

GASPer VERSION 5.3.0 REFERENCE GUIDE

AeroSoft, Inc.
2000 Kraft Drive, Suite 1400
Blacksburg, VA 24060-6363

(540) 557-1900
(540) 557-1919 (FAX)
<http://www.aerosoftinc.com>

Table of Contents

1	Getting Started	11
1.1	Overview of Steps	11
1.2	Installing <i>GASPex</i> on Linux Platforms	12
1.2.1	Setting necessary environment variables	12
1.2.2	Generating Your Machine's HostId	14
1.2.3	Requesting a License Key	14
1.2.4	Installing the License Key	15
1.2.5	Running the License Manager	15
1.2.6	Documentation and Tutorial Information	15
1.3	Installing <i>GASPex</i> on Windows	17
1.3.1	Downloading the Software	17
1.3.2	Downloading on Windows	17
1.3.3	Setting Necessary Environment Variables	18
1.3.4	Generating Your Machine's HostId	18
1.3.5	Requesting a License Key	19
1.3.6	Installing the License Key	19
1.3.7	Running the License Manager	19
1.3.8	Documentation and Tutorial Information	19
1.3.9	Running the <i>GASPex</i> Flow Solver	20
2	What's New In <i>GASPex</i>5.3	23
3	Using <i>GASPex</i>	25
3.1	AeroSoft Directory Structure	25
3.1.1	The bin directory	25
3.1.2	The etc Directory	26
3.1.3	The share Directory	26
3.2	The RLM License Server	27
3.3	The <i>gaspex</i> Binary	30
3.3.1	Running The <i>GASPex</i> GUI	31
3.3.2	Running The <i>GASPex</i> Flow Solver	32
3.3.3	Running The Database GUI	34

3.3.4	Running The Replace Grid Utility	34
3.3.5	Running The Replace Solution Utility	36
3.3.6	Running The Batch Output Utility	37
3.3.7	Running The Update Version Utility	38
3.4	<i>GASPer</i> Environment Variables	38
3.5	Solver Interrupt Signals	39
3.6	The <code>units.txt</code> File	40
3.7	Overview of Running a Problem	42
3.8	Fluid Flow Problems	44
3.8.1	Unstructured Grids	44
3.9	Solid Material Problems	45
3.10	Conjugate Heat Transfer Problems	46
3.11	Helpful Information on Select Topics	46
3.11.1	Solution Limiting Options	46
3.11.2	Plot3D Solution Import	47
3.11.3	Steady State Global Problems	47
3.11.4	Time Accurate Methods	48
3.11.5	Time Accurate Simulations	49
3.11.6	Turbulence Simulations	49
3.11.7	Detached Eddy Simulation	49
3.11.8	Chemistry Simulations	50
3.11.9	Space Marching	50
3.11.10	Overlapping Grid Simulations	51
3.11.11	Grid Generation Issues	51
3.11.12	Axi-Symmetric Problems	51
4	<i>GASPer</i> Menu Bar	53
4.1	The Pull-Down Menus	53
4.2	The File Pull-Down Menu	53
4.2.1	The File/New Menu Item	53
4.2.2	The File/Open Menu Item	54
4.2.3	The File/Save Menu Item	54
4.2.4	The File/Save As Menu Item	55
4.2.5	The File/Close Menu Item	55
4.2.6	The File/Generate HostId	55
4.2.7	The File/Import Menu Item	56
4.2.8	The File/Recover Archive Menu Item	61
4.2.9	The File/Compress Grid File	62
4.2.10	The File/Compress Soln File	62
4.2.11	The File/Quit Menu Item	62
4.3	The Commands Pull-Down Menu	62
4.4	The Wizards Pull-Down Menu	65

4.5	The View Pull-Down Menu	65
4.6	The Options Pull-Down Menu	66
4.7	The Tabs Pull-Down Menu	67
4.8	The Help Pull-Down Menu	67
5	The Main Frame	69
5.1	Chemistry and Thermodynamics Database	69
5.2	Unit System	70
5.3	Grids Directory	70
5.4	Solution Directory	71
5.5	Reference Quantities	71
5.6	Reference Pressure	72
6	The Graphics Window	73
6.1	Introduction	73
6.2	Manipulating The View	73
6.2.1	Rotation	73
6.2.2	Translation	74
6.2.3	Zooming	74
6.3	Interactive Picking Objects	74
6.3.1	Single Select	75
6.3.2	Multiple Select	75
6.3.3	Toggle Select	75
6.3.4	Cycle Select	75
7	The Tree View	77
7.1	Introduction	77
7.1.1	Zones	77
7.1.2	Zonal Boundaries	79
7.1.3	Surfaces	79
7.1.4	Solution Visualization	80
7.1.5	File Output	80
7.2	Tree Node Properties	80
7.2.1	The Render Property	81
7.2.2	The Color Property	82
7.2.3	The Line Width Property	83
7.2.4	Use Parent	83
7.3	Manipulating the Tree	83
7.3.1	Expanding and Collapsing Tree Nodes	83
7.3.2	Renaming a Tree Node	83
7.3.3	Selecting Tree Nodes	84
7.3.4	Moving Tree Nodes	84
7.4	Pop-Up Menus	85

