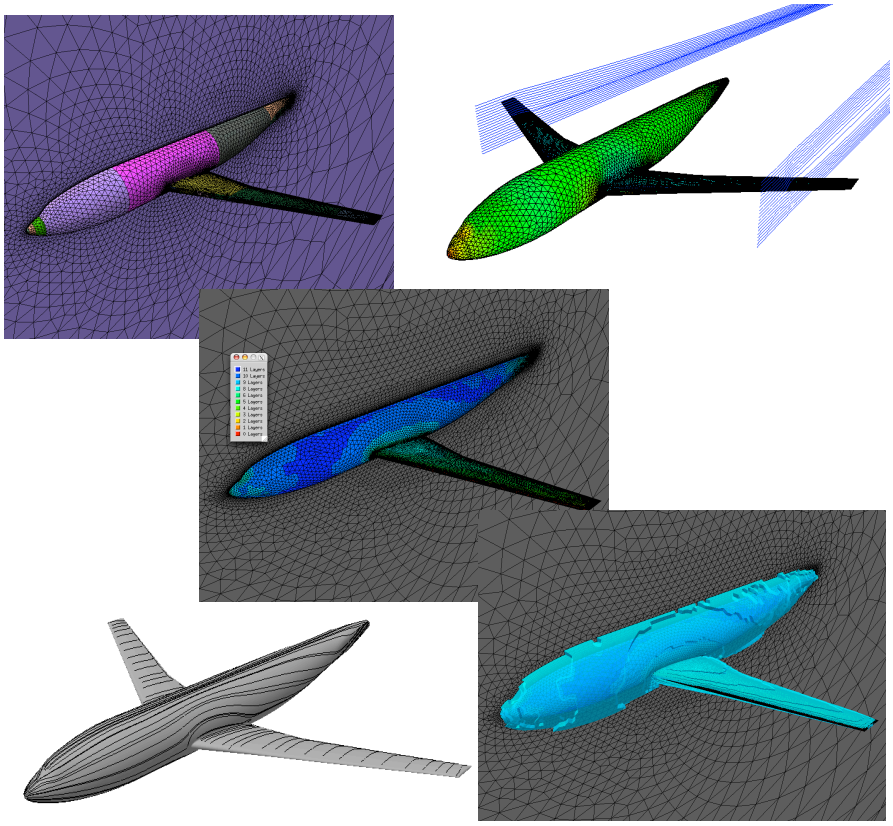
ViGYAN SimpleView and Unstructured Grid Utilities

TetrUSS Grid and Solution Viewer
Version 2.5

User's Guide

Nicholas Parlette, Edward Parlette and Javier Garriz
ViGYAN, Inc.



CONTENTS

INTRODUCTION AND CONTACT INFORMATION

The Tetrahedral Unstructured-grid Software System (TetrUSS) is a suite of computer programs developed at the NASA Langley Research Center for the purpose of performing computational fluid dynamics (CFD) analyses of aerospace problems. SimpleView is a postprocessor for TetrUSS that allows users to view unstructured grid and flow solution details. This document provides an introduction to SimpleView, along with the accompanying set of VGRID utility codes (bcFrontTest and CheckLayers). This document is intended to provide basic instructions on how each of the programs is run, and to familiarize users with the steps involved in basic grid and solution post-processing using SimpleView.

Please direct any questions, suggestions or comments on SimpleView to tetruss_support@vigyan.com.

Users are strongly encouraged to attend the TetrUSS training course conducted by ViGYAN, Inc. This course has been developed in cooperation with the primary developers of the TetrUSS system at the NASA Langley Research Center. For more information on the TetrUSS training classes, visit the ViGYAN TetrUSS training website at www.vigyan.com/tetruss_training (or contact ViGYAN at tetruss_training@vigyan.com). For more information on TetrUSS and details on the component codes, visit the official NASA TetrUSS web page at http://tetruss.larc.nasa.gov/.

ACKNOWLEDGEMENTS

This software contains routines which are based in part on the work of the Independent JPEG Group.

1 GETTING STARTED

This document provides information pertaining to SimpleView version 2.5, which was released in January 2008. SimpleView version 2.5 is intended for use with grid (.cogsg) files produced by VGRID 3.9 and 4.1, and flow solution (.flo) files produced by USM3D versions 5.3 and 6.0.

To a large extent, many of the instructions included in this document apply to earlier versions of SimpleView. Users are strongly encouraged to use the latest version of SimpleView, however, as it contains several features that greatly enhance your capability to glean useful information from the grids and flow solutions, as well as increases in over-all speed and ease-of-use.

1.1 Supported Platforms and System Requirements

All of the ViGYAN utilities described in this document (SimpleView, bcFrontTest and CheckLayers) run on the following platforms:

- Apple™ Mac OS X (10.3.9 and better, PowerPC and Intel-based CPUs)
- Red Hat™ Linux (7.3 or better)

The Linux variant of SimpleView Version 2.5 uses the Motif® toolkit for it's graphical user interface. In order to run the Linux version of SimpleView, X11 must be installed (and running) on your computer.

The Mac variant of SimpleView Version 2.5 is a native "Cocoa" application, and does not require any additional software to be installed or running on your computer.

As with any application, the speed of your computer's processors and the amount of available RAM will determine the types and amount of post-processing that you will be able to perform (in a timely fashion) on a particular dataset (as measured by the grid size). Additionally, the performance you experience will also be dependent on the capability of the graphics cards installed in the computers on which you run SimpleView. Throughout this document, we've tried to provide ratings to help identify some of the more memory and CPU-intensive viewing options and diagnostic operations available in SimpleView and the grid diagnostic codes (bcFrontTest and CheckLayers).

There are no differences in functionality when running SimpleView on any of the supported platforms. There are only minor differences in the look and feel of the program; in the course of this document, we'll highlight these differences when necessary.

1.2 Installing SimpleView

The Linux variant of SimpleView is supplied as an executable file that you can save in any location on your computer; to run it, you simply type the name of the executable file (including the full path to the file's location) on a command line.

The Mac variant of SimpleView variant is supplied as a "package" file which you double-click to initiate the installation process; follow the on-screen prompts to complete the process. By default, the SimpleView application is placed in the /Applications directory. You can then simply double-click on the SimpleView program icon to run it; note that as with any Mac application, you can drag the SimpleView program icon into the dock for easier access.

1.3 Unlocking Your Copy of SimpleView

SimpleView is distributed with node-locking enabled. This means that you will only be able to run SimpleView on a machine that has a suitable machine-specific "key file." The key file contains a single line with a string of characters; this string must be obtained from the TetrUSS Team at ViGYAN (reachable via e-mail at tetruss_support@vigyan.com).

What information do I need to provide in order to obtain the key file?

In order to generate the key string for a particular computer, you will need to determine a machine-specific identifier (the Host ID or the Serial Number, depending on the operating system of the computer), and provide that identifier to the ViGYAN TetrUSS Team.

If you are running on a Linux computer, simply type the command `hostid` in a terminal window. You can also redirect the output of the `hostid` command to a file, as follows:

```
hostid > hostid_file
```

You can then send the *hostid_file* to the ViGYAN TetrUSS Team via e-mail, and they will provide you with the required key string.

If you are running on a Mac, click on the Apple icon on the upper left-hand corner of the screen and choose "About This Mac." A panel like the one shown in Figure 1.2.1 will then appear.

Figure 1.2.1 "About This Mac" panel (default appearance)

If you click on the line underneath the "Mac OS X" (the area where the OS version number is displayed by default), the display will change to reflect the OS build number and (if you click again) the Serial Number of your computer, as shown in Figure 1.2.2.

Figure 1.2.2 "About This Mac" panel (with Serial Number displayed)

Alternatively, you can click on the "More Info" button (shown in Figures 1.2.1 and 1.2.2), and the System Profiler will display the Serial Number, as shown in Figure 1.2.3.

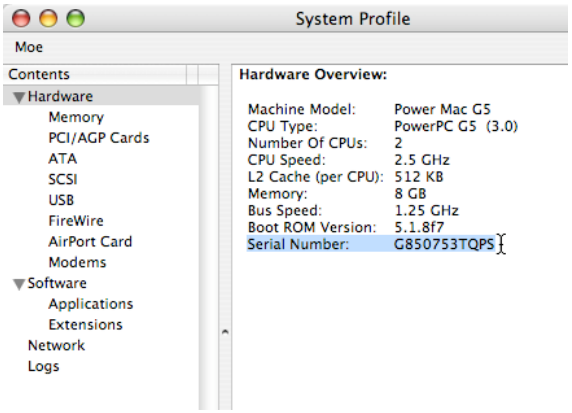


Figure 1.2.3 System Profiler panel (with Serial Number displayed)

You can then send the serial number to the ViGYAN TetrUSS Team via e-mail, and they will provide you with the required key string.

Where should I place the key file?

Once the ViGYAN TetrUSS team receives your host id or serial number, they will generate the key string and send you a "key file" containing the key string. Precisely what you do with the key file once you have received it depends on the variant of SimpleView you will be running, as described below.

If you are running the Linux variant of SimpleView : you are free to place the key file in any location on your computer, so long as you specify where it is so that SimpleView will be able to locate it. You do this by including a line which specifies the location of the key file in either the .cshrc file (if you are using C shell as your default shell), or the .bashrc file (if you are using bash as your default shell), as described below.

If you are unsure what shell is the default shell on the computer you will be running SimpleView, the following section should help.

In order to determine what the default shell is on the computer on which you wish to run SimpleView, type the command `PRINTENV` and hit the return key. Among the results returned will be one line like

`SHELL=/bin/bash` (this indicates that C-shell is the default shell)

or

`SHELL=/bin/csh` (this indicates that bash is the default shell)

If you are using C shell as your default shell, you should include the following line in your .cshrc file:

`setenv SVKEYFILE /path to sv_keyfile/`

where *`/path to sv_keyfile/`* corresponds to the complete path to the directory where you have placed the key file provided you by ViGYAN.

For example, if you have placed the key file called "sv_keyfile_1" in the directory

/usr/local/bin

then the line to be added to your .cshrc file will appear as

`setenv SVKEYFILE /usr/local/bin/sv_keyfile_1`

If you are using bash as your default shell, you should include the following line in your .bashrc file:

`export SVKEYFILE=/path to sv_keyfile/`

where *`/path to sv_keyfile/`* corresponds to the complete path to the directory where you have placed the key file provided you by ViGYAN.

For example, if you have placed the key file called "sv_keyfile_1" in the directory

/usr/local/bin

then the line to be added to your .bashrc file will appear as

`export SVKEYFILE=/usr/local/bin/sv_keyfile_1`

Remember that you may have to "source" the .bashrc or .cshrc files before the changes you have made to them take effect. To do this, simply type the following at a terminal prompt:

```
source .bashrc
```

or

```
source .cshrc
```

If you are running the Mac variant of SimpleView : the very first time that you run SimpleView, a pop-up box (shown in Figure 1.2.4) will appear, prompting you to enter the key string.

Once you've entered the proper key string (and clicked the "ok" button), SimpleView saves the key in it's internal application structure. From that point on, you will not be prompted for the key when you run SimpleView.

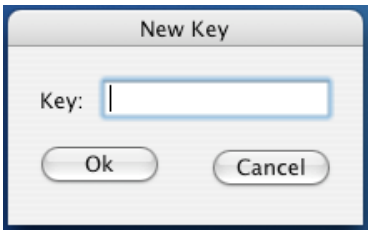


Figure 1.2.4 Recommended directory structure for running VGRID and POSTGRID.

Note : Mac users upgrading to SimpleView Version 2.5 from SimpleView Version 2.2.3 will have to enter their existing key string into the prompt shown in Figure 1.2.4 the very first time that they run SimpleView Version 2.5.

1.4 Recommended Directory Structure

As described in the VGRID User's Guide, VGRID users are encouraged to set up a particular directory structure in which to run the different stages of grid generation. This structure (shown in Figure 1.4.1) minimizes the chances of losing any intermediate grid data that might be useful in trouble-shooting grid-generation problems. This directory structure also allows users to run the grid diagnostic utilities bcFrontTest and CheckLayers at the different, designated stages of the process (and display the appropriate results using SimpleView).

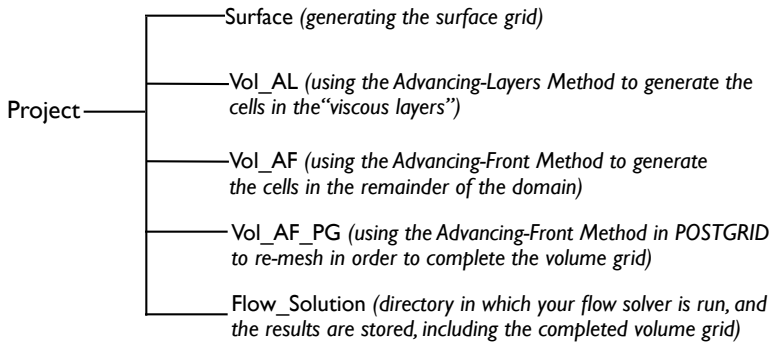


Figure 1.4.1 Recommended directory structure for running VGRID and POSTGRID.

We'll be referring to this directory structure throughout this document. Specifically, we'll highlight the kinds of operations that you can perform and the types of data you can view using SimpleView in each of the directories; similarly, we'll point out the directories in which you should run each of the grid diagnostic utilities bcFrontTest and CheckLayers.

Note that the directory structure illustrated in Figure 1.4.1 corresponds to the case where a viscous-type grid has been generated; in the case of an inviscid-type grid, the "Vol_AL" stage (and corresponding directory) are not needed.

It should also be noted that this document assumes that the directories listed in Figure 1.4.1 contain the files enumerated below. Additional files pertaining to grid-generation inputs (for example, the GridTool restart and VGRID .d3m files) as well as the flow solution input and output files (such as the .iface and .inpt USM3D input files) can also be present in those directories; the files listed below are the only ones that are actually used by SimpleView (and the grid utilities bcFrontTest and CheckLayers). In the list of files below, the prefix *project* refers to the "project name" you have chosen and assigned in GridTool; please refer to the VGRID User's Guide and the USM3D Training Guide for details on the contents of each of the files listed.

- The "Surface" directory contains the results of having run VGRID to generate the surface mesh. As such, at the completion of that VGRID run, this directory ought to contain the following files :

1. project.mapbc
2. project. bc
3. project.front
4. project.cogsg

- The "Vol_AL" directory contains the results of having run VGRID to generate the "advancing layers" (also called the "viscous layers") portion of the volume mesh. As such, at the completion of that VGRID run, this directory ought to contain the following files:

1. project.mapbc
2. project.bc
3. project.front
4. project.cogsg
5. project.poin1

- The "Vol_AF" directory contains the results of having run VGRID to generate the "advancing front" (also called the "inviscid") portion of the volume mesh. As such, at the completion of that VGRID run, this directory ought to contain the following files:

1. project.mapbc
2. project.bc
3. project.front
4. project.cogsg
5. project.poin1

- The "Vol_AF_PG" directory contains the results of having run POSTGRID to complete the volume grid using the "advancing front"method. As such, at the completion of that POSTGRID run, this directory ought to contain the following files:

1. project.mapbc
2. project.bc
3. project.front

4. project.cogsg
5. project.poin1

- The "Flow_Solution" directory contains the results of having run USM3D to generate a flow solution using the completed volume mesh. As such, at the completion of that USM3D run, this directory ought to contain the following files:

1. project.mapbc
2. project.bc
3. project.front
4. project.cogsg
5. project.poin1
6. project.flo
7. cells. 1storder
8. cells.problem

   (note: the "cells. 1storder" and "cells.problem" files are diagnostic files written by USM3D only as needed and requested)

2 USING *SIMPLEVIEW*

2.1 Background and Nomenclature

At each stage of grid generation (as described in section 1.4), VGRID writes out files containing all the information needed to fully describe the grid that has been generated up to that point. Similarly, the flow solver USM3D writes out files containing flow solution variables at the completion of a run (and at user-specified intervals). SimpleView reads in several of those files in order to visualize the unstructured grids generated by VGRID (at the various stages of grid generation), as well as the flow solutions generated on those grids by the the flow solver USM3D. In this section, we provide brief definitions for the terminology used to describe some of these files--specifically, the files that SimpleView reads in, displays and allows you to manipulate--and short descriptions of the file contents. We also provide definitions for the terms used within the SimpleView user interface itself.

As discussed in the VGRID User's Guide, VGRID employs two algorithms to generate unstructured tetrahedral volume grids :

1. the "advancing front" algorithm is used to generate the triangles comprising the surface grid (located on the surface patches defining the configuration and outer boundaries), as well as to generate the tetrahedral cells in the "inviscid portion" of the domain.

2. the "advancing layers" algorithm is used to generate the (more ordered) tetrahedral cells in the "viscous layers" region of the grid ; the remainder of the volume is then filled by tetrahedral cells generated by the "advancing front" algorithm.

The "Current Front," and the VGRID .front file :

At any stage of the grid generation, the current "front" refers to the set of points (generated by either algorithm in VGRID) comprising the triangular faces that have not yet been connected to other triangular faces to form complete tetrahedra. As such, the front continually changes as the volume grid is being generated. At the end of the surface mesh generation stage (the beginning of the volume grid generation stage), the front is composed of the points comprising the triangular faces making up the surface mesh. At the completion of the volume grid generation (after POSTGRID has managed to complete the grid), there will no longer be a "front," as every

triangular face will be part of a completed tetrahedron. VGRID writes out a ".front" file at the completion of each VGRID run; this file contains the coordinates of each point on the current front, along with connectivity information specifying how each point is connected to form each of the triangular faces on the current front. Although you can read the .front file into SimpleView at any stage of the grid generation (to see how far into the field the grid has progressed), you will most commonly read in the .front file at one specific stage of the grid generation process--at the completion of the advancing-layers stage, so that SimpleView can perform the automated check for "zero-layer faces" (described below); this is performed in the "Vol_AL" directory described in section `1.4`.

The VGRID .cogsg file :

After running VGRID at any stage of the grid generation, VGRID writes a .cogsg file, which contains the cartesian coordinates of every grid point that has been generated, along with the connectivity information specifying the points comprising each triangular face and each tetrahedron in the field. SimpleView allows you the option of reading in the entire volume grid, or just the surface .grid. Reading in the surface grid file will prompt SimpleView to read in only that portion of the .cogsg file corresponding to the surface mesh; that is, SimpleView will only read in the coordinates and connectivity of the triangular faces on the surface of the configuration (including the outer boundary). If you then read in a USM3D .flo file (described below), SimpleView will only read in the solution quantities corresponding to the surface grid; choosing this "surface only" option reduces the amount of storage required by SimpleView while still allowing you to examine the surface grid and flow solution on the surface--including generating oil flows and extracting flow-derived surface quantities (surface C_p, for example) at locations of interest. As such, this is the suggested mode of operation if you are running on a relatively "thin" client (a laptop instead of a workstation, for example), especially when working on large datasets/grids. If, however, you are interested in examining the variation of grid or flow solution characteristics in the field--for example, to examine the variation of flow quantities on planar cuts through the domain, generating streamlines, iso-surfaces, or simply looking at the tetrahedra crossing a specified plane (a "crinkle cut")--you must read in the "full" .cogsg file (by loading the volume grid). Reading in the .flo file after having loaded the complete volume grid file will prompt SimpleView to read in the entire .flo file.