7.4.1	The Edit Menu	85
7.4.2	The Drop Selected Menu	86
8	The Zones Frame	87
8.1	Introduction	87
8.2	Zone Initialization	88
8.2.1	Zone Name and Dimensions	89
8.2.2	Soln For and Initial Physical Model and Q Specification	89
8.2.3	When Does Initialization Occur	89
8.3	Sequencing	90
8.3.1	Zone Name and Dimensions	90
8.3.2	Structured Mesh Sequencing Levels	91
8.3.3	Unstructured Mesh Sequencing Levels	92
8.4	Grid Management	93
8.4.1	Table Parameters	94
8.4.2	Delete Grids Option	95
8.4.3	Replace Grids	95
8.4.4	Transform Grids	95
8.4.5	Delete Wall Distance	95
8.4.6	Display Grid Info	96
8.5	Solution Management	96
8.5.1	Soln For Physical Model	96
8.5.2	Table Parameters	97
8.5.3	Average From Finest Sequence	97
8.5.4	Interp from Coarser Sequence	97
8.5.5	General Interp	97
8.5.6	Insert Solution	98
8.5.7	Initialize Solution	98
8.5.8	Delete Solns	99
8.5.9	Import Solns	99
8.6	Editing Surfaces	99
8.6.1	Combine Surfaces	100
8.6.2	Splitting Surfaces	100
8.7	Interp Coefs	106
8.8	Zone Algorithms	106
8.8.1	Grid Adaptation for Unstructured Soln	107
8.8.2	Shock Fit Algorithm	111
8.8.3	Solution Interpolation	115
9	The Zonal Boundaries Frame	117
9.1	Introduction	117
9.2	Zonal Boundaries Tabs	117

9.3	Point-to-Point Zonal Boundaries	118
9.3.1	Computing Singular Lines	118
9.3.2	Introduction to Zonal Boundaries	119
9.3.3	Automated Point-to-Point Zonal Boundaries	120
9.3.4	Creating Zonal Boundaries Manually	121
9.3.5	Setting Zonal Boundary Parameters	122
9.3.6	Checking Zonal Boundaries	123
9.4	The Chimera Algorithm	123
9.4.1	The Importance of Surface Groupings	124
9.4.2	Hole Cutting	124
9.4.3	Hole Filling Algorithms	127
9.4.4	Computing Chimera Coefficients	134
10	The Physical Models Frame	147
10.1	Introduction	147
10.2	Parent/Child Relationships	148
10.3	Physical Models Select List	149
10.4	Creating a New Physical Model	149
10.5	Copying or Cutting a Model	150
10.6	Summary Information	151
10.7	Q Specification	151
10.7.1	The Q Spec Select List	152
10.8	Pointwise Data	153
10.8.1	Creating a Pointwise Data Set	154
10.8.2	Pointwise Data Types	155
10.8.3	Entering Pointwise Data From a File	155
10.8.4	Connection to Boundary Conditions	158
10.8.5	Pointwise Select List	158
10.9	Boundary Conditions	159
10.9.1	Boundary Condition Selection	159
10.9.2	Split Versus Full Flux Setting	159
10.9.3	Q Source Settings	160
10.9.4	Q Spec and Q Time Settings	161
10.9.5	BC Data Settings	161
10.10	Changing Solution Variables	161
10.10.1	General Procedure for Changing the Solution	162
11	Physical Models: Navier-Stokes	165
11.1	Introduction	165
11.2	Thermodynamics and Chemistry	166
11.2.1	Chemistry Model Scroll List	166
11.2.2	Number of Species in the Model	167

11.2.3 Charge Balance	167
11.2.4 Artificial Ignition	168
11.2.5 Chemistry Mode	168
11.2.6 Thermodynamics Model	168
11.2.7 Condensation Model	169
11.2.8 McBride Warnings	170
11.2.9 Available Chemistry Models With <i>GASPer</i>	170
11.3 Inviscid Information	173
11.3.1 Global/Marching Strategy	173
11.3.2 Inviscid Flux Schemes	174
11.3.3 Spatial Accuracy for Structured Data	177
11.3.4 Spatial Accuracy for Unstructured Data	181
11.4 Viscous Information	181
11.4.1 Viscous Flux Mode	182
11.4.2 Viscous Flux Terms (Structured Mesh)	183
11.4.3 Cross-Derivative Terms (Structured Mesh)	183
11.4.4 Viscous Gradient Method (Unstructured Mesh)	183
11.4.5 Laminar Transport Properties	184
11.4.6 Wall-Gradient Calculation	187
11.4.7 Turbulence Parameters	187
11.5 Q Specification	195
11.6 Q Time	197
11.6.1 Creating a Q Time Data Set	198
11.6.2 Reading Q Time Data From A File	198
11.6.3 Q Time and Boundary Conditions	200
11.6.4 Q Time and Turbulence	200
11.6.5 Q Time Select List	200
11.7 Pointwise Data	201
11.7.1 Pointwise Data Types	201
11.8 Boundary Conditions	201
11.8.1 List of Fluid Flow Boundary Conditions	202
11.8.2 List of Turbulent Boundary Conditions	207
11.8.3 Special BC Data	207
11.9 Buoyancy Sources	210
11.10 Rotation Src	211
11.11 Coupled PhysMods	212
11.12 User Src	212
11.13 Changing Species	214
11.13.1 Changing Non-Equilibrium Vibrational Energies	214
11.13.2 Changing Turbulence Models	214

12 Physical Model: Solid Thermodynamics	217
12.1 Introduction	217
12.2 Solver Overview	217
12.3 Material Property	218
12.4 Q Spec	218
12.5 Pointwise Data	219
12.6 Boundary Conditions	221
12.6.1 List of Solid Material Boundary Conditions	221
12.7 User Src	222
13 Physical Model: Lagrange Spray	225
13.1 Introduction	225
13.2 Solver Overview	225
13.3 Boundary Conditions	226
13.3.1 List of Spray Boundary Conditions	226
13.4 Coupled PhysMods	226
13.5 Particle Src	227
13.6 Constraints	229
13.7 ThermoChemistry	230
13.8 Particle Dynamics	231
13.9 Output Files	231
14 Physical Model: Mechanical Stress	233
14.1 Introduction	233
14.2 Solver Overview	233
14.3 Material Property	234
14.4 Q Spec	235
14.5 Boundary Conditions	236
14.5.1 List of Solid Material Boundary Conditions	236
14.6 Coupled PhysMods	237
15 The Run Definitions Frame	239
15.1 Run Definitions Select List	239
15.1.1 Creating a New Run Definition	240
15.1.2 Cutting a Run Definition	241
15.1.3 Copying a Run Definition	241
15.1.4 Pasting a Run Definition	241
15.2 Main Run Definition Tab	241
15.2.1 Run Settings	241
15.2.2 Convergence Information	243
15.2.3 Solution Backup Information	243
15.2.4 Temporal Information	244
15.2.5 Time Dependent Information	246