Boundary Conditions–the .bc and .mapbc files :

GridTool (the graphical preprocessor to VGRID) writes out several files that are used as input files to VGRID; the .mapbc file is one of them. This file lists the boundary condition, family name, and surface number(s) you have assigned to each surface patch. When you load a .cogsg file (either the "full" or "surface") into SimpleView, the .mapbc file is automatically read in as well.

Once VGRID generates a surface mesh, it writes out a .bc file; this file contains a list of the surface triangles comprising the surface mesh, along with the patch number that each surface triangle lies on. When you load a grid (either the "volume" or "surface" grid) into SimpleView, the .bc file is automatically read in as well.

The .mapbc and .bc files ultimately become input files to the flow solver USM3D; reading them into SimpleView allows you to display the surface triangles along with their associated boundary conditions, allowing you to check the boundary conditions that have been assigned to each patch before actually running the flow solver. If it is determined that the wrong boundary condition has been associated with a particular surface patch, changes to the boundary conditions can be made in SimpleView, and a new .mapbc written out (see the "Viewing and Editing Boundary Conditions" section).

The USM3D flow solution (.flo) file:

This file contains the values of the flowfield data at each grid point, specified in terms of five conserved quantities (the density, the three components of the momentum, and the total energy), and is written at the completion of a USM3D run. As discussed in the description of the "VGRID .cogsg file" earlier in this section, SimpleView will read the entire .flo file if it is loaded into SimpleView after having loaded the volume grid; if the surface grid is loaded, SimpleView will read only the portion of the .flo corresponding to the surface grid (including the outer boundaries of the domain).

Cell: In SimpleView , a "cell" refers to a tetrahedron generated by VGRID (using either algorithm described at the beginning of this section).

Face: In SimpleView, a "face" refers to a triangle (i.e., three grid points comprising a face of a tetrahedron), located either on the surface mesh or the "current front."

Node: In SimpleView, a "node" refers to a grid point generated by VGRID (using either algorithm); these can be on the surface or in the field/volume.

Streamline: In SimpleView, a "streamline" refers to a curve in space representing the path taken by a fluid particle released from a user-specified point in the domain. The tangent to the curve at any point is in the direction of the velocity vector at that point. Streamlines are useful for illustrating the nature of the velocity field at user-selected regions of the domain (for example, to visualize wing-tip vortices).

Oilflow: In SimpleView, an "oilflow" refers to a streamline that is confined to the surface on which it is placed. As its name implies, an "oilflow" in SimpleView is the numerical analog of the type of pattern that would be obtained by coating the surface of a configuration with oil and running the configuration in a wind-tunnel (or in free air). These "surface-only" streamlines can be generated on surface triangles which lie on patches with any of the "solid surface" boundary conditions (inviscid surface, viscous surface, "blunt base" or "wake" boundary conditions), as well as surface triangles on patches with the "reflection plane" boundary condition. Oilflows are useful for illustrating the nature of the velocity field on user-selected regions of the surface (for example, to visualize regions of recirculation, flow separation and attachment).

Surface Patches : In SimpleView, surface patches refer to the closed, three-dimensional polygons which comprise the configuration and the outer boundary of the domain. As described in the GridTool and VGRID User Guides, boundary conditions are applied on each surface patch, and VGRID generates the surface mesh on a patch-by-patch basis. Patch information is loaded into SimpleView via the .bc and .mapbc files, which are read automatically by SimpleView once a volume or surface grid is loaded. SimpleView can then display the patches, and the boundary conditions associated with each.

Zero-Layer Faces : In SimpleView, a "zero-layer face" refers to a surface triangle (i.e., a triangle on the surface mesh) which is supposed to have "advancing-layer" type cells above it, but which, for some reason, does not. These "zero-layer faces" can lead to problems in completing the grid, or (if the grid can be closed) locally degraded solution accuracy. As such, it is always important to know if the grid you are in the process of generating contains such faces, and users are encouraged to check for the existence of any such faces **at the completion of the advancing-layers generation stage**–that is, in the "Vol_AL" directory shown in Figure 1.4.1. SimpleView

automatically determines the presence of "zero-layer faces" once the .surface mesh and the "current front" files are read in, as discussed in section 2.4 of this document.

Iso Surface: In SimpleView, an iso-surface refers to a surface on which the value of any (user-specified) data quantity is a constant (user-specified) value. For example, it is common to show surfaces of constant pressure or Mach number.

2.2 Running SimpleView

Once SimpleView is launched, the program opens a windows as shown in Figure 2.2.1.

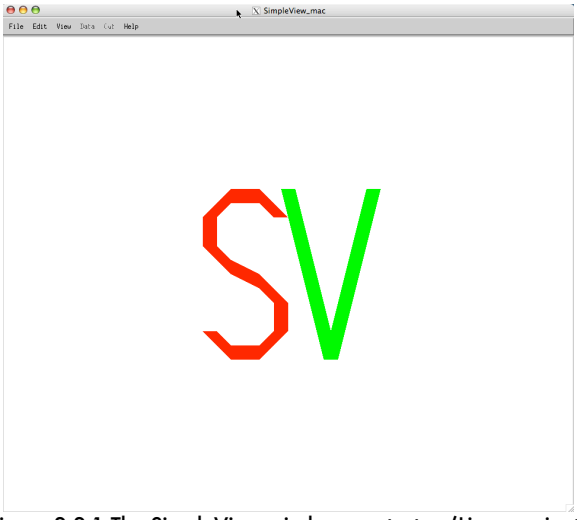


Figure 2.2.1 The SimpleView window on startup (Linux variant)

A number of "Hot Keys" (keyboard shortcuts) are provided in SimpleView. These allow users to accomplish specific tasks using the keyboard (instead of clicking on a button with the mouse). To activate each of the Hot Keys, place the cursor over the active display window and type the key (or key sequence). To find out which hotkeys are available, look in the Hotkey Settings dialog box, found in the "Edit" panel; that panel also allows you to change the default alphanumeric hotkeys found there. The Mac version allows the user to define hot keys for a very wide array of commands--in fact, the user can create hotkeys for just about every command that is available via the drop-down menus that are accessible from any of the panels that appear at the top of the screen when running SimpleView (for example, the "File," "Edit," "View," "Data" and "Cut" panels).

Loading the Grid into SimpleView

In order to view a grid in SimpleView, you must first "load" (i.e., read in) the grid. This is performed by selecting one of the "Load " options from the Load submenu of the File panel, as shown in Figure 2.2.2, for both the Linux and Mac variants of SimpleView.

(a)

(b)

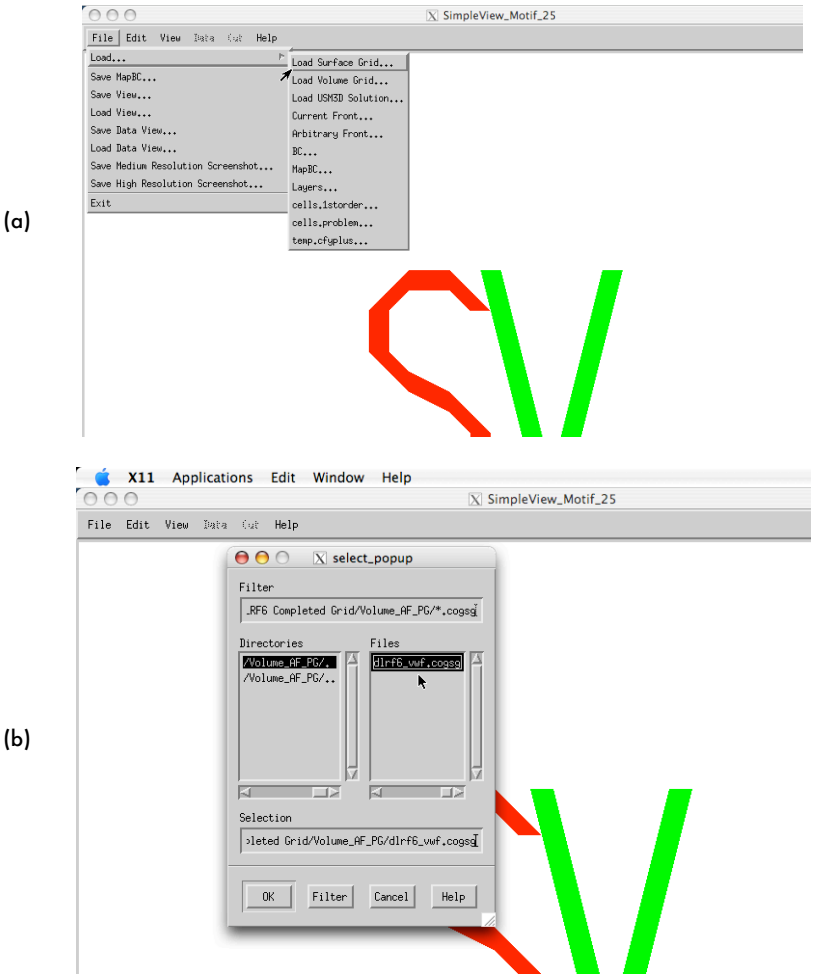
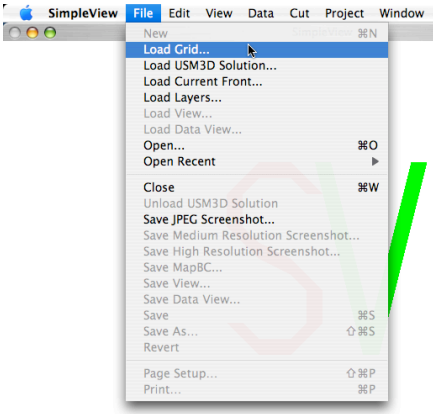
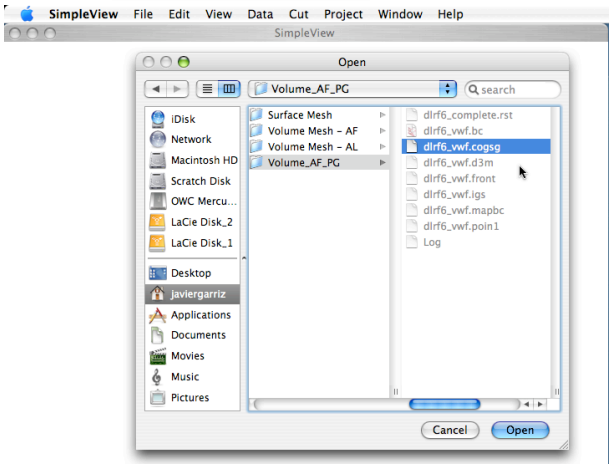


Figure 2.2.2 "Load Grid" options in the Linux variant of SimpleView : (a) the Load options, (b) the file selector

(c)



(d)



(e)

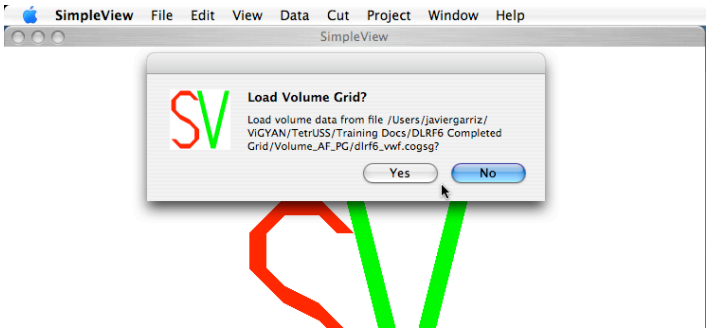


Figure 2.2.2 "Load Grid" options in the Mac variant of SimpleView :
(c) the Load option, (d) the file selector, and (e) the surface or volume grid selector

In theLinux variant, you choose either the "Load Surface Grid" or "Load Volume Grid" options (as shown in Figure 2.2.2 a); you then navigate to the directory containing the grid. In the Mac version, you choose the generic "Load Grid" option (as shown in Figure 2.2.2 c), and navigate to the directory containing your grid (as shown in Figure 2.2.2 d). As shown in Figures 2.2.2 b, and 2.2.2 d, SimpleView will display only the grid files in the file selector box (all other files will not be shown, or be grayed out, indicating that they are not selectable as grid files). The Mac version will then prompt you to specify whether you want to read the entire grid file (including any portions of it corresponding to the volume grid), or only the surface mesh portion of the grid (as shown in Figure 2.2.2 e). As mentioned earlier, reading in the surface grid file will prompt SimpleView to read in only that portion of the .cogsg file corresponding to the surface mesh; that is, SimpleView will only read in the coordinates and connectivity of the triangular faces on the surface of the configuration (including the outer boundary), even though the grid file may contain additional data corresponding to the coordinates of the cells in the volume grid. If you then read in a USM3D .flo file (described below), SimpleView will only read in the solution quantities corresponding to the surface grid; choosing this "surface only" option reduces the amount of storage required by SimpleView while still allowing you to examine the surface grid and flow solution on the surface--including generating oil flows and extracting flow-derived surface quantities (surface C_p, for example) at locations of interest. As such, this is the suggested mode of operation if you are running on a relatively "thin" client (a laptop instead of a workstation, for example), especially when working on large datasets/grids. If, however, you are interested in examining the variation of grid or flow solution characteristics in the field--for example, to examine the variation of flow quantities on planar cuts through the domain, generating streamlines, iso-surfaces, or simply looking at the tetrahedra crossing a specified plane (a "crinkle cut")--you must load the "full" volume grid. Reading in the .flo file after having loaded the complete volume grid file will prompt SimpleView to read in the entire .flo file.

Regardless of which Load option you choose (surface or volume), SimpleView will read in the .cogsg file, as well as the associated .bc and .mapbc files corresponding to the project.

It should be noted that SimpleView version 2.5 is capable of reading in the "multi-segment" .cogsg file format that is an available output option in VGRID/POSTGRID version 4.1.

Note : you can also choose to load the .front file (instead of the .cogsg file) by choosing the "Load Current Front" option from the Load submenu of the File panel. However, because the contents of the .front file change during the various stages of grid generation (as discussed in section 2.1), reading in the current front will not necessarily load a surface grid (unless you are running SimpleView in the "Surface" directory, as discussed in Section 2.1). As it's name implies, the "Load Current Front" option will load the contents of the .front file as a set of triangles corresponding to the location of the front at that particular stage of grid generation. Furthermore, if you load a front file instead of a surface or volume grid (.cogsg) file, the corresponding .bc and .mapbc files will **not** be loaded into SimpleView. Consequently, you will not be able to view the surface patches, or edit the boundary conditions and/or family names (as discussed later on). Loading the front file is typically only performed at the completion of the advancing-layers stage of grid generation, in order to check for "zero-layer faces--the procedure for doing this will be described later on.

Moving Around in SimpleView

SimpleView is designed to work with a three-button mouse, and uses the same "PLOT3D style" manipulation/movement controls as GridTool, VGRID and POSTGRID. Moving the mouse while holding the left mouse button rotates about the X and Y axes. Moving the mouse while holding the right mouse button pans. Moving the mouse left and right while holding the middle mouse button rotates about the Z axis. Moving the mouse up and down while holding the middle mouse button zooms out and in. respectively.

To see the center of rotation (the point about which rotations are currently occurring), turn on the axis display by either selecting the "Axis" option from the View Menu or hitting the "display axis" hotkey ("x" by default).

The left mouse button is used for displaying the choices within each of the menu items listed in the main selection panel; that is, clicking with the left mouse button on the "File," "Edit," "View," "Data" or "Cut" menus displays additional options, which are also selected using the left mouse button, as shown in Figure 2.2.2.

Selecting Objects in SimpleView

In the course of examining a grid or flow solution, it is useful to be able to select certain types of objects, in order to examine their properties, or to simply reposition the center of rotation. In

SimpleView, you can select any face on the surface mesh (or the current front) by placing your cursor over the face and double clicking the left mouse button; this highlights the face under the cursor and sets the center of rotation to be on the center of the selected face. Selecting the "Select Face" option from the "Select" submenu of the "Edit" panel opens the Face Select dialog. Enter the number of a face into the text field and click OK to set the center of rotation to that face. The "Select Node" option works the same way, only it selects a node rather than a face. All selected objects are displayed in the Select Color (purple by default; the color can be changed from the Color submenu, found in the "View" panel). When a surface mesh face is selected, information about that face is shown in a box that appears in the lower left corner of the SimpleView screen ; this is shown in Figure 2.2.3.

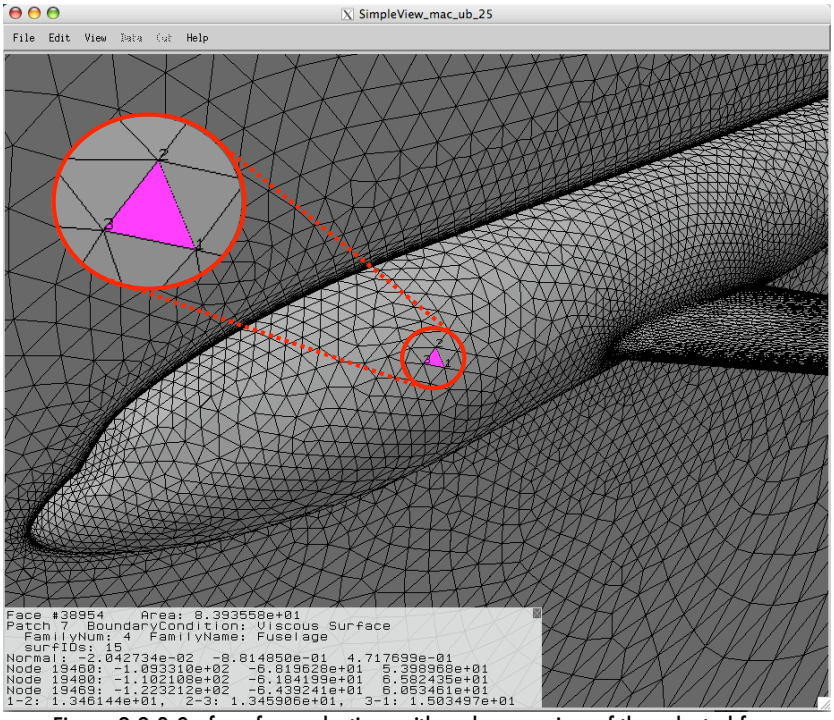


Figure 2.2.3 Surface face selection, with a close-up view of the selected face

When a face on the surface mesh is selected (as in the figure), the box that appears in the lower left corner of the screen displays the following information:

- the face number
- the area of the selected face

- the patch number of the surface patch on which the selected face lies
- the USM3D boundary condition that is currently associated with the patch on which the selected face lies
- the family name that is currently associated with the patch on which the selected face lies
- a "surface ID" corresponding to the underlying surface definition which has been associated with the patch (if any)
- the components of a unit normal vector for the selected face
- the node numbers and coordinates of each of the nodes comprising the face
- the edge lengths of each side of the selected face; these are listed for edges "1-2," "2-3" and "3-1," corresponding to the edges shown on the face (see the inset in Figure 2.2.3)

When a node is selected (via the "Select Node" option from the "Select" submenu of the "Edit" panel), the box that appears in the lower left corner of the screen shows the node number and coordinates of that node.

If you have loaded a complete volume mesh into SimpleView, you can also display specific cells. SimpleView allows you to select the cell to be displayed by hitting the Cell Select hotkey ("c" by default on the Linux version) and entering the cell number in the pop-up menu that appears. The center of rotation will be set to the first node in the cell. The selected cell will not be displayed in the Select Color, but will instead be displayed as a tetrahedron with red, green, blue, and yellow faces, as shown in Figure 2.2.4.

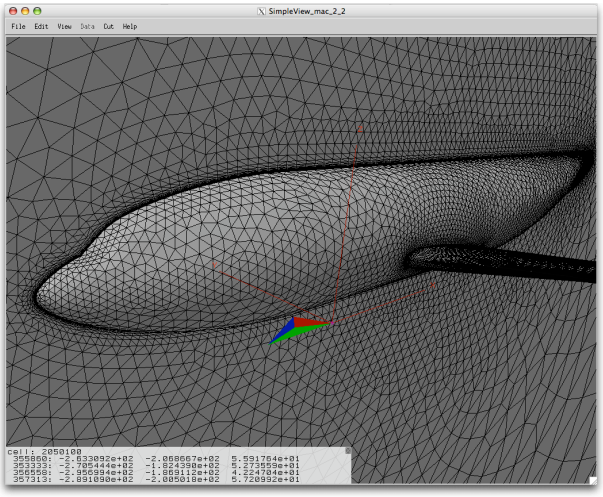


Figure 2.2.4 Cell selection

At the present time, this method only allows one cell to be displayed at a time. The cell number and coordinates of each node comprising the cell will be displayed in the dialog box on the lower left corner of the display window.

You can also select an individual face lying on the surface mesh by right clicking on it; this will toggle selection of the face, but will not move the center of rotation to that face (nor will the dialog box on the lower left corner of the screen reflect any information on the selected face). This type of selection is used when defining the face(s) from which you wish to generate oilflows (as described in the "Generating and Viewing Oilflows" section). Selecting individual faces in this way works well if you only want to select a handful of faces from which to generate oilflows; if you want to select several adjacent faces at once, SimpleView provides a handy way to do so: while positioning your cursor in the vicinity of the surface mesh faces that you wish to select, hold down the shift key and left mouse buttons simultaneously; as you move the cursor, a transparent blue selection box will appear, as shown in Figure 2.2.5(a).

(a)

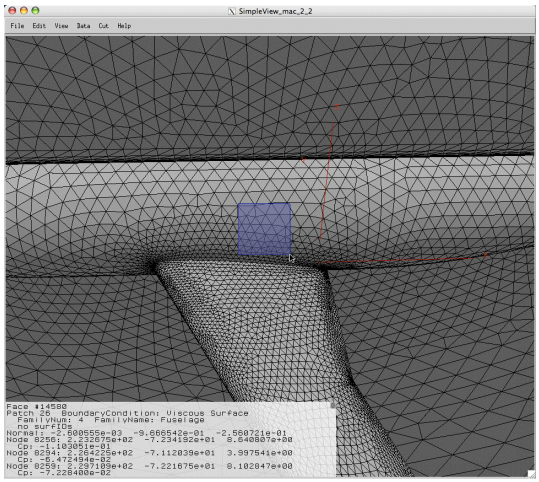


Figure 2.2.5 Group selection
(used for group-selecting faces from which to generate oilflows):
(a) the selection box

Move the cursor until the blue selection box covers the faces from which you want the oilflows to emanate; then, release the left mouse button (while still holding the shift key) to select the faces in the outlined area; the selected faces will be highlighted by your chosen Select Color (purple by default), as shown in Figure 2.2.5(b).

(b)

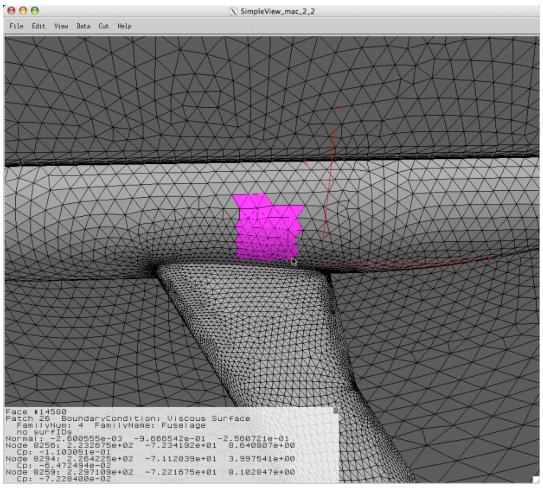


Figure 2.2.5 Group selection
(used for group-selecting faces from which to generate oilflows) :
(b) result of selection

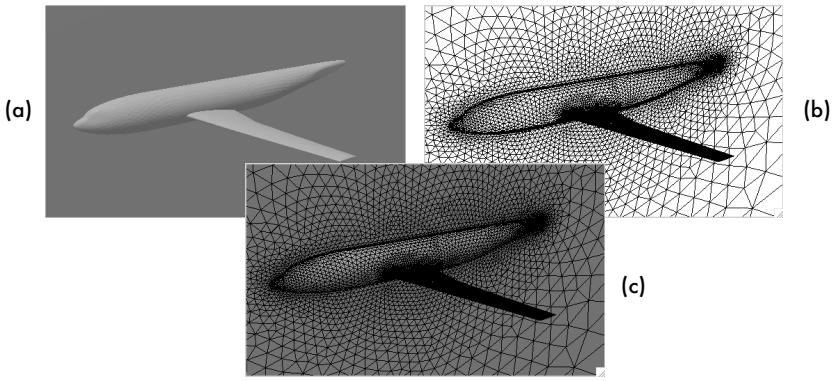
This kind of "group select" only selects faces within a narrow depth range based on the nearest triangle in the selection area, and therefore works best on surfaces which are parallel to the screen.

To "deselect" a face (or group of faces) that you've selected, you can hit the "Clear Selection" hotkey ("C" by default) or select "Clear Selection" from the Edit panel.

2.3 Viewing Surface Grids, Patches and Boundary Conditions

Viewing the Surface Grid

After loading the surface grid, SimpleView will automatically display it as a shaded surface with a wireframe of the grid overlaid on it. You can choose to suppress the display of the shaded surface, or the wireframe; to cycle through different combinations of these display modes, hit the 't' hotkey. The surface grid display options are shown in Figure 2.3.1. To toggle shaded surface mode, select the "Surface Faces" option in the View Menu; the shaded surface will be displayed when there is a check mark next to the "Surface Faces" entry in the View menu. Similarly, to



Figures 2.3.1 Surface Grid displayed in shaded surface mode (a), wireframe mode (b), and in both (c)

toggle wireframe mode, select the "Surface Wireframe" option in the View Menu; the wireframe will be displayed when there is a check mark next to the "Surface Wireframe" entry in the View menu.

Memory Requirement: Low (but proportional to grid size) CPU: Very Low

Surface Smoothing

SimpleView can display the shaded surface triangles on the configuration as smooth (rather than faceted) surfaces, as shown in Figure 2.3.2 below. To toggle surface smoothing, select the "Smooth Surface" option from the View panel. Note that the surface grid must already be displayed in shaded mode--as shown in Figure 2.3.1(a)--for you to see the effect of the smoothing.

Memory Requirement: Low CPU: Low

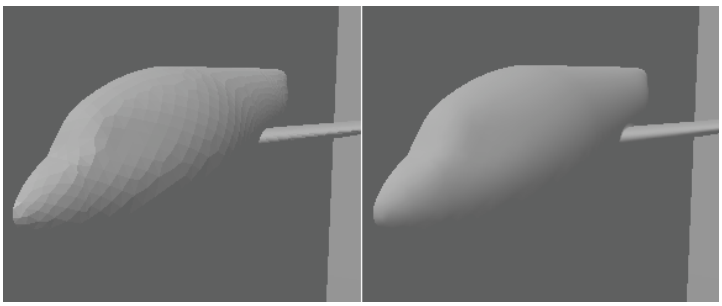


Figure 2.3.2 Faceted and Smoothed Surfaces

Surface Patch Viewing Options

Once a surface grid is loaded into SimpleView, you can display the configuration with each surface patch shaded in a different color, as shown in Figure 2.3.3. There are two ways to do this: you can hit the "Display Patches" hotkey ("p" by default) to toggle the display of the surface patches; alternatively, you can click on the "Surface Patches" option in the View panel.

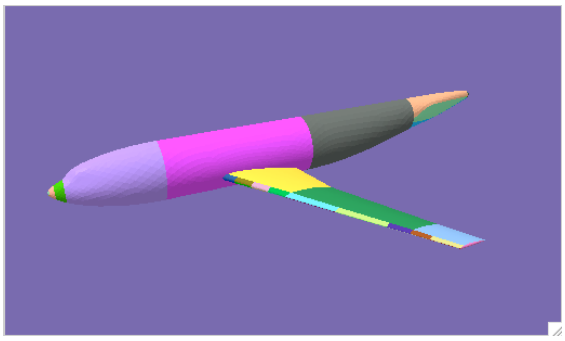


Figure 2.3.3 Displaying Surface Patches

You can also have SimpleView display only the boundaries of the surface patches, as shown in Figure 2.3.4. To activate this view, click on the "Surface Patch Boundaries" option in the View panel. This view option is very useful when displaying flow solution quantities on the surface grid-especially when there is a "symmetry plane" patch abutting the patches on the configuration itself. Note that the display of the patch boundaries is enabled by default; as such, suppressing the display of the surface wireframe still leaves the display of the patch boundaries.

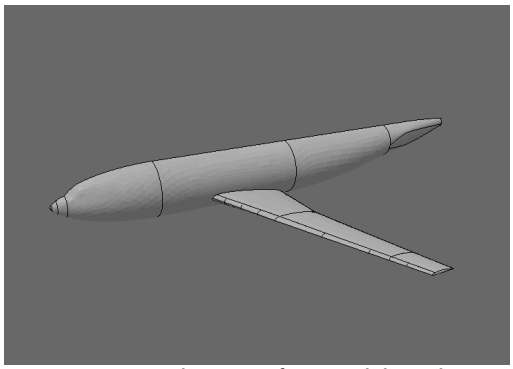


Figure 2.3.4 Displaying Surface Patch boundaries :
(a) all patches displayed

Placing your cursor over any surface triangle and clicking the middle mouse button will "blank out" the patch on which that surface triangle lies; to re-display/turn on the patch, click the middle mouse button once again (with your cursor anywhere on that same patch). The ability to turn patches "on and off" is useful when you are examining the flow features on the surface of a complex configuration and you wish to momentarily "remove" the display of the patches that get in the way of viewing the surface.

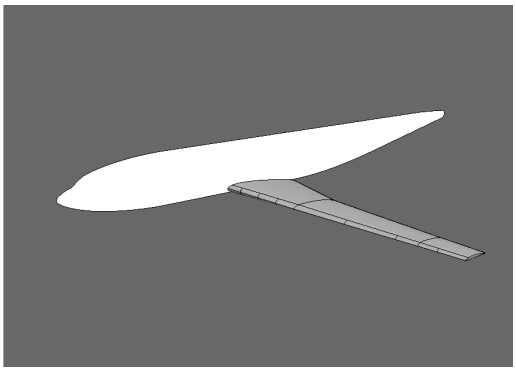


Figure 2.3.4 Displaying Surface Patch boundaries :
(b) with patch family blanked out

In fact, if you've taken the time to group your patches into patch "families" in GridTool, you can blank out the display of all the patches belonging to a particular family. To do this in the Linux variant, hit the "Turn Off Patches By Family Name" hotkey ("F" by default), and type the name of the patch family you want to blank out in the pop-up menu that appears. In order to restore the display of all blanked out patches, hit the "Restore All Patches" hotkey ("G' by default). On the Mac version, choose the "Turn Off Patches By Family" option in the View panel; to restore the display of the patches, choose "Turn On All Patches."

An example of this is shown in Figure 2.3.4(b), where the patches corresponding to the "fuselage" family have been blanked out.

Memory Requirement: Low CPU: Low

Viewing and Editing Boundary Conditions

When either a surface or volume grid is loaded, SimpleView will automatically attempt to load the MapBC file having the same project name and path (i.e., if you load "/home/Surface/airplane.cogsg", SimpleView will attempt to load "/home/Surface/airplane.mapbc"). You can

then display the configuration with each surface patch shaded according to the boundary condition that has been associated with it, as shown in Figure 2.3.5.

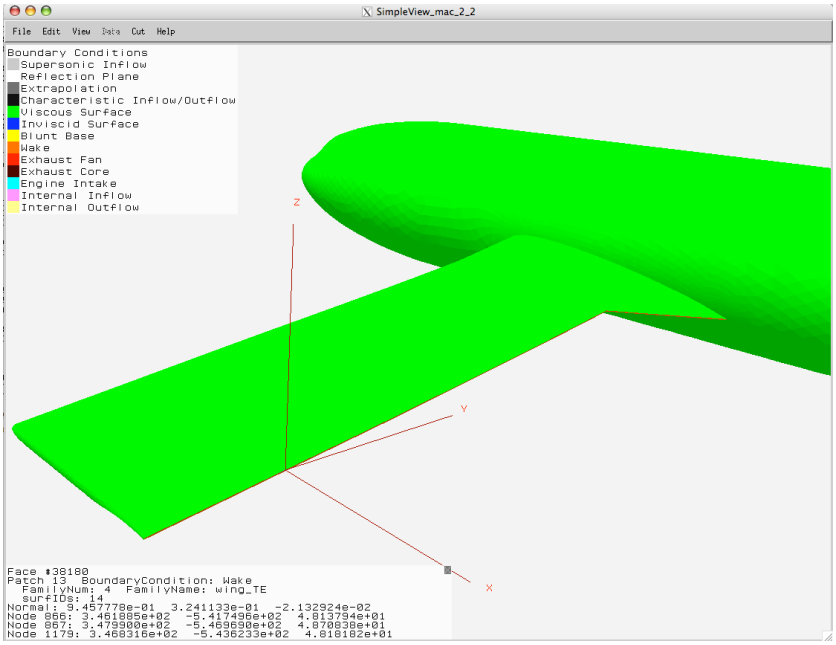


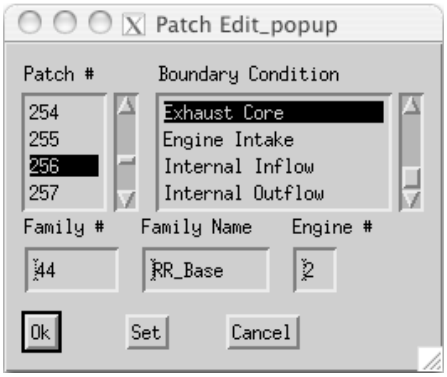
Figure 2.3.5 Displaying Boundary Conditions

There are two ways to do this: you can hit the "Display Boundary Conditions" hotkey ("b" by default) to toggle the display of the surface patches; alternatively, you can click on the "Boundary Conditions" option in the View panel. To display the Boundary Conditions legend, select the "Boundary Conditions Legend" option from the Legend submenu in the View panel.

The boundary conditions displayed correspond to those available in USM3D version 5.3.

Displaying the boundary conditions on each patch allows you to perform a rapid visual check to ensure that the appropriate boundary conditions are associated with each surface patch before running USM3D. Then, if you realize that an incorrect boundary condition has been associated with a particular surface patch, you can change the boundary condition from within SimpleView using the "Edit Patch Information" function. Selecting "Edit Patch Information" from the Edit menu will open the Patch Edit dialog box, examples of which are shown in Figure 2.3.6.

(a)



(b)

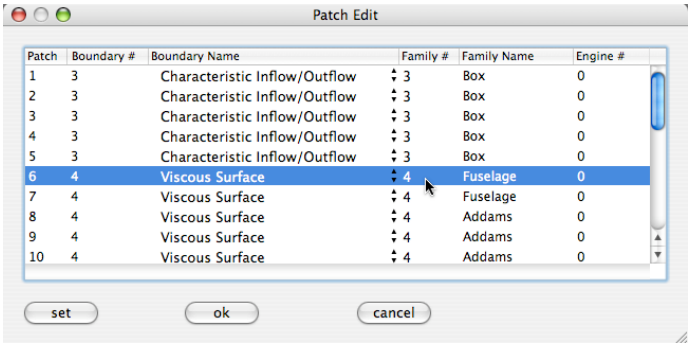


Figure 2.3.6 The Patch Edit Dialog in (a)the Linux Variant, and (b) the Mac variant

From within the Patch Edit dialog box, you can change the Boundary Condition, Family Name, and Engine Number (if applicable) of any patch. We suggest the following method for making changes to the data associated with a surface patch :

1. Make sure that you are viewing the surface boundary conditions (i.e., that the "Boundary Conditions" option is checked in the View panel), as shown in Figure 2.3.5.

2. If you've located a patch that has an incorrect boundary condition (or whose family name you wish to change), double-click on any triangle in that patch with the left mouse button. In addition to resetting the center of rotation to the center of that triangle, the lower left corner of the display window will now reflect the information (including Family Name and Boundary Condition) of the selected triangle.

3. Select the "Edit Patch Information" option from the Edit panel. The patch number corresponding to the patch on which you just selected will automatically be displayed and highlighted in the list, as shown in Figure 2.3.6. In the Linux variant, you need to scroll down the list of boundary conditions until you locate the one that is currently associated with that patch. In the Mac variant, the boundary condition currently associated with the selected patch is shown on the same line as the highlighted patch. You then select the appropriate boundary condition : in the Linux variant, this is done by clicking on the appropriate entry in the list of boundary conditions; in the Mac variant, you click on the disclosure triangles next to the boundary condition name, and select the appropriate entry, as shown in Figure 2.3.7.

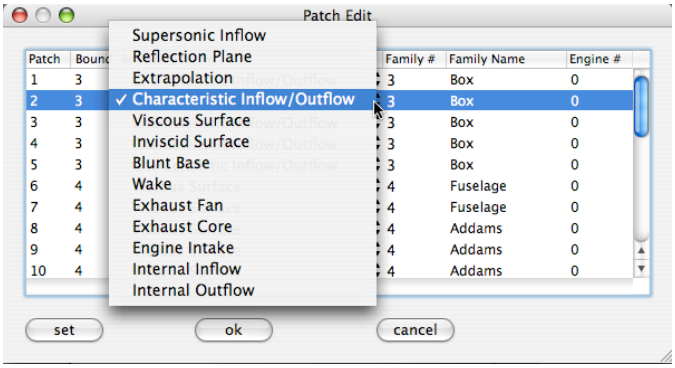


Figure 2.3.7 Boundary Condition Selector in the Mac variant

Note that in the Mac variant, the currently assigned boundary condition will have a check mark next to it (as shown in Figure 2.3.7).

Note also that when selecting an engine-related boundary condition (engine fan, core, or intake), you must select the boundary condition first, and then select the engine number to assign to that patch (otherwise, SimpleView does not apply the engine number to the patch).

4. If necessary, enter the Family Name you'd like to associate with the patch : in the Linux variant, this is done by typing in an entry in the Family Name field; in the Mac variant, you double-click on the existing Family Name, and type an entry into the resulting text field, as shown in Figure 2.3.8. then click on the "Set" button. Note that selecting a face on a different patch while the Patch Edit dialog is open will **not** automatically update the Patch Edit menu and select the new patch.