15.2.6 Solution Averaging	246
15.2.7 File Information	247
15.2.8 Marching Information	247
15.3 Time Integration Run Definition Tab	247
15.3.1 Solver Algorithm	248
15.3.2 Inner Iteration Scheme	249
15.3.3 Time Step Selection	250
15.3.4 CFL Basis	251
15.3.5 Automated Time Step Limiting	252
15.4 Solver Info Run Definition Tab	252
15.4.1 Solver Info Select List	253
15.4.2 Navier-Stokes Solver Info	253
15.4.3 Solid Thermodynamics Solver Info	256
15.4.4 Lagrange Spray Solver Info	256
15.4.5 Mechanical Stress Solver Info	256
15.5 Run Group Run Definition Tab	257
15.5.1 Run Group Zone Family	257
15.5.2 Run Group Solver Info	258
15.5.3 Run Group Physical Model	258
15.5.4 Run Group Reference Q Spec	258
15.5.5 Run Group Cycle Information	258
15.5.6 Run Group File Info	259
15.5.7 Run Group Implicit Zonal Boundaries Setting	260
15.5.8 Run Group Select List	260
15.6 Algorithms Run Definition Tab	260
15.7 Physical Resources Run Definition Tab	261
15.7.1 Resource Table	261
15.7.2 Host Name and Host Number	262
15.7.3 Execution String	262
15.7.4 Scale Factor	262
15.7.5 Processor Memory	263
15.7.6 Number of Processors	263
15.8 Decomposition Run Definition Tab	263
15.8.1 Decomposition Type	264
15.8.2 Creating a Decomposition	266
15.8.3 Implicit Partition Boundaries	267
15.8.4 Number of Partition Boundaries	269
15.8.5 Setting Partition Values Manually	270
15.8.6 Solver Decomposition	270
15.8.7 Reported Information	271

16 The Solution Visualization Section	273
16.1 Introduction	273
16.2 The VTK Pipeline	273
16.3 The Visualization Folder	275
16.4 The Data Node	275
16.4.1 Solution Variables	276
16.4.2 Sequence	280
16.4.3 The Grid Range Property	280
16.4.4 The Solution Mapper	289
16.5 The Contour Node	292
16.5.1 The Contours Property	294
16.6 The Vector Node	295
16.6.1 The Vectors Property	297
16.7 The Cutter Node	298
16.8 The Clip Volume Node	300
16.8.1 The Clip Volume Property	301
16.9 The Streamtracer Node	302
16.9.1 The Streamtracer Property	303
16.10 The Displacement Node	306
16.10.1 The Displacements Property	308
16.11 The Line Source Node	309
16.11.1 The Line Source Property	310
16.12 The Plane Source Node	311
16.12.1 The Plane Property	311
16.13 The Transformation Node	312
16.13.1 The Transformation Property	314
16.14 The Text Node	316
16.14.1 The Text Property	316
16.15 The Logo Node	318
16.15.1 The Logo Property	318
16.16 The Lagrange Src Node	319
17 The File-Output Section	321
17.1 Introduction	321
17.1.1 What variables to output?	321
17.1.2 Where in the domain to output?	322
17.1.3 When to output the data?	322
17.1.4 In what format should the data be output?	322
17.2 Output Nodes	322
17.2.1 Creating Output Nodes	323
17.2.2 Folder/Node Properties	325
17.3 The Export Property	325

17.3.1 Output Type	325
17.3.2 The Output Button	336
17.3.3 The Enable Batch Option	337
17.3.4 The Output From Solver Section	337
17.4 The Variables Property	339
17.4.1 Available Variables Filtered By Option	339
17.4.2 The Active Variables List	353
17.5 The Grid Range Property	354
17.5.1 The Range Type Selection Widget	355
17.5.2 The Interpolation Type Selection Widget	356
17.5.3 The Include Cell Types Toggle Buttons	358
17.5.4 The I, J and K Range Widgets	360
17.5.5 Incremental Viewing	362
17.5.6 Viewing of Boundary Ghost Cells	362
17.5.7 Output of True Boundary Values	363
18 Database GUI	365
18.1 Displaying the GUI	365
18.2 Primary GUI Window	366
18.2.1 File Menu Bar	366
18.2.2 Options Menu Bar	366
18.2.3 Tree View Frame	367
18.2.4 Data View Frame	368
18.3 Species	368
18.3.1 Gas Species Main Frame	369
18.3.2 Gas Species Summary	371
18.3.3 Solid Species Main Frame	378
18.3.4 Solid Species Summary	382
18.3.5 Liquid Species Main Frame	386
18.3.6 Liquid Species Summary	388
18.4 Reactions	394
18.4.1 Creating a Reaction	395
18.4.2 Entering the Reaction Equation	396
18.4.3 The Main Reaction Frame	397
18.4.4 Rate Equation Types	397
18.4.5 Entering Rate Data	408
18.4.6 Final Word on Reactions	410
18.5 Models	410
18.5.1 Creating a Model	411
18.5.2 Gas Models	411
18.5.3 Solid Models	412
18.5.4 Liquid Models	414

Chapter 1

Getting Started

1.1 Overview of Steps

The following list provides a general overview of the steps needed to install *GASPer*.

1. Downloading and installing the software
2. Setting any necessary environment variables
3. Generating your machine's hostId
4. Requesting a license key
5. Installing the license
6. Running the license manager
7. Documentation and tutorial information

The *GASPer* software package can be installed on either linux or windows based platforms. Depending on the platform, the user will download either specific linux files or windows files. The specifics of each are explained in the following sections.