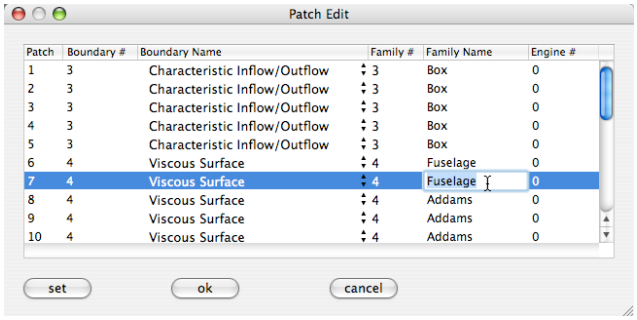


Figure 2.3.8 Patch Family Name specification in the Mac variant

5. Click on the "Ok" button to exit from the Patch Edit dialog.

6. Locate another patch whose boundary condition (or Family Name) needs to be changed, and repeat the process beginning with step 2 above.

It is important to note that SimpleView will **not** automatically write out a modified mapbc file containing the changes that you have made; if you have edited the patch information using the Patch Information editor, and wish to keep the changes, you should explicitly prompt SimpleView to write out a new .mapbc file by choosing the "Save MapBC" option from the File panel. SimpleView will then prompt you for a file name, giving you the option of writing your changes out to a new .mapbc file, or overwriting the original one.

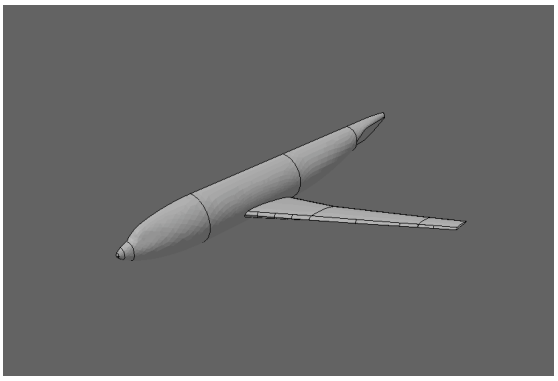
Memory Requirement: Low CPU: Low

Mirroring

SimpleView can display a mirror image of the configuration (i.e., the surface patches, surface grid and volume grid cuts, as well as any flow solution-related objects), mirrored about the patch set to the "reflection plane" boundary condition. SimpleView assumes that the patch associated with the "reflection plane" boundary condition is an x, y, or z-constant plane containing/passing through the origin (0,0,0). To activate mirroring, you can hit the "Mirror Grid" hotkey ("M" by default) to toggle the display of the mirrored objects This is illustrated in Figure 2.3.7.

It is important to note that when SimpleView performs this mirroring, the mirror image that it displays is exactly that–a mirror *image*. SimpleView does not actually "add" to the original grid, or alter any grid-related files (as would result from using the grid-mirroring options available in the "usgutils" package, for instance). In fact, when mirroring is enabled in SimpleView, you cannot select any surface faces on the mirrored side.

(a)



(b)

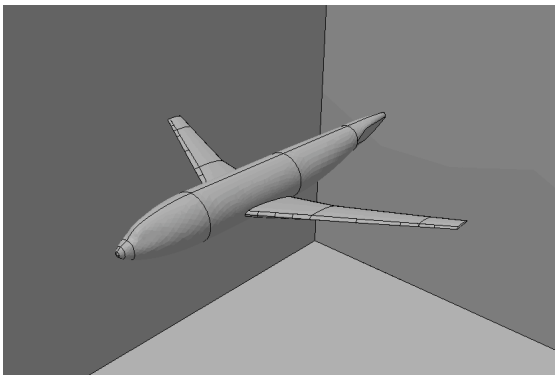


Figure 2.3.9 Mirroring : (a) half-span (actual) configuration, and (b) full-span (mirrored) display

Furthermore, in this release of SimpleView, you cannot display different flow solution quantities on each "side" of the configuration (the flow quantities displayed on one side will simply be "reflected" on the other half). The mirroring feature is useful mainly for creating images meant for presentations where it is more useful to show "full-span" images rather than

"half-span" ones, despite the fact that the grid generation and flow solution were actually carried out on the "half-span" configuration.

It should also be noted that if any other surface patches (other than the actual/geometric reflection plane) is set to the "reflection plane" boundary condition, mirroring may not work correctly.

Memory Requirement: Low CPU: Very Low

2.4 Viewing the Current Front

Viewing the "Current Front" at the Completion of the Advancing-Layers Stage of Volume Grid Generation

As mentioned in section 1.4, once an acceptable surface grid has been generated, the next stage in the viscous grid generation process is the generation of the so-called "advancing layers" portion of the grid. At the conclusion of this stage, it is very useful to load the surface grid and the current front into SimpleView (in the "Vol_AL" directory described in section 1.4); this allows you to view the "current front," and prompts SimpleView to automatically perform a search designed to determine the existence of any so-called "zero-layer faces"(surface triangles which were supposed to have advancing-layer type cells constructed above them, but which, for some reason, do not have a layer directly above them). It is very important to perform this automated search for the "zero-layer faces"; in fact, the results of this operation will often dictate whether you should proceed any further with the grid generation process, or return to GridTool to make changes to the surface patches and/or spacing specifications.

To view the current front at the completion of the advancing layers stage, begin by loading the surface grid file into SimpleView. Then, load the front file by choosing the "Load Current Front" option from the File panel. By default, the current front is displayed as a semi-transparent blue surface (which gives you an idea of how much of the volume has been taken up by the advancing layers) while still leaving the underlying surface grid visible; an example of this is shown in Figure 2.4.1.

Checking for "Zero-Layer Faces" at the Completion of the Advancing-Layers Stage of Volume Grid Generation

In addition to displaying the current front at the completion of the advancing-layers stage of grid generation, SimpleView will also automatically check for the existence of any zero-layer faces once you have loaded the surface grid and front files (as described in the preceding paragraph). SimpleView will then print the total number of zero-layer faces it finds in the lower left portion of the display window; in the example shown in Figure 2.4.1, no such faces have been found, and the text on the lower left corner reads "0 bad faces."

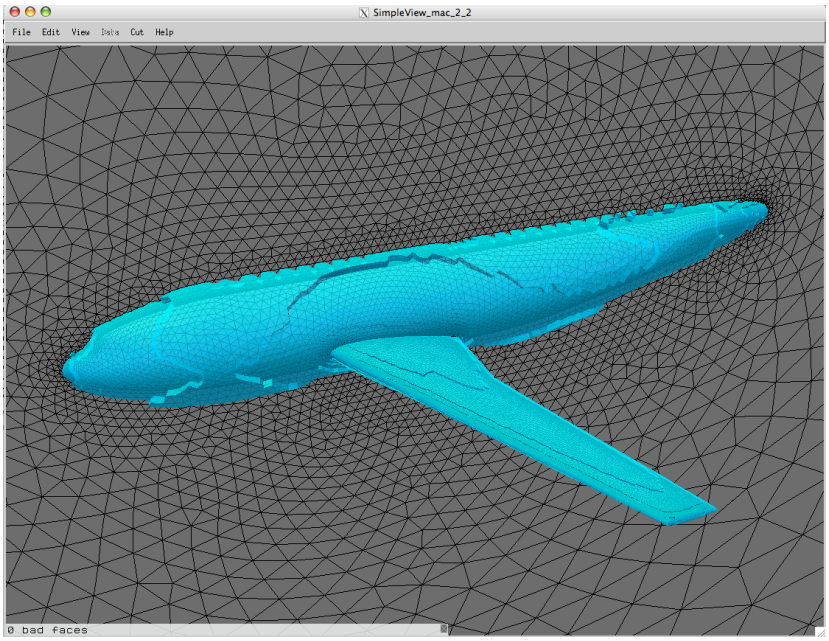


Figure 2.4.1 The "current front" at the completion of the "advancing-layers" stage

When SimpleView detects the existence of zero-layer faces, the total number of such faces is displayed in the lower left corner of the display window, and the center of rotation is automatically moved to the first zero-layer face in the list; enabling the display of the axes (by hitting the "x" hot key) provides a visual cue as to the location of the center of rotation.

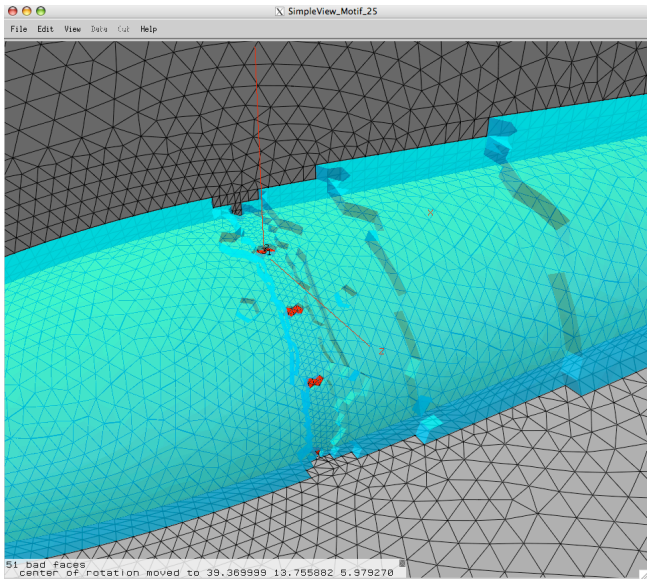
There are two scenarios that typically result in zero-layer faces being present at this stage; let's consider each of them separately:

1. zero-layer faces occur because VGRID's criteria for generating layers on a given face was not met : this occurs if the first-layer spacing ("Delta1") is on the order of the "inviscid" spacing in the surface grid. The "Delta1" parameter (specified in GridTool) corresponds to the height of the first layer of advancing-layers cells that VGRID will generate on every surface face lying on a patch having the "viscous surface" boundary condition. Since VGRID stops generating advancing-layer type cells when the height of those cells matches the height of "advancing front"/inviscid type cells (as dictated by the background grid sources), VGRID will not generate *any* layers at all on a given face if the very first cell height (as dictated by Delta1) already matches the inviscid spacing at that location. This scenario is shown in Figure 2.4.2; the zero-layer faces appear (in red) as "holes" in the current front. This occurs more commonly in "wall function" type grids which have larger Delta1 values compared to their "sublayer grid resolved" or "full viscous" counterparts. Typically, if this scenario occurs, subsequent runs of VGRID and POSTGRID are likely to be able to complete the grid; however, since these subsequent runs only use the advancing-front method, the grid that is generated off of the zero-layer faces will not exhibit the ordering present in the advancing-layers portion, and can thus lead to a degradation in the accuracy of the flow solution in that area of the configuration.

   Note that, as shown in Figure 2.4.2, SimpleView will automatically highlight any patches on which such "zero-layer faces" are found by displaying them in yellow.

2. zero-layer faces occur because there is an underlying problem with the surface mesh : sometimes, even a perfectly "legitimate" surface mesh can have problem areas containing "folded" surface triangles (that is, surface triangles that have somehow been generated in such a way that they "collapse" onto adjacent surface triangles); these will always lead to further problems in subsequent stages of grid generation. In this scenario, VGRID will attempt to build layers off of the folded surface triangles, but, as part of its routine diagnostics, VGRID will determine that the very first layer off of such a "folded" triangle contains a "negative (volume) cell," and will simply remove the problem cell (as it does whenever it encounters any "negative volume" cells at this stage). The result is that the "column" of advancing-layer type cells that is supposed to emanate from each surface

(a)



(b)

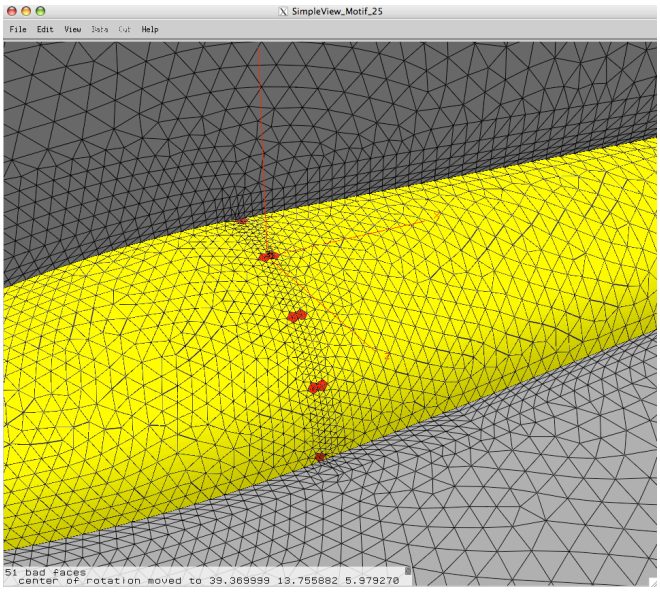
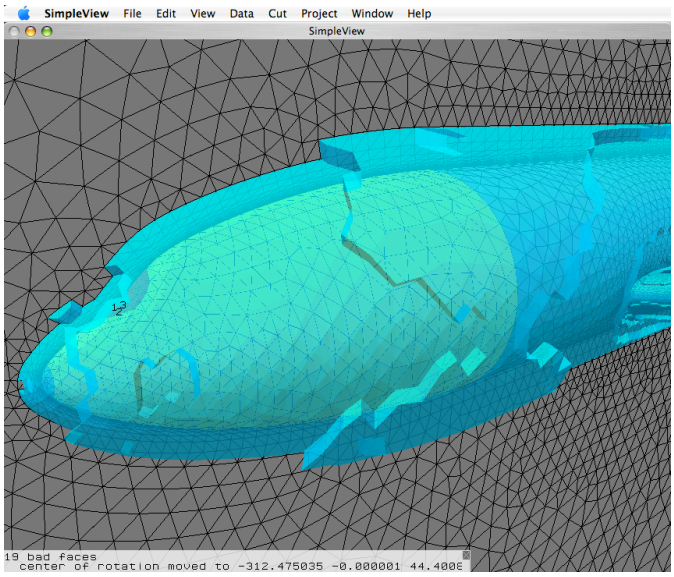


Figure 2.4.2 Display of "zero-layer faces" (shown in red) :
(a) with the front displayed, and (b) on the surface

(a)



(b)

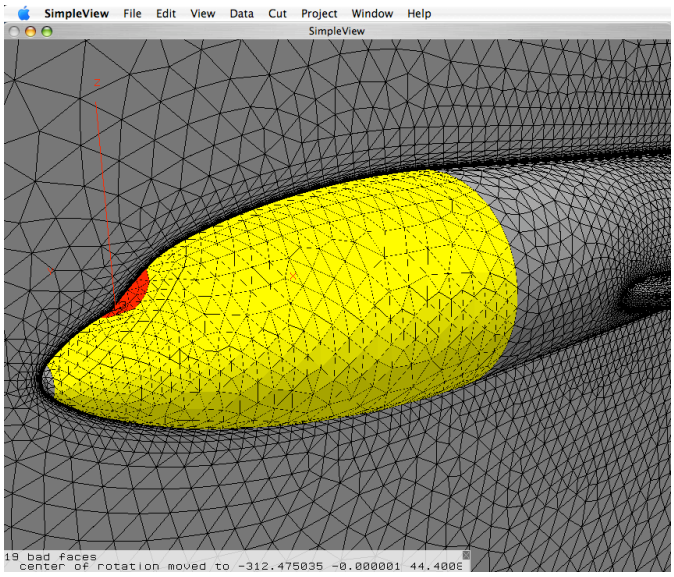


Figure 2.4.3 Display of "zero-layer faces" (shown in red) :
(a) with the front displayed, and (b) on the surface

triangle will have one or more missing layers in the vicinity of the folded surface triangles. This is an insidious type of error, as even a detailed visual examination of the surface mesh will not always show any obvious problems. Furthermore, this type of error is less likely to result in a completed grid (i.e., POSTGRID usually fails to complete the grid, as POSTGRID can only generate advancing-front type cells, and it is difficult to generate the kind of narrow, thin cells found within the advancing-layers portion of the grid). With SimpleView, you are at least made aware of the problem, and the location(s) at which it occurs. In these cases, you should return to GridTool and determine the source of the anomaly in the surface mesh (checking the details of the surface patch on which the problem triangles lie is always a good first step). This scenario is shown in Figure 2.4.3; the zero-layer faces are visible (in red) through the current front. In order to assist you in finding all the locations at which SimpleView has found such "zero-layer faces," SimpleView automatically highlights any patches containing zero-layer faces by displaying them in yellow. It is then a straightforward matter of locating the zero-layer faces themselves, which SimpleView displays in red, as shown in Figures 2.4.2 and 2.4.3.

Memory Requirement: Low CPU: Low

Viewing the "Current Front" During the Advancing-Front Stage of Volume Grid Generation

You can use SimpleView to examine the "current front" at any point during the advancing-front stage of volume grid generation, as well (that is, during the grid generation that occurs in the "Vol_AF" directory described in section 1.4). Since this stage of the grid generation can be performed in a series of "restarts," you can use SimpleView to examine the front at the end of each VGRID run. To view the current front at this stage, begin (as in the previous section) by loading the surface grid (or indeed, the entire volume mesh that has been generated up to that point) into SimpleView. Then, load the current front by choosing the "Load Current Front" option from the File panel. By default, the current front is displayed as a semi-transparent blue surface, which gives you an idea of how much of the volume has been taken up by the grid, while still leaving the underlying surface grid visible; examples of this are shown in Figure 2.4.4. The first figure (Figure 2.4.4(a)) shows an intermediate point in the advancing-front stage.

(a)

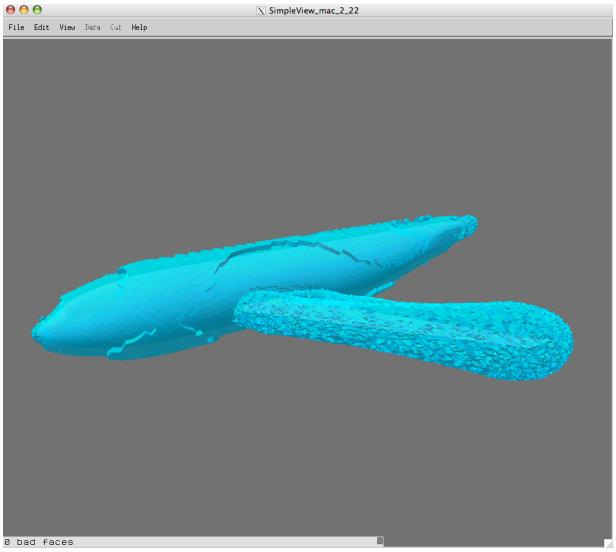


Figure 2.4.4 The "current front" (shown in blue) :
(a) at an intermediate point in the advancing-front stage of volume grid generation

(b)

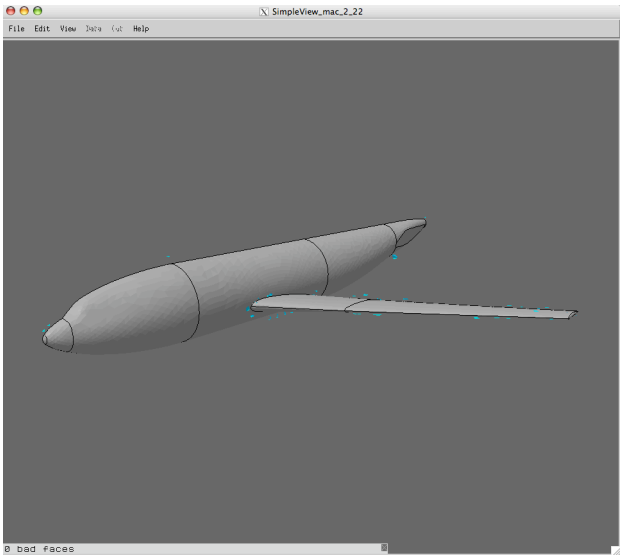


Figure 2.4.4 The "current front" (shown in blue) :
(b) at the completion of the advancing front stage of volume grid generation

At the completion of the advancing-front stage (when VGRID writes out the "No more cells can be formed, grid may now be completed by post-processing" message), the front file will contain the holes ("pockets") in the volume grid that will need to be re-meshed and filled in by POSTGRID. An example of this is shown in Figure 2.4.4(b).

Memory Requirement: Low CPU: Low

2.5 Viewing Completed Volume Grids

Once you have successfully run through POSTGRID and have a completed volume grid, SimpleView allows you to display the grid cells crossing an arbitrary plane (so-called "crinkle cuts") whose orientation you specify. Here's how you do it:

1. Load the full volume grid into SimpleView (as described in the "Loading the Grid into SimpleView" portion of section 2.2).
2. Select the "Define New Cut" option from the "Cut" panel. The "Data Cut Popup"/"Define New Cut" window will appear, and, by default, SimpleView will display a transparent blue plane at the x=0 location, as shown in Figure 2.5.1.

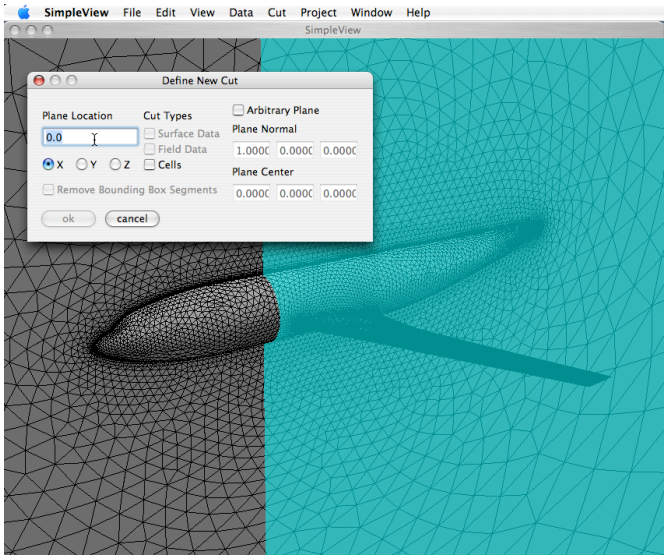


Figure 2.5.1 The Data Cut Pop-up Menu : specifying the cutting plane location

3. If you wish to examine the cells crossing a constant coordinate plane, enter a value corresponding to the location of the plane in the field on the upper left corner of the pop-up, hit return, and click on either the x, y or z boxes to indicate the orientation of the constant-coordinate plane (either constant x, y or z). The transparent blue cutting plane will move to the location specified.

4. Click on the "Cells" selection box to indicate that you want SimpleView to find the cells crossing the specified plane, as shown in Figure 2.5.2.

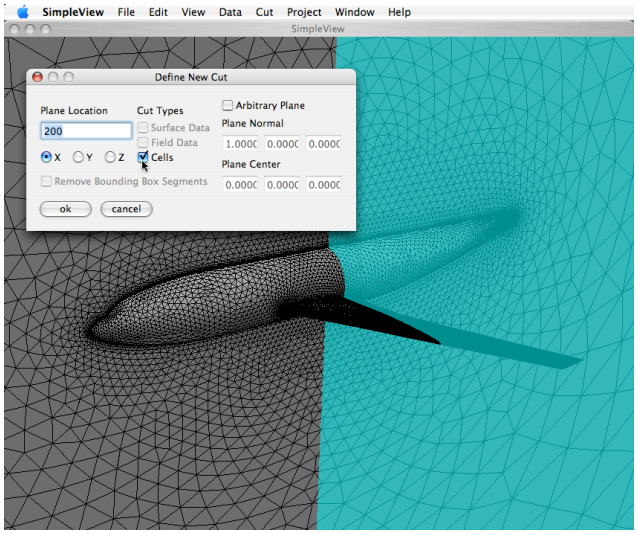


Figure 2.5.2 The Data Cut Pop-up Menu : selecting the "Cells" option from the Data Cut dialog

5. Click on "OK." You'll notice that the transparent blue plane disappears, and in its place, a display of all the cells crossing that plane appears; by default, the cells are shown with both the wireframe and face-shading enabled, as illustrated in Figure 2.5.4.

6. From within the "Cut" panel, you can choose to deselect the "Show Volume Grid Cut Shaded Surface" button, which leaves the display of the wireframe view of all the cells crossing the specified plane. Alternatively, you can deselect "Show Volume Grid Cut Wireframe" if you wish to display only the shaded representation of the cells crossing the specified plane. An example of this is shown in Figure 2.5.5.

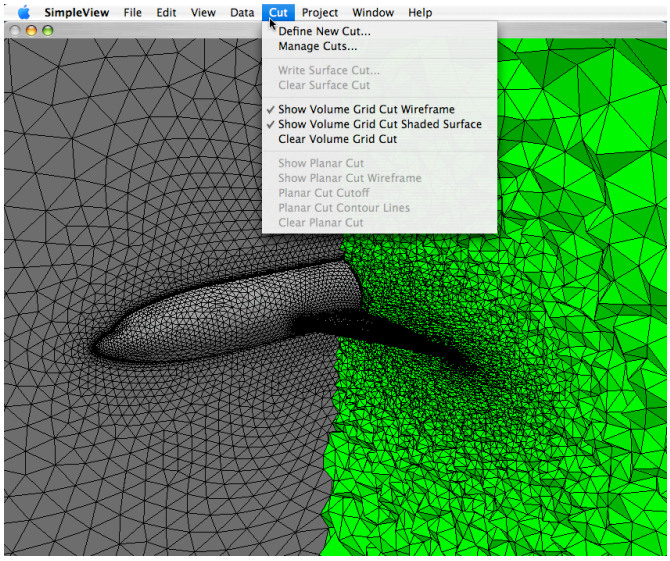


Figure 2.5.4 Displaying cuts through the volume :
shaded and wireframe views activated simultaneously (default)

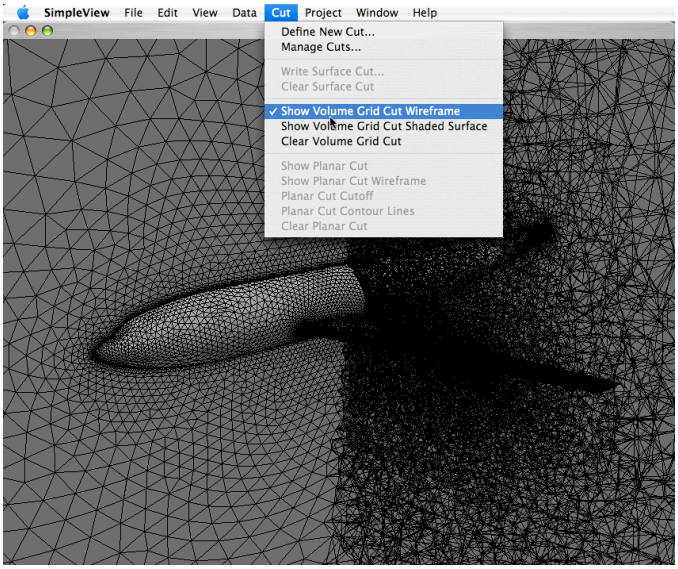


Figure 2.5.5 Displaying cuts through the volume :
wireframe view

If you wish to generate additional cuts, you simply repeat the process (beginning with step 2). If you wish to clear the display of volume grid cuts, choose the "Clear Volume Grid Cut" option from within the "Cut" panel, and all displayed cuts will disappear.

As mentioned in the "Selecting Objects in SimpleView" portion of section 2.2, you can also select individual cells in the grid by hitting the "Select Cell" hotkey ("c" by default) and entering the cell number. SimpleView will then display the cell as shown in Figure 2.2.3.

You are not limited to defining cutting planes that are constant-coordinate surfaces; you can also create a cutting plane by specifying the components of a vector normal to the plane, and a point lying on the plane. The procedure for creating these "advanced" cuts is similar to that outlined in the steps above, with the exception of the fact that in step 3, you must select the "advanced cut" option in the Data Cut Pop-up, and specify the two additional parameters (the plane normal and point) described earlier.

At this point, it is also strongly suggested that you run the " CheckLayers" utility (discussed in section 3.2) in order to ascertain that your grid provides you with the level of resolution that you anticipated in the advancing-layers portion of the grid.

Memory Requirement: High CPU: Low

2.6 Viewing Flow Solution Data

This section covers the procedures for using SimpleView to display USM3D flow solution-related quantities of interest on the surface of the configuration as well as in the field surrounding it.

To view USM3D flow solution data, you must load the USM3D solution by selecting the project's .flo file after loading either the full volume grid or the surface mesh. SimpleView will read the entire .flo file if it is loaded after having loaded the full volume grid; if you only load the surface grid, SimpleView will read only the portion of the .flo corresponding to the surface grid (including the outer boundaries of the domain).

In addition to the five data sets loaded from the .flo file (density, the x, y, and z components of momentum, and total energy), SimpleView can calculate an additional six data sets (static pressure, Mach number, total pressure, entropy. Cp, and temperature). By default,

SimpleView will only keep one of these additional data sets in memory at a time, to reduce memory usage. If memory usage is not a concern (for example if you are running on a workstation as opposed to a laptop), SimpleView can be made to keep all six additional data sets in memory by changing the "Only store six datasets" option in the Preferences dialog. When the button next to this option is depressed, SimpleView operates in the default mode; otherwise, SimpleView will store all six additional datasets in memory. Note: this setting cannot be changed after a solution has been loaded; as such, if you wish to have SimpleView store all six additional datasets in memory, make the change immediately after starting SimpleView, or immediately after reading in the grid.

Viewing Surface Quantities

If you are only going to be examining flow features on the surface, you can get by with loading only the surface grid; then, when you load the .flo file (by choosing the "Load USM3D solution" option from the File menu), SimpleView will read in only the portion of the file pertaining to the surface grid. In this mode, SimpleView uses less memory than it would if the entire volume mesh and solution files had been read in.

Shaded Surface Contours

After loading the . USM3D solution (flo file), SimpleView will automatically display the static pressure surface data. To switch to a different data set, select it from the Dataset submenu of the Data Menu (as shown in Figure 2.6.1).

To change the scaling range, open the Data Range dialog by the selecting the "Data Range" option in the Data Menu.

You can also display a legend for the currently selected dataset by selecting the "Data Legend" option from the Legends submenu, found in the View panel. This will toggle display of the Data Legend, as shown in Figure 2.6.2.

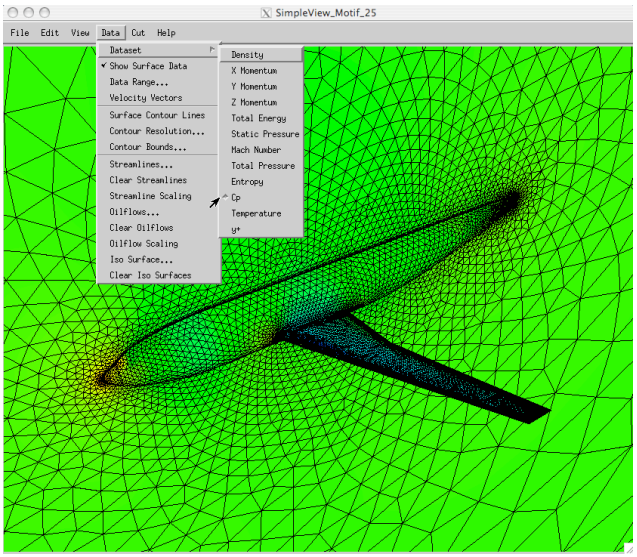


Figure 2.6.1 The flow quantity dataset selector

(a)

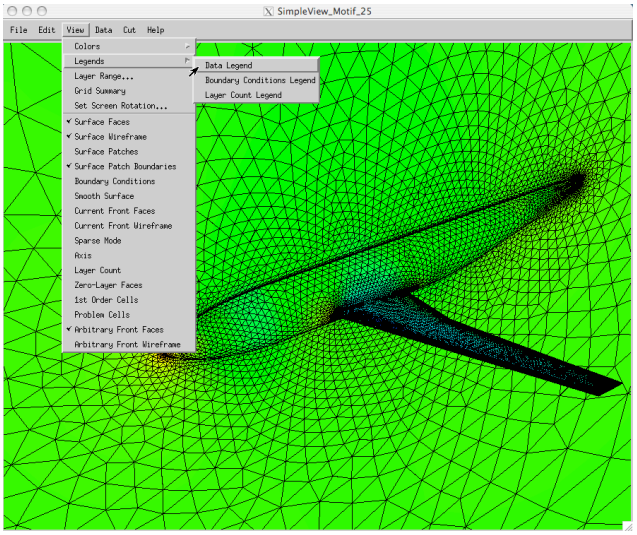


Figure 2.6.2 The Data Legend selector :
(a) selecting the display of the legend for the currently selected dataset

(b)

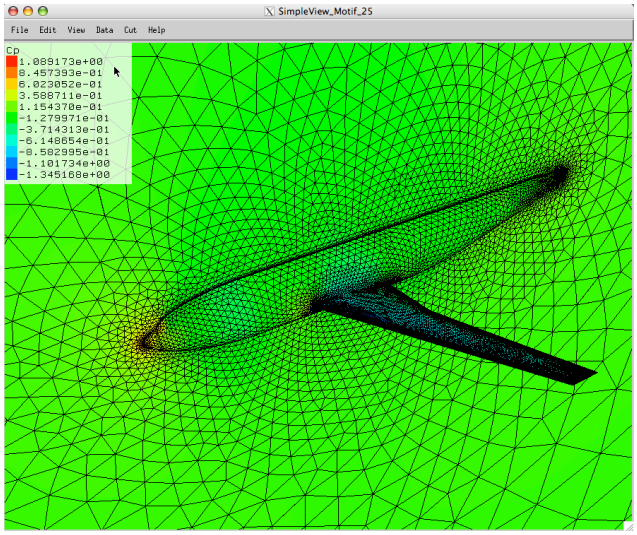


Figure 2.6.2 The Data Legend selector
(b) with the Cp legend displayed

To suppress the display of the surface data, uncheck the "Show Surface Data" option in the "Data" menu. To suppress the display of the legend, uncheck the "Data Legend" option from the Legends submenu in the View panel.

Memory Requirement: Moderate CPU: Low

Surface Contour Lines

To display surface contour lines (as shown in Figure 2.6.3), select the "Surface Contour Lines" option from the Data menu. Note that if you have the current dataset displayed as a shaded surface, you should first suppress the display of the surface data (as described in the previous section), as the contour lines will be "hidden" by the shaded surface data.

To change the Contour resolution (the number of contour lines), select the "Contour Resolution..." option from the Data menu, and enter the new resolution (i.e., enter the number of contour lines you'd like to see) in the Contour Resolution dialog.

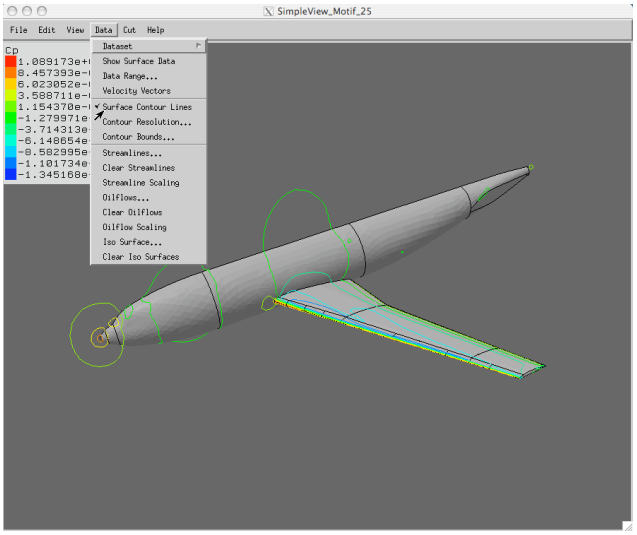


Figure 2.6.3 Surface contour line representation of current dataset.

The data values of individual contour lines can be changed with the Contour Bounds dialog, which is opened by selecting the "Contour Bounds..." option from the Data menu; this dialog is shown in Figure 2.6.4.

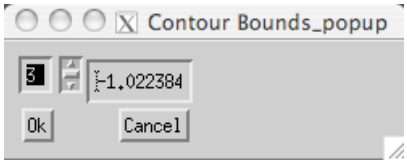


Figure 2.6.4 The contour bounds dialog

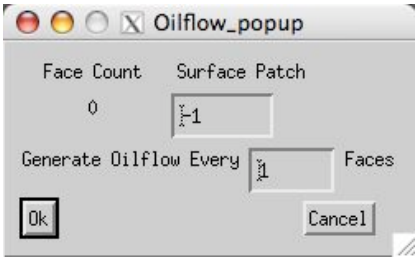
This capability is useful if you want particular contour lines to correspond to particular values. The listbox on the left-hand side of the dialog allows you to select a given contour line; clicking on an item in the list provides you with the current value that the particular contour is set to. The edit box on the right allows you to set the data value of that contour line. Contour line values cannot be set outside the current data range, and must be between the values of the neighboring contour lines (i.e., the value of the 4th contour line must be greater than the value of the 3rd and less than that of the 5th). Contour lines will be antialiased if antialiasing is on.

Memory Requirement: Moderate/High CPU: Low

As discussed in section 2.1, oilflows are an effective way of visualizing the flow on the surface of a configuration. In SimpleView, oilflows are computed beginning at the center of selected surface triangles. There are two ways to select surface triangles for use in generating oilflows in SimpleView; we'll discuss each of them separately.

1. You can select surface triangles belonging to a particular patch by selecting the "Oilflows..." option from the Data menu. Doing so brings up the Oilflow dialog box, an example of which is shown in Figure 2.6.5.

(a)

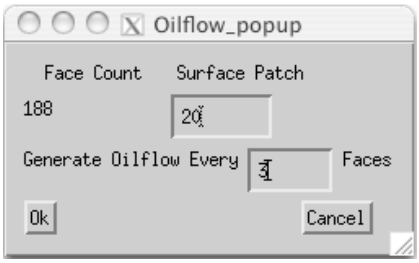


(b)

Figure 2.6.5 The Oilflow Dialog
(a)initial settings, Linux variant, and (b)initial settings, Mac variant

When you first bring up the Oilflow dialog in the Linux variant, the number "-1" appears in the Surface Patch field, and the "Face Count" field displays a value of zero; similarly, in the Mac variant, the "Number of Oilflows to Generate" field initially displays a value of zero. This is shown in Figure 2.6.5 a and b.When you enter/select a particular surface patch number in the Surface Patch/Patch Number field, the Face Count/"Number of Oilflows to Generate" fields are updated to reflect the number of faces on that patch, as shown in Figure 2.6.5 c and d.