Note: If a *GASPer* license has already been installed on the local network, the user can skip steps 3 to 6. For installations where the license server is run on a different machine, the `aerosoft/etc/keys/gasp.lic` file should be modified to reflect the correct license server host. This is done using a text editor and changing "localhost" to the license server host name.

1.2 Installing *GAS*Pex on Linux Platforms

For linux installation, the user will need to install a common (platform independent) file archive, and a binary (platform specific) file archive. Both the common file archive and platform file archive help create the `aerosoft` directory (for more information about the directory structure, see Chap. 3 beginning on pg. 25).

The common file archive contains files that are independent of platform (*i.e.*, operating system and hardware). Specifically, the common file archive has files related to the following:

- The thermo-chemical database
- GTK user interface resource files
- Tutorials
- Documentation

The platform file contains executable binary files that were compiled specifically for a certain type of hardware architecture and operating system. The executables in the platform file archive normally consist of the `gaspex` executable and the license server executables. The specific use of the binaries is described in Chap. 3 beginning on pg. 25.

In addition to the common file archive and platform file archive, there may also be an MPI file archive to download and install. This file will contain specific binaries and support files related to the MPI package against which *GAS*Pex was compiled. This is sometimes available when *GAS*Pex is statically built against the MPI package (only select packages are statically built).

Both the common and platform file archives can be downloaded from the AeroSoft web site (www.aerosoftinc.com). Once downloaded, the archives can be installed by running the following command on each download archive (shown here is the common file archive).

```
tar -xzf common.tgz
```

Once the downloaded archives have been installed, they will create the `aerosoft` directory. The directory name can be renamed as desired.

1.2.1 Setting necessary environment variables

Setting the AEROsoft_HOME variable

*GAS*Pex v5 requires the user to identify the default location of a number of dependent files. *GAS*Pex does this by reading the environment variable `AEROsoft_HOME`.

Note:

Simultaneous installation of GASP Version 5.3 and a previous version of GASP, such as Version 5.2 is supported with the addition of a version specific environment variable, `GASP53_HOME`. Version 5.3 will first attempt to read the `GASP53_HOME` variable in order to locate the default location of the installation. If not defined, then the executable will search for the `AEROSOFT_HOME` variable. If the user wishes to maintain an older Version of GASP, set the location of the older Version using the `AEROSOFT_HOME` variable, and set the location of Version 5.3 using the `GASP53_HOME` variable.

The recommended location for linux installation of the *GASPE* distribution is in `/opt`, but this is not a requirement. For the remainder of this chapter, all references to the file structure will be with respect to the recommended installation location.

```
setenv AEROSOFT_HOME /opt/aerosoft
```

Note:

The above linux command is suitable for use with a `csh` or `tcsh`. Please see the on-line man pages for assistance with alternate command shells.

Setting the `OPAL_PREFIX` variable

Binaries that are linked against static, custom-built versions of OpenMPI may require that the user set an additional environment variable (`OPAL_PREFIX`) in order to properly identify the location of the MPI distribution. For example, if the *GASPE* distribution is in `/opt`, then the location of the OpenMPI distribution provided by AeroSoft will be `/opt/aerosoft/openmpi-x.x.x` where `x.x.x` is the specific version number of the OpenMPI distribution. The user should set the environment variable as below.

```
setenv OPAL_PREFIX /opt/aerosoft/openmpi-1.4.3
```

Note:

If the `OPAL_PREFIX` environment variable is not properly set, then the user may receive an error with respect to an `ORTE_ERROR_LOG` when attempting to run the flow solver.

Setting the execution path

For linux users, the `gaspex` binary should be added to the users execution path. This can be done by adding the `/opt/aerosoft/bin` directory to the users execution path as shown below.

```
setenv PATH "$AEROSOFT_HOME/bin:$PATH"
```

or

```
set path=($AEROSOFT_HOME/bin $path)
```

Note:

On some GNU based unix systems, there is a program called *gasp*, which is a preprocessor for assembly programs. In particular, this binary is distributed by many early Linux systems. This *gasp* executable may be in the default path, and will be called instead of the AeroSoft executable, if it appears before the AeroSoft directory. AeroSoft advises its users to place the */opt/aerosoft/bin* directory at the beginning of the path environment, so that the proper *gaspx* executable is called.

1.2.2 Generating Your Machine's HostId

New to Version 5.1 of *GASPEX* is the RLM (reprise license manager) software. This is now the default license manager for all supported platforms of *GASPEX*. Similar to other licensing software, the RLM software supports both node-locked and floating licenses. In most cases the user will be supplied with a floating license (one machine runs the license server which allows other machines on the local network to access the license). For a floating license setup, the user needs to select a machine that will run the license server.

In order for AeroSoft to create a license file for the user, information needs to be supplied to AeroSoft by the user. This information can be obtained in one of two ways:

1. Open the *GASPEX* GUI by running *gaspx*. Once open, go to the menu bar located at the top of the GUI and select the File menu. In the File menu select **Generate HostId**. Follow the instructions to generate a *hostId* file and send it to AeroSoft (see the next section). Be sure to run this from the machine that will serve as the license server.
2. From an open shell, run the command *rlmutil rlmhostid* located in the *\$AEROSOFT_HOME/bin* directory. The *rlmutil rlmhostid* program will print a *hostId* for the machine. Be sure to run this from the machine that will serve as the license server.

1.2.3 Requesting a License Key

The *hostId* must be generated using either the *GASPEX* GUI or running *rlmutil rlmhostid* (see the previous section). Once the *hostId* is generated, a license key can be requested by mailing the *hostId* file (or output from the command line) to support@aerosoftinc.com. When sending the *hostId* file, please include your name, organization and a contact phone number.

1.2.4 Installing the License Key

Once AeroSoft has received your license key request, it will be processed as quickly as possible. AeroSoft will then send the user a license file (`gasp.lic`). Copy this file to the `$AEROSOFT_HOME/etc/keys` directory.

The `$AEROSOFT_HOME/etc/keys` directory may contain a default license file from the installation process. If you are installing the license on the machine used to run the license server, then replace the existing `gasp.lic` file.

For installations where the license server is run on a different machine, the `gasp.lic` file should be modified to reflect the correct license server host. This is done using a text editor and changing “localhost” to the license server host fully qualified domain name.

1.2.5 Running the License Manager

Once the license key is installed, you can start the RLM license manager. For floating licenses, the license manager *must* be active before you can run the *GASPE* flow solver.

To launch the RLM license manager, execute the following command:

```
$AEROSOFT_HOME/bin/rlm -c $AEROSOFT_HOME/etc/keys
```

By default, the RLM server will search for the license file in the same directory where `rlm` is run. The `-c` option is used to specify either the license file or the directory where it resides. If a directory is given, RLM will load all license files with the `.lic` extension. A log file is written to `license.log` by default, which is set using the RLM options file `$AEROSOFT_HOME/etc/keys/aerosoftlm.opt`.

More information on the license server binary (`rlm`) and RLM utility binary (`rlmutil`) can be found in Sec. 3.1.1 on pg. 25. An on-line users guide to RLM can be found at http://www.reprisesoftware.com/RLM_Enduser.html.

1.2.6 Documentation and Tutorial Information

Since you are reading this document, you have found some documentation. However, you may also find documentation for *GASPE* in pdf form in the directory `aerosoft/share/doc`. Note that documentation efforts are on-going, and we will be adding chapters from time to time.

You can find tutorials in the `aerosoft/share/tutorial` directory as follows:

- Duct-1 A simple 2-dimensional single zone duct problem. This problem is designed to be the first tutorial when learning *GASPE* v5.
- Rocket-2 A more complex, multi-zone, 3-dimensional problem. This problem is designed to be the second tutorial when learning *GASPE* v5. *Note:* This problem is over 2.5 million grid points, and requires more file space and computational resources than the Duct-1 tutorial.

- Forebody-3 A simple, 3-dimensional analytic problem. This problem is designed to cover aspects of the space marching algorithm in *GASPEX*.
- Cavity-4 An axi-symmetric high speed blunt body with a cavity located in the nose. This example discusses the techniques used to obtain a time dependent solution about a simple geometry.
- Chimera-5 A complex, 3-dimensional wing-body problem discretized using an overset grid topology. This example discusses aspects of the Chimera zonal boundary set-up.
- Visualization-1 An introduction to solution visualization using the *GASPEX* GUI. This tutorial begins with a completed solution and discusses different visualization features and techniques.

Note:

The Tutorials as distributed in `common.tar.gz` contain only the grid file and documentation. *GASPEX* v5 input decks which are ready to run, as well as completed solutions can be found on our sftp site:
<sftp://gasp53ex@ftp.aerosoftinc.com/outgoing/Tutorial>

g

1.3 Installing *GASPerx* on Windows

1.3.1 Downloading the Software

AeroSoft provides an MPI (message passing interface) version of *GASPerx* for the Windows platform. MPI allows users to run the *GASPerx* flow solver on multiple processors.

Installation of *GASPerx* requires the user to download a single package installer from the AeroSoft sftp or web site. The installer is supported on current 64-bit Windows operating systems. The file for installing will contain the GASP executable and common files and will be named:

`GASP53-x64-Setup.exe`

Once the installer has been downloaded, users should install *GASPerx* by executing the installer package. When installing the package, select the full install when asked. In some cases, the user may need to restart their computer after installing all the files.

Some of the installation packages (like OpenMPI) have options for setting the environment variable for the user. Be sure to select this option during the installation process.

By default, the *GASPerx* files will be installed in the directory AeroSoft located in Program Files. The user will find documentation and tutorial information there as well.

1.3.2 Downloading on Windows

AeroSoft requires all downloads to be performed using the SFTP protocol using a public SSH key. If you are not familiar with this process, the following steps can be followed to assist you. To begin, a user must have a public/private SSH key pair. One way to create this is to use the “Putty” software (free to download and use).

1. Download and install the “Putty Key Generator” software, which contains a key generator that is needed for this task. The Putty software can be downloaded from www.putty.org
2. Run PuTTYgen. Select the “SSH-2 RSA” key type. The number of bits in the generated key should be set to at least 1024
3. Generate a public/private key pair and save both files. Make note of the location of both files. The public key file is to be given out to confirm your identity. The private key file should never be given out, and should be secured using a pass phrase.
4. Send the public SSH key file to aerosoft (email to support@aerosoftinc.com).

Once the public and private SSH keys are generated, the user can use any supported file transfer software to perform SFTP. One such software package is WinSCP. This can be installed and used as follows.

1. Download and install the WinSCP software (free to download and use). The software can be downloaded from www.winscp.net
2. Create and save a new session with the following settings:
 - Host name: ftp.aerosoftinc.com
 - User name: gasp53
 - Private key file: select your private key file generated above
 - File protocol: SFTP
3. Attempt to login to AeroSoft's sftp site. If you receive an error or warning, please contact AeroSoft via e-mail at support@aerosoftin.com and provided the exact time you attempted to login. As an extra layer of security, we only allow access from known IP addresses, and we may need to add your IP address to our list.

1.3.3 Setting Necessary Environment Variables

GASPEX v5 requires the user to identify the default location of some common files. *GASPEX* does this by reading the environment variable **GASP53_HOME**. This environment variable is set automatically during the install. The default location is

C:\Program Files\AeroSoft.

1.3.4 Generating Your Machine's HostId

GASPEX uses the RLM (reprise license manager) software to control use of the flow solver. Similar to other licensing software, the RLM software supports both node-locked and floating licenses. In most cases the user will be supplied with a floating license (one machine runs the license server which allows other machines on the local network to access the license). For a floating license setup, the user needs to select a machine that will run the license server.