(c)



(d)

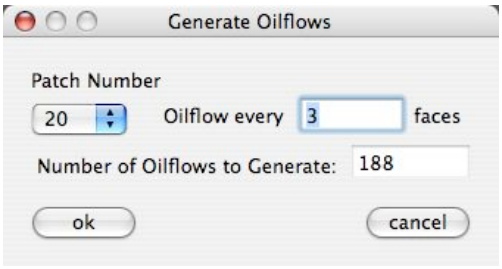
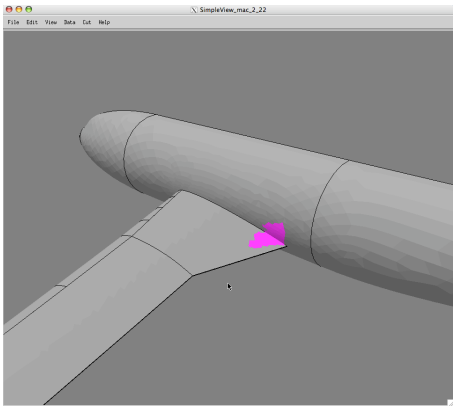


Figure 2.6.5 The Oilflow Dialog
(c) updated settings for selected patch, Linux variant
(d) updated settings for selected patch, Mac variant

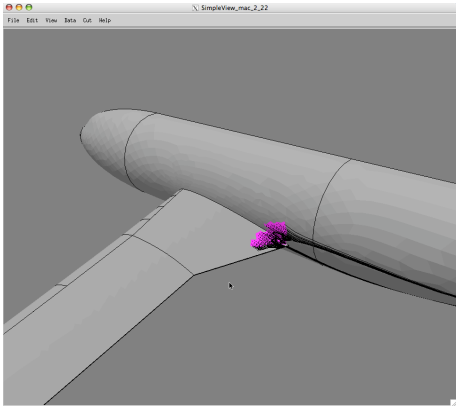
If you click "ok" at this point, SimpleView proceeds to generate oilflows emanating from the center of each surface triangle on that patch. Since the calculation of oilflows is a relatively CPU-intensive operation, and since some patches have relatively large numbers of surface triangles on them, SimpleView allows you to generate oilflows using a subset of the surface triangles on the given patch. You do this by specifying a value in the "Generate Oilflow Every _ Faces" field; specifying a value of 3, for instance, instructs SimpleView to calculate oilflows beginning on every third surface triangle. Every time you enter a value in this field, SimpleView updates the Face Count value to indicate how many faces it will use to generate oilflows emanating from that patch. Click the "ok" button to begin the calculation (and display) of the oilflows.

2. You can graphically select specific faces, either singly or as a group, from which the oilflows will emanate. In this case, you are not limited to selecting from faces on a patch--you can choose a group of faces that cross patch boundaries. The methods used for selecting surface triangles for this purpose (either singly, or in a group) is discussed in the "Selecting Objects in SimpleView" portion of section 2.1. The result of one such "group select" operation is shown in Figure 2.6.6. Once you've selected the faces from which you want the oilflows to emanate, hit the "o" key to begin the calculation (and display) of the oilflows.

(a)



(b)



(c)

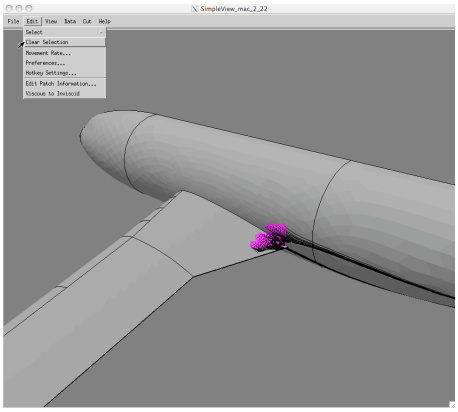
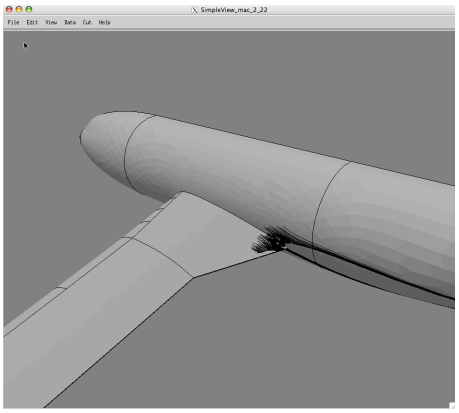


Figure 2.6.6 Generating oilflows:
(a) selected faces, (b) the resulting oilflow
(c) clearing the selected faces

If the "Oiflow Scaling" option is selected, the oilflow lines are colored by velocity, which should vary as the Mach number. An example of this is shown in Figure 2.6.6(f). The colors are scaled by default from low (blue) to high (red). The scaling range can be changed from the command-line by hitting the 'O' hotkey. If Oilflow Scaling is off, the oilflow lines will be drawn in a constant color, which can also be changed in Color submenu.

(d)



(e)

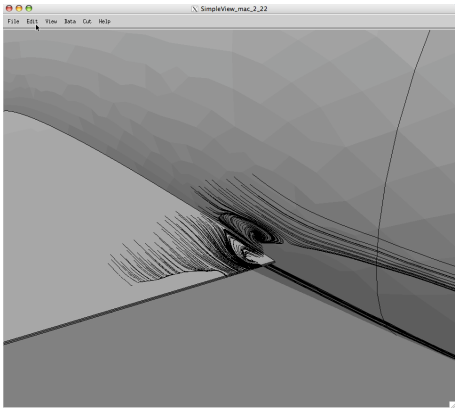


Figure 2.6.6 Generating oilflows:
(d) oilflow without selected faces, (e) close-up of the oilflow

Oilflows will be displayed until they are cleared from memory by selecting the "Clear Oilflows" option from the Data Menu (as shown in Figure 2.6.6(c)), or by hitting the "Clear Oilflows" hotkey ("C" by default).

Note: you should always perform a "clear selection" (by using the "shift c" hotkey, or by clicking on "Clear Selection," as shown in Figure 2.6.6(c)) between successive oilflow calculations. SimpleView considers any selected face as a starting point for an oilflow. As such, failure to clear the selection before moving on to another set of oilflows emanating from another group selected faces will result in the re-calculation of already existing oilflows, which introduces needless delays in generating the oilflows.

(f)

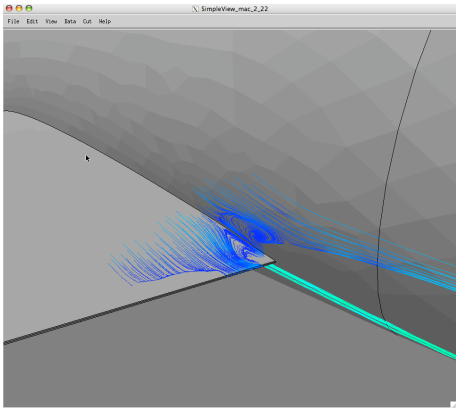


Figure 2.6.6 Generating oilflows:
(d) oilflow without selected faces, (e) close-up of the oilflow
(f) oilflow scaling

As shown in Figure 2.6.6 (e), oilflows are a particularly effective way to locate the extent of recirculation regions, as well as flow separation (and reattachment) lines.

Memory Requirement: High/Very High CPU: High/Very High

Viewing Surface Cuts and Writing Surface Cut Data to a File

Just as SimpleView can display the cells of the volume grid that cross a user-defined "cutting plane," it can also be used to extract the values of flow quantities along the intersection of the user-defined plane and the surface grid. Those values can then be written out onto a file for use in a 2-d plotter, for example.

To extract the flow quantities, you begin by defining a plane in the same way that you would do so in order to perform a volume grid cut (as shown in Figure 2.5.1)--by selecting the "Define New

Cut" option from the "Cut" panel. Then, click on the "Surface Data" button in the Data Cut dialog box (as shown in Figure 2.6.7(a)) before or after specifying the location of the plane. As with the volume grid cuts, the cutting plane can be a constant-coordinate plane, or an "advanced" cut specified by providing a coordinate on the plane, and the components of the plane normal.

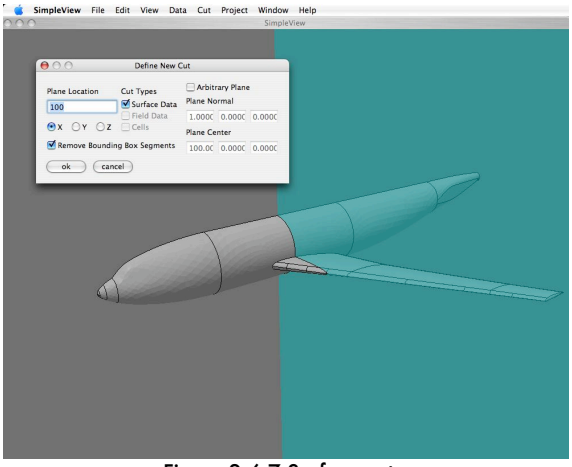


Figure 2.6.7 Surface cuts:

(a) Defining the location for a surface cut

After the "OK" button is pressed, SimpleView will display a black line corresponding to the intersection between the plane and the surface grid; an example of this is shown in Figure 2.6.7(b).

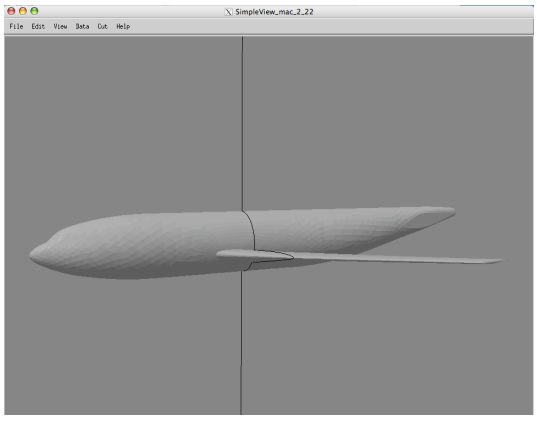


Figure 2.6.7 Surface cuts:

(b) display of the surface cut

To write the data along the intersection out to a file, select the "Write Surface Cut" option from the "Cut" panel. This will bring up the "Write Cut" dialog, an example of which is shown in Figure 2.6.7(c).

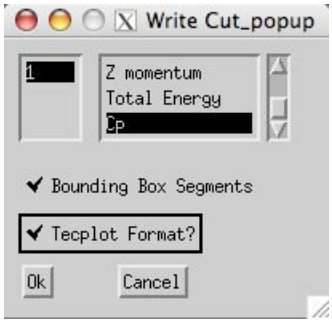


Figure 2.6.7 Surface cuts:
(c) the "Write Cut" dialog

If you wish to write out the extracted data values on the entire intersection curve--including the portion of the intersection curve that lies along the outer boundary/bounding box--then leave the "Bounding Box Segments" option checked (this is the default setting). If you do not want to include the data values along the "box" portion of the intersection line, click on the box next to the "Bounding Box Segments" in the dialog so that the box is unchecked (as shown in Figure 2.6.7(d)). Note that in the Mac variant of SimpleView, the choice of including the bounding box segments in your selected cut is performed in the "Define New Cut" panel, as shown in Figure 2.6.7(a).

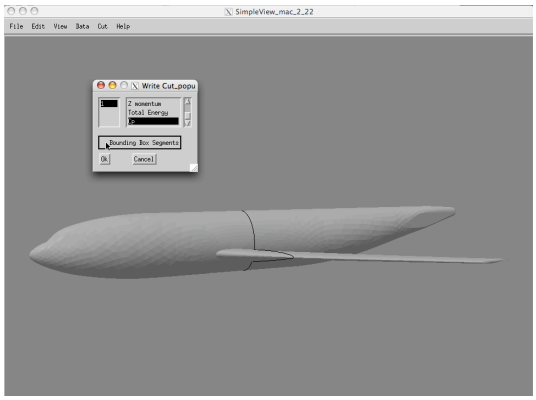


Figure 2.6.7 Surface cuts:
(d) removing the bounding box from the surface cut

Select the cut number to save in the left-hand listbox, and select the quantity you wish to write from the available datasets listed in the right-hand listbox. If the particular dataset that you want to write out does not appear in the listbox on the right-hand side of the "Write Cut" dialog, it indicates that SimpleView has not yet calculated that dataset from the quantities specified in the .flo file. If this is the case, then choose "cancel" from the "Write Cut" dialog, and have SimpleView compute the dataset by selecting the desired dataset from among the choices shown in the "Dataset" menu (found in the "Data" panel, as shown in Figure 2.6.1).

Selecting "Ok" from the "Write Cut" dialog will open a file selector so you can specify an output file to write the data to. You have the option of writing the data in Tecplot format (so that the appropriate variable names will appear when the data are plotted using Tecplot). The first few lines of a sample output files (written in Tecplot format, containing the Cp values at the x= 100 location, for example), is shown in Figure 2.6.8.

```
VARIABLES = "X","Y","Z","Cp"
ZONE, DATAPACKING=POINT
100.00000000000000000   -0.00027209618201596    -49.02113000000099419   -3.337779e-02
100.00000000000000000   -7.78447480579400786    -48.39463048821978219   -3.430791e-02
100.00000000000000000   -7.78447480579400786    -48.39463048821978219   -3.430791e-02
100.00000000000000000   -12.50642076220892385   -47.95336149645135038   -3.469267e-02
100.00000000000000000   -12.50642076220892385   -47.95336149645135038   -3.469267e-02
100.00000000000000000   -19.77806010998607178   -46.16507374087794346   -4.095671e-02
100.00000000000000000   -19.77806010998607178   -46.16507374087794346   -4.095671e-02
100.00000000000000000   -22.68662041317469402   -45.46844814995095874   -4.293733e-02
```

Figure 2.6.8 An example of surface cut data written to a file

To clear the most recent surface cut from memory, select the "Clear Surface Cut" option from the Cut Menu.

Memory Requirement: Low CPU: Moderate

Viewing Field Quantities

If you are going to be examining flow features in the field, you need to choose the "Cogsg and BC (Full)..." option to load the entire mesh; then, when you load the .flo file (by choosing the "Flo..." option from the Load menu), SimpleView will read in the entire file, and not just the portion pertaining to the surface. All of the operations discussed in this section require you to read in the entire grid and solution file.

Generating and Viewing Streamlines

There are two ways to generate streamlines in SimpleView: you can generate a single streamline emanating from an arbitrary point in the field, or generate a group of streamlines emanating from a "rake" of points in the field. Lets' discuss each of these separately:

1. To generate a single streamline, hit the Streamline hotkey ("J" by default). In the terminal window from which you launched SimpleView, you will be prompted to enter the coordinates of the origination/starting point of the streamline, as well as the direction of the single streamline. A "downstream" direction tells SimpleView to generate a streamline that will begin at the starting point, and proceed downstream until the end of the domain. An "upstream" direction tells SimpleView to generate a streamline that will begin at the upstream boundary of the domain, and terminate at the specified starting point.

2. To generate a set of streamlines emanating from a "rake," select the "Streamlines" option from the Data Menu, which opens the Streamlines dialog, as shown in Figure 2.6.9.

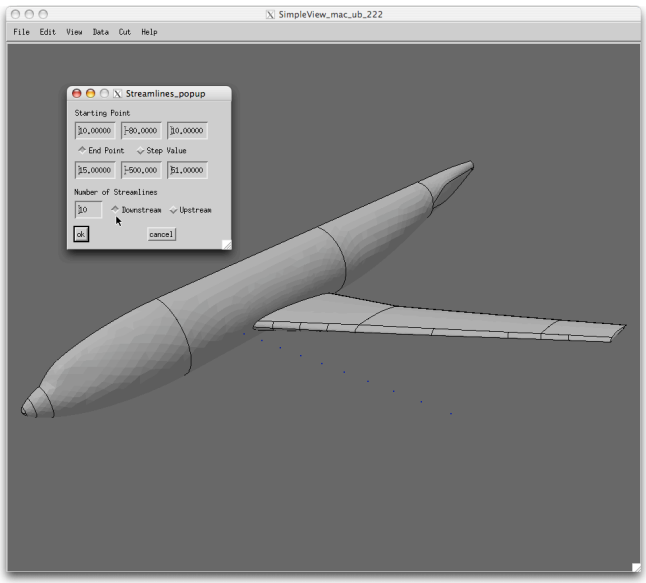


Figure 2.6.9 The Streamlines Dialog

The top row of text boxes in the Streamlines dialog specifies the coordinates of the starting point for the rake, and the second specifies either the ending point or the step value (the increment

from the starting point in the three coordinate directions), as selected in the radio buttons between the two rows. The lowest text box specifies the number of points in the rake (points from which streamlines will emanate), and the radio buttons beside it determine the direction of the streamlines. A "downstream" direction tells SimpleView to generate streamlines that will begin at the starting point, and proceed downstream until the end of the domain. An "upstream" direction tells SimpleView to generate streamlines that will begin at the upstream boundary of the domain, and terminate at the specified starting points.

As you enter data into the Streamlines dialog, SimpleView will display the points on the rake from which the streamlines will emanate (these are the small blue dots seen just upstream of the wing leading-edge in Figure 2.6.9). Hitting the "ok" button will begin streamline generation. If the "Streamline Scaling" option is off, streamlines are drawn in the default streamline color (yellow by default), which can be changed from the Color submenu. If the "Streamline Scaling" option is on, streamlines are shaded by velocity. The default scaling range is from low (blue) to high (red). Examples of streamlines generated using the rake shown in Figure 2.6.9 are shown in Figure 2.6.10 (for both "downstream" and "upstream" streamlines).

(a)

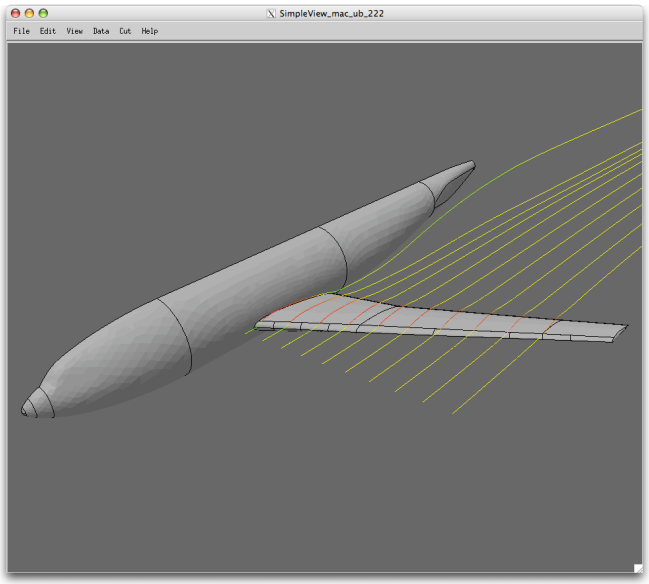


Figure 2.6.10 Streamlines
(a) "Downstream" streamlines (with streamline scaling activated)

(b)

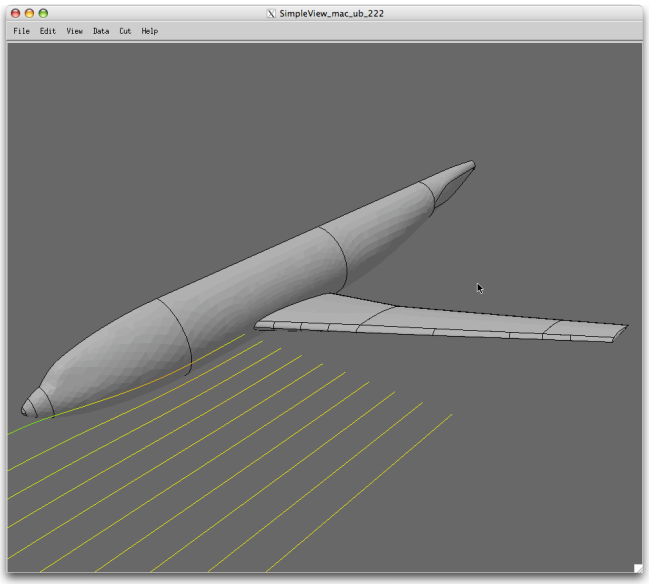


Figure 2.6.10 Streamlines
(b) "Upstream" streamlines (with streamline scaling activated)

The calculations involved in streamline generation are relatively intensive; as such, streamline generation is significantly slower than oilflow generation, particularly when streamlines pass through the "advancing layers/viscous layers" portion of a grid. It is also possible for streamlines to "crash" into viscous surfaces--that is, some streamlines can simply appear to stop, and not continue travelling up or downstream. If you encounter this behavior, simply specify a slightly different rake position, and try generating the streamlines again.