In order for AeroSoft to create a license file for the user, machine specific information needs to be supplied by the user. This information can be obtained in one of two ways:

1. Open the *GASPEX* GUI by running *gaspex*. Once open, go to the menu bar located at the top of the GUI and select the File menu. In the File menu select **Generate HostId**. Follow the instructions to generate a hostId file which you will send to AeroSoft (see the next section).
2. From a command shell, run the command **rlmutil rlmhostid** located in the **Program Files\AeroSoft\bin** directory. The **rlmutil rlmhostid** program will print a *hostId* for the machine. Be sure to run this from the machine that will serve as the license server.

1.3.5 Requesting a License Key

The hostId was generated using either the *GASPer* GUI or running `rlmutil rlmhostid` (see the previous section). In either case, a license key can be requested by mailing the hostId file (or output from the command line) to support@aerosoftinc.com. When sending the hostId file, please include your name, organization and a contact number.

1.3.6 Installing the License Key

Once AeroSoft has received your license key request, it will be processed as quickly as possible. AeroSoft will then send the user a license file (`gasp.lic`). Copy this file to the `AeroSoft\etc\keys` directory.

The `AeroSoft\etc\keys` directory may contain a default license file from the installation process. If you are installing the license on the machine used to run the license server, then replace the existing `gasp.lic` file.

For installations where the license server is run on a different machine, the `gasp.lic` file should be modified to reflect the correct license server host. This is done using a text editor and changing “localhost” to the license server host name.

1.3.7 Running the License Manager

Now that you have the installed the license key file, you need to start the RLM license manager. For floating licenses, the license manager *must* be active before you can use the *GASPer* flow solver.

To launch the RLM license manager, execute the following command:

```
C:\Program Files\AeroSoft\bin\rlm -c <path_to_keys>
```

By default, the RLM server will search for the license file in the same directory where `rlm` is run. The `-c` option is used to specify either the license file or the directory where it resides. If a directory is given, RLM will load all license files with the `.lic` extension.

More information on the license server binary (`rlm`) and RLM utility binary (`rlmutil`) can be found in Sec. 3.1.1 on pg. 25.

1.3.8 Documentation and Tutorial Information

You may find documentation for *GASPer* in pdf form at

```
C:\Program Files\AeroSoft\share\doc\gasp_usersguide.pdf
```

Note that documentation efforts are on-going, and we will be adding chapters from time to time.

You can find tutorials in the `AeroSoft\share\tutorial` directory as follows:

- Duct-1 A simple 2-dimensional single zone duct problem. This problem is designed to be the first tutorial when learning *GASPEX* v5.
- Rocket-2 A more complex, multi-zone, 3-dimensional problem. This problem is designed to be the second tutorial when learning *GASPEX* v5.
- Forebody-3 A simple, 3-dimensional analytic problem. This problem is designed to cover aspects of the space marching algorithm in *GASPEX*.
- Cavity-4 An axi-symmetric high speed blunt body with a cavity located in the nose. This example discusses the techniques used to obtain a time dependent solution about a simple geometry.
- Chimera-5 A complex, 3-dimensional wing-body problem discretized using an overset grid topology. This example discusses aspects of the Chimera zonal boundary set-up.
- Visualization-1 An introduction to solution visualization using the *GASPEX* GUI. This tutorial begins with a completed solution and discusses different visualization features and techniques.

Note:

The Tutorials as distributed contain only the grid file, and documentation. *GASPEX* v5 input decks which are ready to run, as well as completed solutions can be found on our sftp site:
<ftp://gasp53ex@ftp.aerosoftinc.com/outgoing/Tutorial>

1.3.9 Running the *GASPEX* Flow Solver

The solver must be run from the command line. You can access a Windows command line prompt from the Start Menu/Run/cmd or Start Menu/All Programs/Accessories/Command Prompt.

There two ways to run the flow solver. The first is using the auto solver method. This method is intended to take care of the MPI calls and solve on the run definitions specified in the GUI. If your environment is set up correctly, you can run the flow solver with

```
<path-to-gasp>\gasp.exe --solve -i input-deck.gsp
```

The second method to run gasp is more manual and is normally used with batch scripts. Because *GASPEX* is compiled with OpenMPI, the mpirun command must be used to execute the *GASPEX* flow solver. In this case, the solver is executed with the MPI command and looks like the following

```
mpirun -np $ <path-to-gasp>\gasp.exe --mpi -i input-deck.gsp --run #
```

where \$ is the number of processors to use and path-to-gasp is typically

C:\Program Files\AeroSoft\bin,

is the run definition to execute. For the manual launch, the user must specify which run definition to execute.

Chapter 2

What's New In *GASPex* 5.3

GASPex Version 5.3 has a number of new features and changes that are discussed here. With each major release (*e.g.*, 5.2 to 5.3), changes to the input deck, grid, and solution files are commonly made. Therefore, users will need to convert older input decks to the current version. This is done by opening the input deck in the GUI and selecting save from the main menu. This will automatically save the input deck to the current version. Note that this will also prevent older versions of *GASPex* from reading the updated input deck. Therefore, it is recommended that users save a copy of older case files before converting to 5.3.

A list of the new features and changes in *GASPex* 5.3 are listed next. A brief description is given with a link to the chapter or section in the manual where more details are provided.

Database Format The *GASPex* physical properties database has been updated. The result is a slight change to the XML based database file. The new version of *GASPex* will still read the older Version 5.2 database files, but the user is encouraged to open older database files in the database GUI and select save. This will convert the older database file to the new file format. The database GUI can now import reactions from Chemkin files with limited support. See Chap. 18 beginning on pg. 365 for more details regarding the database GUI.

Database GUI The database GUI can now be launched from within the *GASPex* GUI. This allows users to more quickly access the database when setting up or using the GUI.

Structured/Unstructured *GASPex* now supports coupled solutions between structured and unstructured meshes. Multi-zone grids containing a combination of both grid types (and point-to-point zonal boundaries) is now supported by *GASPex*. Full mesh import can be done via the CGNS grid format or zones can be imported independently.

User Defined Output Variables User defined output variables can now be used to represent any pre-defined output variable. This allows users to build output expressions using any of the output variables in *GASPex*. Prior versions of *GASPex* supported a limited set of user defined variables.

Global Variables Global variables are now supported for every integer or real based numeric input. Global variables may also be set up as cycle or time dependent variables. See Sec. 4.6 on pg. 66 for more details.

CGNS Output File output can now be sent to CGNS format files. This includes options for the grid, solution, or both.

New BC Dialog Boundary conditions are now set using a dialog. The new dialog allows the user to filter boundary conditions based on categories, such as inflow, outflow, solid walls, etc. A short description of each BC is also provided.

Special BC Data Unique data for boundary conditions has been moved from the **Q Spec** to the **Boundary Conditions** section. As a result, there is now a “BC Data” selection in the boundary condition table.

Crank Nicolson Time Integration A new time integration scheme has been added which is the second order Crank Nicolson method. The set up and application of the scheme is very similar to the Dual-Time Stepping option.

Conservative Interpolation A new second order conservative interpolation algorithm has been implemented. This allows the solution from one set of zones to be interpolated to another set in a conservative manner.

Zone Algorithms A new component has been added the *GASPex* solution framework called the “Zone Algorithm”. The “Zone Algorithm” operates on a collection of Zones either at user controlled events during the solution process, or from within the GUI. Current Zone Algorithms include structured grid shock fitting, grid adaptation for unstructured grids, and conservative solution interpolation. Note that while some algorithms existed in previous versions of *GASPex*, the unique algorithms have been refactored to derive from a single Zone Algorithm Framework. See Sec. 8.8 on pg. 106 for more details.

Chapter 3

Using *GASPer*

3.1 AeroSoft Directory Structure

Installation of the *GASPer* Version 5 software (see Chap. 1 beginning on pg. 11) will create a directory called `aerosoft` (linux platforms) or `(AeroSoft)` (windows platforms). This directory contains all the files for the *GASPer* Version 5 software package. Upon installation, the `aerosoft` directory may contain most, if not all, of the following directories and files.

- bin** Directory containing the executables provided by AeroSoft. At a minimum, this directory will contain the *GASPer* executable, and executables associated with the license manager. Depending upon the distribution, executables associated with the MPI environment may also be included.
- etc** Directory containing local configuration files. This directory is not intended to be shared across multiple platforms.
- share** Directory containing files intended to be stored on a single host or shared among multiple hosts. Included in this directory are documentation, tutorials, chemistry database, image files used by the GUI and customization files.
- lib** Directory containing library files for the GUI. This directory will appear on Windows installations only.

3.1.1 The bin directory

This directory contains all the Version 5 executable binary files. This includes the RLM license manager files and the *GASPer* Version 5 binary (`gaspex`).

In *GASPer*, the different components of the application suite are accessed from a single executable. This one executable (`gaspex`) will run the GUI, flow solver, and chemistry/thermodynamic database GUI. Information on how to use the `gaspex` executable can be found in Sec. 3.3 on pg. 30.

For the RLM license manager, three executable files are included with *GASPEX*. The `rlmutil` executable is used to perform command-line operations such as generating the hostId and checking the license status. The `rlm` executable is the license server daemon, and must be run before the flow solver will operate properly. Finally, the executable `aerosoftlm` is used in conjunction with the `rlm` binary and must always be present when running the license server, but should never be executed directly. It contains the licensing information for AeroSoft products such as *GASPEX*. More information on using the license server can be found in Sec. 3.2 on pg. 27.

3.1.2 The etc Directory

The `etc` directory contains a variety of files and sub-directories that represent site customizable configuration files that are specific to the *GASPEX* Version 5 software. The files in `etc` are not intended to be shared among hosts.

The `gtkrc` file may be used to change the fonts and colors used by the GUI. For further assistance modifying this file, please contact AeroSoft technical support.

The `keys` directory is the default location for storing the license manager keys for the RLM license manager. If the host machine is not responsible for running the license server, a license file (*e.g.*,`gasp.lic`) may still be necessary to point to the host running the server.

3.1.3 The share Directory

The `share` directory contains files intended to be stored on a single host or shared among multiple hosts. Included in this directory are documentation and tutorials as well as customization files.

All of the data associated with physical models is stored in a user customizable database file. The default database file distributed with *GASPEX* is the file `thermochemdb.tcd`. The database file name is specified in the main input deck for *GASPEX*. Because of this, the user can copy and change the location or name of this file. The *GASPEX* executable will, by default, search for the database file `$(AEROSOFT_HOME)/share/thermochemdb.tcd`, so always leave a copy of the default file in place.

The XML database file can be edited using the *GASPEX* database GUI. More information on this is given later in the chapter, but in short, the user may run the database GUI using the following command: `gaspex --dbasemgr`.

The `doc` directory contains the *GASPEX* users manual, as well as a technical manual. The users manual is updated from time to time, so the files in this directory should be updated with each major release. The current *GASPEX* manual can also be viewed online at <http://www.aerosoftinc.com>.

Each release of *GASPEX* contains a file named `doc/ReleaseNotes`. The `ReleaseNotes` file contains the release notes for the current version of *GASPEX*. The file contains a brief description of every change that was made to the software since the last major release.

The `tutorial` directory contains a number of sample problems. These tutorials are designed to introduce the user to various features of the software. AeroSoft recommends completing the `Duct-1` and `Rocket-2` tutorials first. They assume the user has no experience with *GASPE* and go step by step in setting up a problem. A description of each tutorial is given below.

Duct-1 A two-dimensional, viscous, single zone problem which solves flow through a transonic duct. This tutorial is recommended for first time users.

Rocket-2 A three-dimensional, viscous, multi-zone problem which solves a rocket/nozzle problem. This tutorial introduces the user to multiple zone problems and parallel processing.