It should also be noted that the very first time you prompt SimpleView to generate streamline s (either singly, or from a rake) for a grid, it will perform a sorting of all the grid's volume cells, which can be time consuming (while this operation is in progress, you will see the status message "sorting..." printed out to the console window from which you launched SimpleView). This sorting only needs to be performed once per session; as such, after you've generated one set of streamlines, subsequent streamlines will not take as long to compute and display as the first set.

Memory Requirement: High CPU: High/Very High

Generating and Viewing Iso-Surfaces

To create an iso-surface, either hit the iso-surface hotkey ("I" by default) or click on the "Iso-Surface..." option in the Data menu; either method will open the iso-surface dialog, shown in Figure 2.6.11. In the iso-surface dialog, select the data set you wish to use in creating the iso-surface from the list in the left of the dialog, as shown in Figure 2.6.11. Then, enter the data value in the "Iso Surface Value" field. Hitting "ok" then prompts SimpleView to display surfaces shaded by that value. The example shown in Figure 2.6.11 shows the iso-surface corresponding to a Mach number of 1 (the freestream Mach number for this case was 0.75).

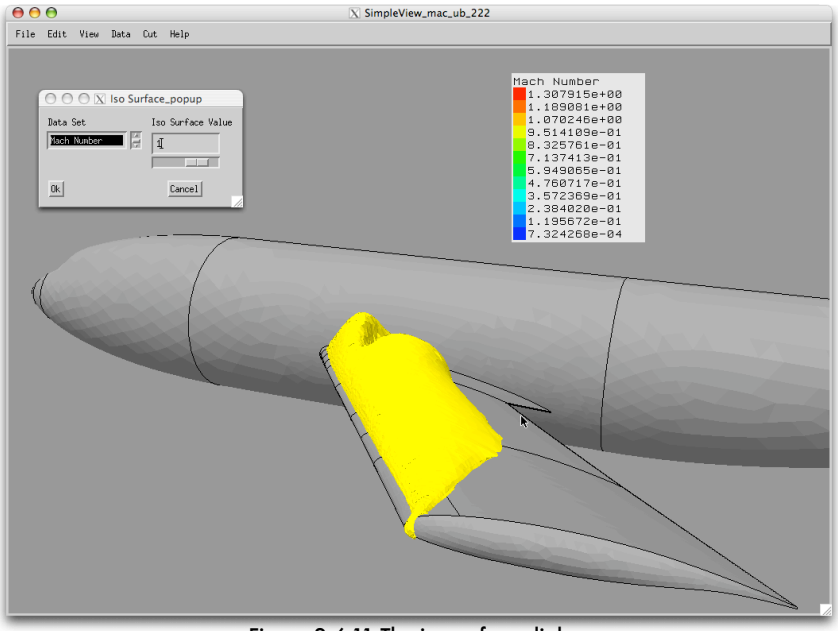


Figure 2.6.11 The iso-surface dialog
(with the M=1 iso-surface shown in yellow)

An iso-surface can be generated for any dataset in memory; this means that if you use the default 6 data set option, it will only be possible to generate iso surfaces on one of the derived data sets). If the particular dataset that you want to write out does not appear in the listbox on the left-hand side of the "Iso-Surface" dialog, it indicates that SimpleView has not yet calculated that dataset from the quantities specified in the .flo file. If this is the case, then choose "cancel" from the

dialog, and have SimpleView compute the dataset by selecting the desired dataset from among the choices shown in the "Dataset" menu (found in the "Data" panel, as shown in Figure 2.6.1).

To delete all Iso Surfaces, select the "Clear Iso Surfaces" option from the Data Menu.

Memory Requirement: High CPU: High

Generating and Viewing Field Data Cuts

SimpleView allows you to see the variation of flow quantities on planar "field cuts" through the grid. The process of defining a field cut is almost identical to that described in section 2.5 for defining a "crinkle cut" (for looking at the cells of the volume grid). As with crinkle cuts, you begin by selecting the "Define New Cut..." option from the Cut Menu, which will open the Data Cut Dialog, as shown in Figure 2.6.12.

(a)

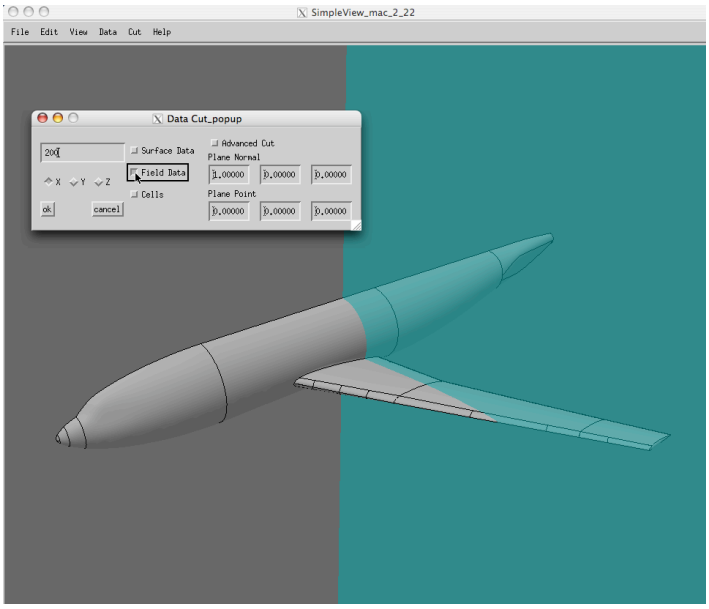


Figure 2.6.12 Field cut definition :
(a) constant-coordinate cut

You then select the "Field Data" option (also shown in Figure 2.6.12), which instructs SimpleView to extract the values of the currently active dataset (Cp, Mach number, etc.) on the cut. As with the crinkle cut definition, you then provide the location of the cuts. Planar field cuts

can be constant-coordinate planes (specified by clicking on the x, y, or z buttons and entering the constant-coordinate value, as shown in Figure 2.6.12(a)), or arbitrary "advanced" cuts (specified by clicking on the "Advanced Cut" button in the Data Cut Dialog and providing the plane normal vector and a point on the plane, as shown in Figure 2.6.12(b)). As you provide these parameters, a transparent blue cutting plane will appear at the specified location.

(b)

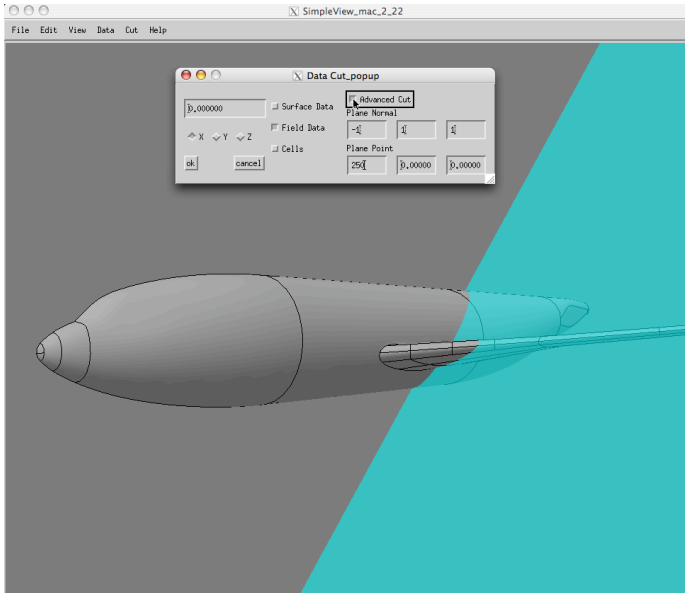


Figure 2.6.12 : Field cut definition :
(b) "advanced" cut

Once you've defined the planar cut (and selected the "Field Data" option), click on "ok." SimpleView will then display a shaded surface contour showing the variation of the current dataset on the defined plane, as shown in Figure 2.6.13. If you are unsure as to what the currently active dataset is, you should activate the display of the data legend by selecting "Data Legend" from the "Legends" panel of the "View" menu. You can then change the displayed dataset by selecting another dataset from within the "Dataset" panel of the "Data" menu (as shown in Figure 2.6.1); choosing another dataset from that list will update the display of the shaded surface contour, as well as the displayed data legend.

Choosing the "Show Planar Cut Wireframe" option allows you to examine the density of the grid on the planar cut by displaying all of the points on the cut onto which the solution was interpolated (in order to display the contour). An example of this is shown in Figure 2.6.14.

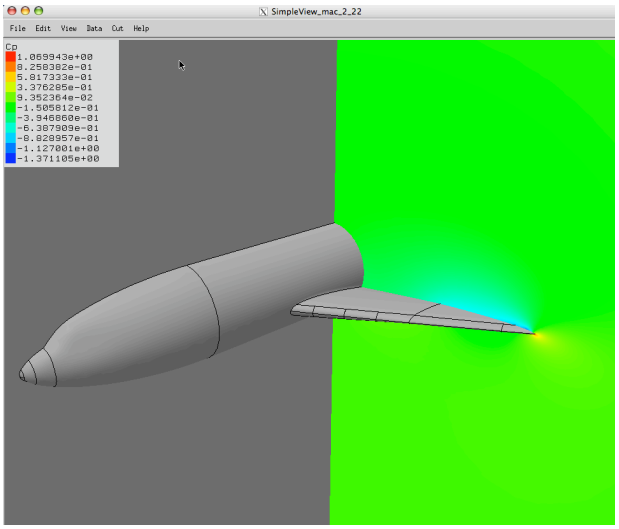


Figure 2.6.13 Data display on field cut

Keep in mind that the planar cut wireframe and the planar cut (solution) itself are separate objects, the display of which can be individually toggled by checking or unchecking the "Show Planar Cut" and "Show Planar Cut Wireframe" options in the "Cut" menu.

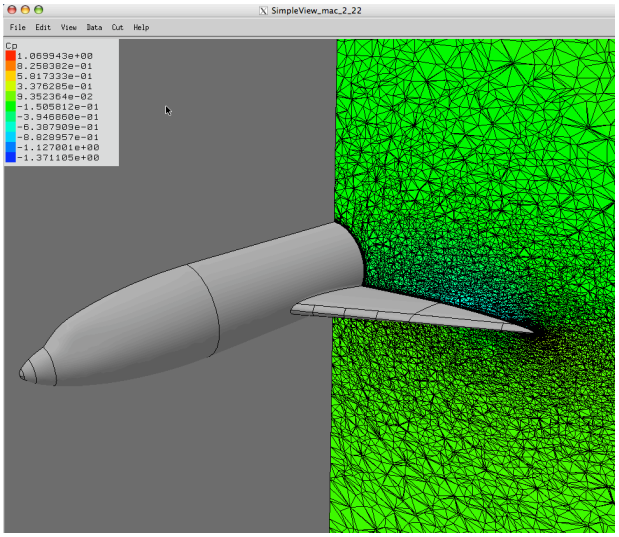


Figure 2.6.14 Display of planar cut wireframe

Multiple cuts of any kind can be generated and displayed, but be aware that each cut remains in memory until you clear it, so you may eventually run out of memory.To clear a planar cut, choose "Clear Planar Cut" from the "Cut" menu.

If the "Planar Cut Cutoff" option is selected from within the "Cut" menu, any portion of the planar cut(s) with data values outside the currently-defined data range will be transparent. This option can be useful if you want to focus attention on areas containing high gradients in a particular dataset; an example of this is shown in Figure 2.6.15.

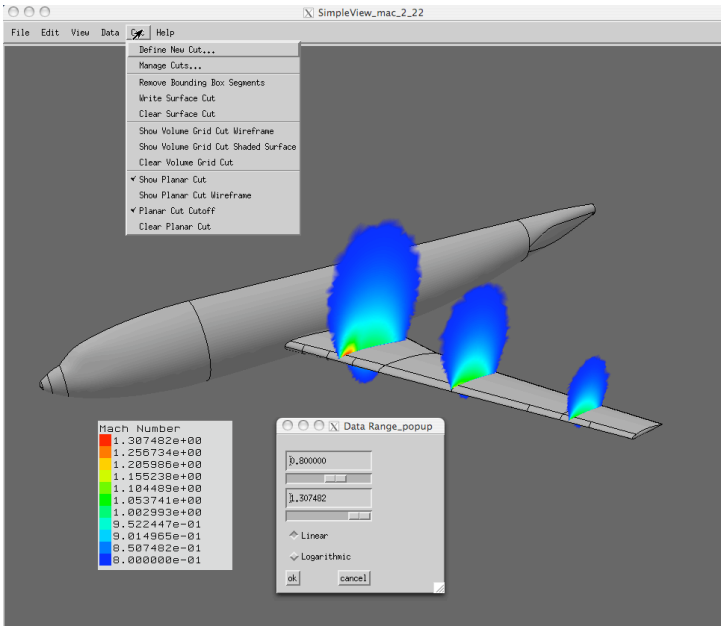


Figure 2.6.15 "Planar cut cutoff" example

Memory Requirement: High CPU: Moderate

Generating and Viewing Field Data Contour Lines

SimpleView can also display the variation in a flow quantity on a planar cut as contour lines. After defining the location of a planar data cut as described in the preceding section, select the "Planar Cut Contour Lines" option from the "Cut" menu to display the planar contour lines.

Planar cut contour lines work exactly like surface contour lines, except that there can be multiple sets of planar cut contour lines displayed (at the different user-defined locations, as shown in Figure 2.6.16).

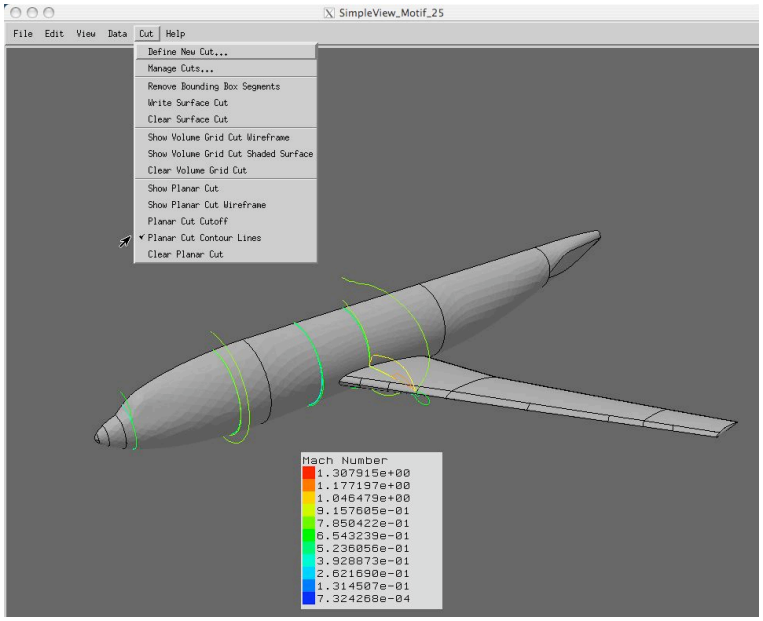


Figure 2.6.16 Planar cut contour lines at various streamwise locations

Memory Requirement: High CPU: Moderate

Viewing USM3D Problem/First-Order Cells

The flow solver USM3D can be prompted to write out several diagnostic files meant to assist you in assessing the convergence and quality of the solution. Among these are two files which specify the location of potential "problem areas" in the grid. The "cells.1storder" file specifies the cells in the grid at which USM3D has calculated a pressure value that is less than the minimum allowable pressure value specified in the USM3D input file. The "cells.problem" file specifies the cells in the grid at which the mass flux residual is 100 times the corresponding average value in the domain. SimpleView can read these two files, and display the location of the cells specified in each.

After loading a full cogsg file, load the cells.1storder or cells.problem files by selecting them from the "Load..." panel of the File menu. Then, select the "1st Order Cells" or "Problem Cells" option from the "View" menu to toggle display of these cells. By default, 1st Order Cells are displayed in yellow and Problem Cells are displayed in red (as shown in Figure 2.6.17).

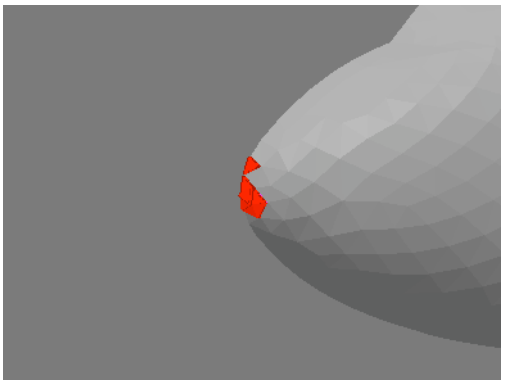


Figure 2.6.17 Displaying a "problem cells" file

As noted in section 1.4, the "cells. 1storder" and "cells.problem" files are diagnostic files written by USM3D only as needed and requested by the user (by specifying the appropriate value for the "idiagnos" parameter in the USM3D control input file).

Memory Requirement: Low CPU: Low

3 RUNNING BCFRONTTEST AND CHECKLAYERS

3.1 Finding "Zero-Layer Faces" with bcFrontTest

In section 2.4, the process of using SimpleView to automatically locate and display zero-layer faces (at the completion of the advancing-layers stage of grid generation) was discussed. There are times when you might want to perform this operation on the grid files without using SimpleView--for instance, if you have a batch script that you use for generating the grid, or when dealing with cases where the grid is so large that having SimpleView perform the operation might take a substantial amount of time. In those situations, you can use the "bcFrontTest" code--an external command-line code that you use in the "Vol_AL" directory to perform this operation and write out a file containing zero-layer face information (which you can then read in to SimpleView). To run bcFrontTest, start the bcFrontTest executable from the command line in the directory containing the files you want to test, and enter the Project Name when prompted; bcFrontTest will then print the output to the screen (an example is shown in Figure 3.1.1).

```
Project Name: dlrf6_vwf
frontFile: [dlrf6_vwf.front]
number of bc faces:             39050
number of NOSLIP bcfaces:       23716
19 zero-layer face(s) found
```

Figure 3.1.1 Screen output of bcFrontTest

In the example whose output is shown in Figure 3.1.1, bcFrontTest has detected the existence of 19 surface faces that ought to be part of advancing-layers type cells, but are not (i.e., faces that ought to have advancing-layers type cells directly above them, but do not).

bcFrontTest will also write the zero-layer face information to a file called "zlf.txt" in the current directory.This file can be read into SimpleView, as described in the following section.

Using SimpleView to Display the "Zero-Layer Faces" Found by bcFrontTest

After loading the surface grid into SimpleView (in the directory containing the "zlf.txt" file), select the "Zero-Layer Faces" option from the View Menu, as shown in Figures 3.1.2.

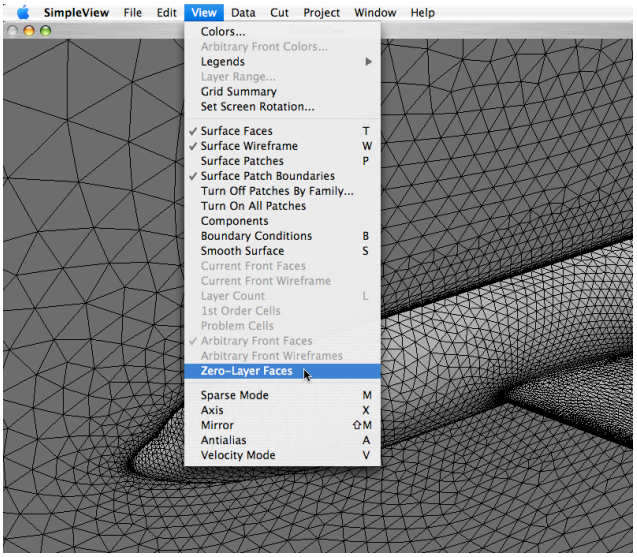


Figure 3.1.2 Reading the "zlf" file into SimpleView

SimpleView will automatically load the "zlf.txt" file, and will set the center of rotation to the first Zero-Layer face in the file (if there are any). Any patch containing Zero-Layer faces will be displayed in yellow, and the Zero-Layer faces themselves will be displayed in red, as shown in Figures 2.4.2 and 2.4.3. If there is no "zlf.txt" file in the current directory, Simple View will open a file selector, allowing you to load a "zlf.txt" file located in a different directory. Cancel out of the file selector if there is no appropriate "zlf.txt" file.

Memory Requirement: Low CPU: Low

3.2 Determining Layer Count with CheckLayers

As discussed in the VGRID User Guide, the generation of viscous-type grids involves the specification of parameters governing the size of the advancing-layer type cells, and the rate at which they grow. In the course of the grid generation sequence, VGRID checks the integrity of

the grid; among other things, VGRID checks the volume of every cell it has generated to ensure that the volume of each is positive (i.e., that there are no collapsed or crossed cells). When VGRID determines that a cell has a "negative volume." it simply removes the cell, and leaves it to the advancing-front algorithm to complete the grid. Furthermore, there is nothing to prevent VGRID from removing "negative volume" cells all the way to the surface of the configuration itself. As a result, you really do not know exactly how many layers VGRID has built on any given part of the configuration. While it is not critical to know how many layers VGRID built in every area of the configuration, it is very useful to be able to know if VGRID was able to successfully generate the number of layers you expected (based on your specification of the first-layer spacing and spacing growth rates) in regions of critical interest. Once you have generated your grid, you can use the "CheckLayers" utility to determine the exact number of layers that VGRID built over each surface triangle, and then use SimpleView to view the results.

To get a layer count, simply run the CheckLayers utility in the appropriate subdirectory (the "Vol_AF_PG" subdirectory shown in Figure 1.4.1) and enter the project name when prompted. CheckLayers will calculate the number of viscous layers generated over each face, and write the information to a file called *project_name*.layers. The messages that CheckLayers prints out to the console window as it runs are shown in Figure 3.2.1.

```
Project name? flap
21362 faces read
10683 points found
Found 13663 viscous faces
Found 6873 viscous points
Allocating 333309 tetrahedra
Unflipped
Reading 0 tetrahedra from cogsg file
Reading 333309 tetrahedra from cogsg file
Done reading cogsg file.
Sorting surface faces
Searching surface faces
Printing surface faces
Minimum face: 10372 (1 layers)
```

Figure 3.2.1 Screen output of CheckLayers

Note that, as shown in Figure 3.2.1, CheckLayers will identify the face number of the surface triangle corresponding to the location with the minimum number of layers (face number 10372, in Figure 3.2.1). You can then use SimpleView to locate and display the particular face.

Using SimpleView to Display the Layer Count Found by CheckLayers

If you would like to know precisely how many advancing-layers VGRID has generated over any surface face, you can load the output of the CheckLayers utility into SimpleView. You do this by loading the .layers file produced by CheckLayers after having loaded a surface or volume grid, as shown in Figure 3.2.2. Since the .layers file essentially contains surface information (i.e., the layer count per viscous surface face), you can load in either the "full" volume grid, or just the surface grid.

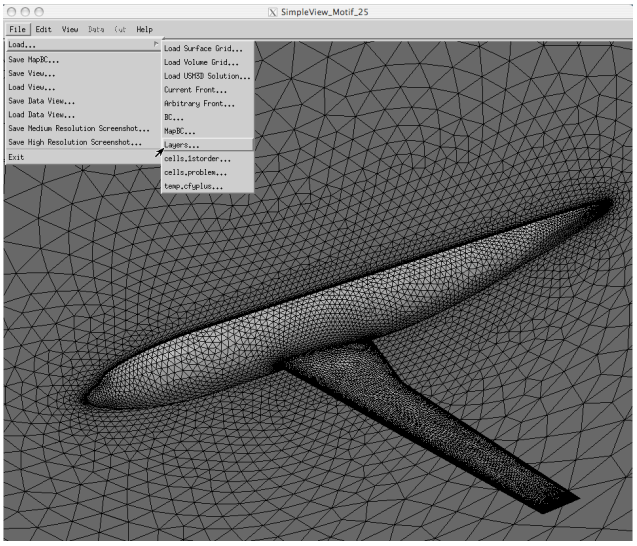


Figure 3.2.2 Loading a .layers file into SimpleView

SimpleView will automatically display the layer count as a shaded surface with the colors representing the different layer count levels, as shown in Figure 3.2.3. Layer Count colors are always scaled from zero to the maximum layer count; orange and red almost always indicate problem areas in the grid, as they indicate areas where the layer count is very low (with red indicating zero layers). To toggle the layer count display, select the Layer Count option from the

View Menu. To toggle display of the Layer Count legend, select the Layer Count Legend option from the Legends Submenu.

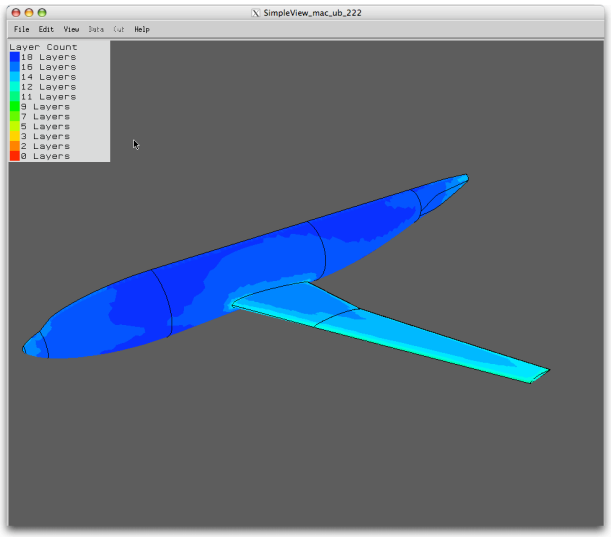


Figure 3.2.3 Layer count display

You can then bring up the Layer Range dialog box (by hitting the 'k' hotkey in the Linux variant, or choosing the "Layer Range" option from the "View" menu panel in the Mac variant). Specifying minimum and maximum values (in the upper and lower fields, respectively) allows you to view only the specified range of layer values. For example, if you wanted SimpleView to display only the areas where there are 15 layers or less, you would type in zero for the minimum value, and 15 for the maximum; this is shown in Figure 3.2.4 (note that the scaling is not changed).

If you have read in the entire grid (i.e., if you loaded the "full" volume grid), and then read in the .layers file produced by CheckLayers, SimpleView has all the data describing the complete volume grid; you can get a "snapshot" of the grid statistics by clicking on the "Grid Summary" option in the "View" menu. An example of the data provided in that summary is shown in Figure 3.2.5. Included in the summary is the minimum number of layers, which can be used with the layer range controls to locate the areas containing the fewest layers.

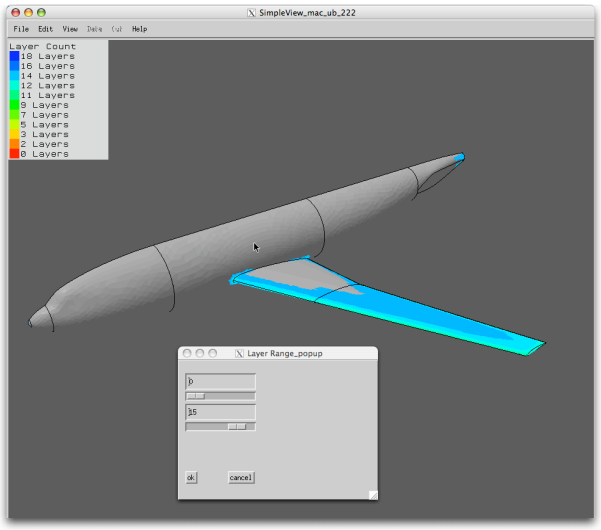


Figure 3.2.4 Layer range dialog
(showing areas with 15 layers or less)

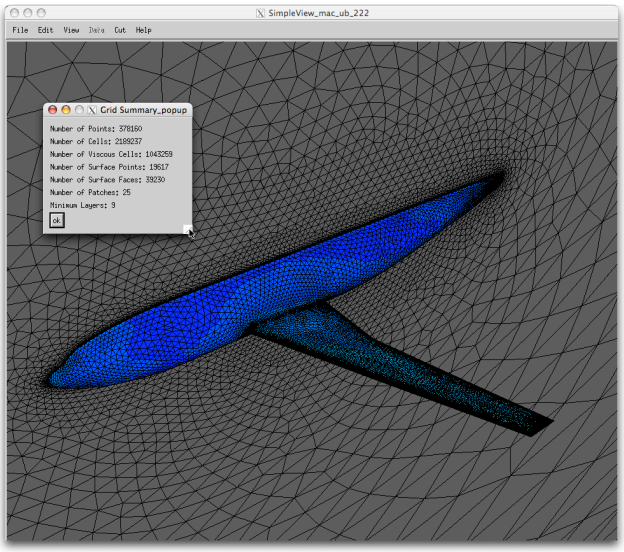


Figure 3.2.5 Grid summary panel

Memory Requirement: Low CPU: Low

4 OTHER USEFUL FEATURES IN SIMPLEVIEW

4.1 Using SimpleView to Check for "Bad Edges"

One characteristic of a "valid" surface mesh is that each and every edge of every surface triangle is shared by two (and only two) surface triangles. If an edge is shared by more (or less) than two triangles, problems can arise with the grid. It can be a relatively subtle and insidious error; in fact, VGRID will happily generate such a surface mesh, without checking for the existence of such so-called "bad edges." If you are going to continue on and generate a viscous mesh, VGRID will then perform an automatic check to determine if any such edges exist (as they often lead to problems determining valid vectors along which to construct advancing-layer cells). If you are going to be generating an inviscid mesh, however, VGRID does not perform any such test, and this can often lead to unexpected results. You can use SimpleView to detect the existence of any such "bad edges." Because these are edges of surface triangles, this check should be performed at the completion of the surface mesh generation stage (in the "Surface" directory, as outlined in Figure 1.4.1).

After loading the surface mesh into SimpleView, select the "Check Edges" option from within the "Edit" menu, as shown in Figure 4.1.1.

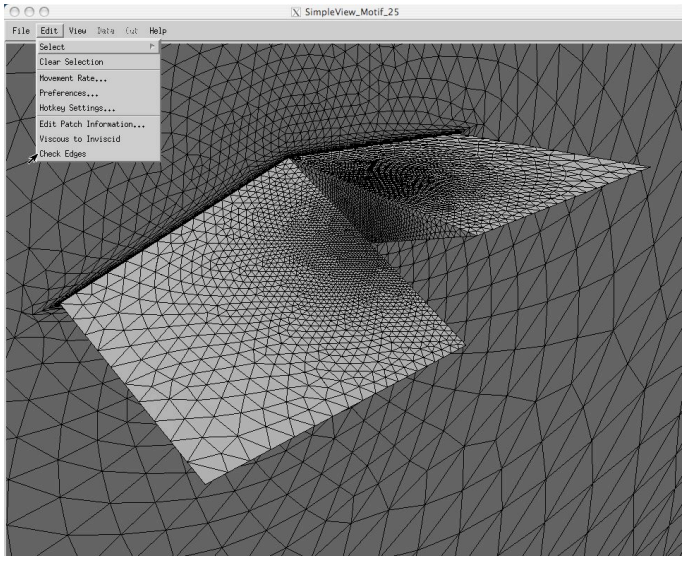


Figure 4.1.1 Selecting the "Check Edges" option

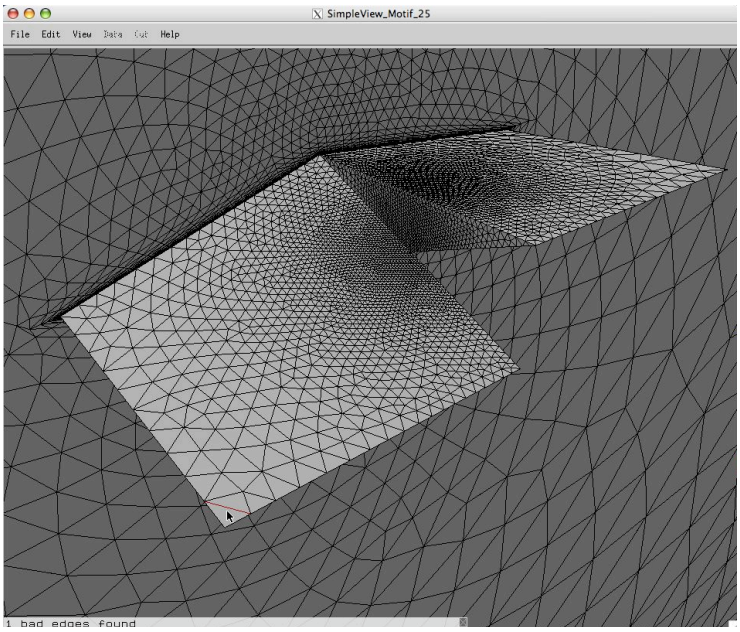


Figure 4.1.2 The result of the "Check Edges" operation

SimpleView will then check for the existence of any "bad edges." Figure 4.1.2 illustrates the result of having found one "bad edge"--SimpleView highlights the edge in red, and sets the center of rotation to the center of the edge. Note that the message " 1 bad edges found" also appears in the dialog box on the lower left corner of the screen.

It should be noted that the "Check Edges" algorithm that is currently implemented is relatively slow (a more efficient version is being developed for inclusion in a future release of SimpleView).

If any "bad edges" are identified by SimpleView, you generally have two options for correcting the problem:

1. you can use the "swap edge" procedure in VGRID to correct the problem, or
2. you can adjust your patching to partially constrain the manner in which VGRID generates the surface triangles in the particular region of the surface mesh, and ensure that the problem will not occur again. This alleviates the need to manually perform the "swap edge" procedure in VGRID to correct the problem every time you generate a new surface mesh on the configuration.

Memory Requirement: High CPU: High

APPENDIX I : FILE FORMATS

FORTRAN pseudo-code is provided for the VGRID .mapbc, .bc, and .cogsg files.

```
**********************************************************************
1)  project.mapbc (created by GridTool): Patch/flow-boundary-condition
 file

                    ----------------------------
       parameter(mpatch=***)
 c
       integer bcpch(mpatch)
       integer nsurf(mpatch)
       integer nsurfs(mpatch)
       character*1 text(80)
 c
       open(13,file='project.mapbc',form='formatted')
 c
       rewind 13
       read(13,900)text
       read(13,900)text
       read(13,900)text
       read(13,900)text
       do 1 ipatch=1,npatch
       read(13,*)ipatch,bcpch(ipatch),(nsurf(i), i=1, nsurfs(ipatch))
  1    continue
 c
   900 format(80a1)
                   ----------------------------

   where:
      npatch = number of surface patches defining the geometry
      bcpch = flow boundary condition assigned to each surface patch
      nsurf  = the surface number of the surface
      nsurfs = the number of surfaces associated with surface patch
ipatch
**********************************************************************
```

```
**********************************************************************
 2)  project.bc (created by VGRID/POSTGRID): patch/surface-triangle
file

                    ----------------------------
       parameter(mbf=***)
 c
       integer kfac2(mf),kfac3(mf)
       integer kfac4(mf),kfac5(mf)
 c
       open(unit=12,file='project.bc',form='formatted')
 c
       write(12,91)nbf,nb1,npatch,igrid
       write(12,*)'Triangle   Surface Patch       Nodes'
       do 92 if=1,nbf
       write(12,93)if,kfac2(if),kfac3(if),kfac4(if),kfac5(if)
  92   continue
  91   format(4i8)
  93   format(5i8)
                    ----------------------------

   where:
         nbf = number of boundary triangular faces
         nb1 = number of surface grid points along patch boundaries
      npatch = number of surface patches
       igrid = 1 for inviscid grids; 2 for viscous grids
   kfac2(if) = surface patch number
   kfac3(if) = node 1 of face if
   kfac4(if) = node 2 of face if
   kfac5(if) = node 3 of face if


**********************************************************************
```

```
**********************************************************************
 3)  project.cogsg (created by VGRID/POSTGRID): grid coordinates and
                                                 connectivity file

                    ----------------------------
       parameter(mp=***, me=***)
 c
       integer int(me,4)
       real*8  crd(mp,3),t

 c
       open(10,file='proj.cogrd',form='unformatted',iostat=ios,
     &      err=555,status='old')
 c
       write(10)inew,ne,np,nb,npv,nev,t,
     &            ((int(ie,in),ie=1,ne),in=1,4)
       write(10)(crd(ip,1),ip=1,np),
     &             (crd(ip,2),ip=1,np),
     &             (crd(ip,3),ip=1,np)
       write(10) 0
 c
  555  continue
                    ----------------------------

   where:
      int = connectivity array
      crd = coordinates array
       ne = total number of cells (tetrahedra)
       np = total number of grid points (including nb)
     inew = a dummy variable
       nb = number of grid points on the boundaries
      npv = number of grid points in the viscous layers
            (for Euler grids = 0)
      nev = number of cells in the viscous layers
            (for Euler grids = 0)
        t = a dummy variable

   *Note:  coordinates for "nb" boundary points are written first.


**********************************************************************
```