Forebody-3 A three-dimensional, space-marching problem which solves supersonic flow about an analytic forebody geometry. This tutorial introduces the user to space marching.

Cavity-4 An axi-symmetric, time accurate problem which solves a forward facing cavity. The freestream has a sinusoidal oscillation, which is amplified inside the cavity. This tutorial introduces the user to the time accurate features of Version 5.

Chimera-5 A three-dimensional, Chimera problem which solves flow about a wing/body geometry. The problem contains multiple overlapping grids used to defined and resolve the flow for a wing and body combination. This tutorial introduces the user to the Chimera feature.

Visualization-1 A tutorial introducing the user to solution visualization within the GUI. The tutorial covers topics such as contours, iso-surfaces, stream traces, cutting planes and movies.

The `units.txt` file contains additional unit system definitions. More on the units file can be found in Sec. 3.6 on pg. 40.

The `images` directory contains PNG files that the GUI uses. These files should not be edited or removed. Images that the user may wish to use for solution visualization is found in the `logos` directory.

3.2 The RLM License Server

Once the user has installed a license file (see Chap. 1 beginning on pg. 11 for more information on the license file), the RLM server can be launched for floating licenses. Node-based licenses do not require the license server daemon to run. For detailed information about RLM, the RLM end-user manual can be accessed online at http://www.reprisesoftware.com/RLM_Enduser.html.

The license server is run using the command `rlm`. The server handles requests from clients and directs them to the aerosoft server. The rlm server also provides status to the various utilities, when appropriate. The server initiates a reread of the license files every day at midnight.

The rlm server is delivered with an embedded Web Server to perform normal administration tasks. This is accessed by pointing a web browser to `http://localhost:5054/home.asp`. The 'localhost' should be replaced with the license server machine host name. The default port number is 5054, which may need to be changed if another application is using that port (see the `-ws` option below to change the port value).

Below is a list of some of the options for the executable.

```
rlm [-nows | -ws <port>] [-x [rlmdown] [rlmremove]] [-c <lic file path>]
     [-dat] [-dlog <debug log path>]
```

- **-nows**

Instructs the rlm server to not start the embedded web server

- **-ws <port>**

The rlm server binds to the web server port 5054 by default. If this port is already being used, the user will need to instruct the server to use another port. This is done using this option.

- **-x rlmdown | rlmremove**

Option controls whether the rlmdown and/or rlmremove commands will be processed by the server. Specifying only `-x` will disable both commands. Specifying either command name after the `-x` will disable just that command

- **-c <lic file path>**

Specifies which license file or directory to use. This option overrides the setting of the RLM_LICENSE environment variable.

- **-dat**

Specifies that license files should have the extension `.dat`, rather than `.lic`. With this option the rlm server will search for all files ending `.dat` only.

- **-dlog <debug log path>**

Specifies the pathname for the server debug log. If logfile is preceded by the '+' character, the logfile will be appended, otherwise it is overwritten.

The web server may be used to monitor and perform a number of administrative tasks related to the license server. While this is the recommended method by RLM, the user may also perform these same tasks from the command line or a script using `rlmutil`. The `rlmutil` binary can be used to perform the following:

```
rlmutil program <args>
where program is one of:
rlmdebug rlmdown rlmhostid rlmnewlog rlmremove rlmreread rlmstat rlmswitch
rlmswitchr

args are:
-c license_file | port@host - to specify license
-dat - use *.dat instead of *.lic for license file
-h - print usage information and exit
-q - don't prompt on rlmdown/rlmremove/rlmswitch(r) and don't check
license checksums
-v - print version and exit
```

- **rlmutil rlmdebug [product]**

Prints information about the specified product (in this case gasp). The debugging information is written to stdout (this requires running in a command window on Windows systems).

- **rlmutil rlmdown -q [isv]**

Shuts down the first license server in the license path RLM_LICENSE. If the -q option is specified, the shutdown happens without a confirming prompt. If the optional isv (individual software vendor) is specified (in this case use 'aerosoftlm'), only that ISV's server is shut down.

- **rlmutil rlmhostid -q [isv]**

Prints the hostid of the machine it is running on. If the -q flag is specified, the hostid is printed without any other output.

- **rlmutil rlmnewlog isv new-log-file-name**

Causes the ISV server isv (*i.e.*,aerosoftlm) to move the current reportlog output to new-log-file-name, and continue logging to the original filename.

- **rlmutil rlmremove [-q] server-host port isv handle**

Removes a checked-out license. If the -q option is specified, the license is removed without a confirming prompt. server-host, port, and handle are indicated in the rlmstat output.

- **rlmutil rlmreread [-q] server-host port isv handle**

Causes the specified ISV server isv (*i.e.*,aerosoftlm) to reread it's license file, and options file if specified in the license file (or if in the default location isv.opt). If isv is omitted, the reread command is sent to rlm and all ISV servers.

- **rlmutil rlmstat [-a] [-i [isv]] [-l [isv]] [-n [host]] [-p [product]] [-u [user]]**
Retrieves status from the license servers and prints it. The -a option prints the status from rlm and all ISV servers.
- **rlmutil rlmswitch [isv] new-log-file-name**
Causes the server isv (*i.e.*,aerosoftlm) to close the current debug log file and begin output to new-log-file-name.
- **rlmutil rlmswitchr [isv] new-log-file-name**
Causes the ISV server isv (*i.e.*,aerosoftlm) to close the current reportlog file and begin output to new-log-file-name.

3.3 The gaspex Binary

The gaspex binary file has multiple functions. These range from running the graphical user interface (GUI), running the flow solver, running the database manager GUI, or running batch utilities for tasks such as replacing grids. The gaspex binary file, when used without any options, will run the GUI. With certain options, the executable will run other applications. The full list of options for the gaspex binary file is given below.

gaspex options are:

Short Options:

```
-g ..... runs graphical interface
-s ..... runs flow solver
-i ..... specifies name of input deck
-b ..... performs output in batch mode
-e ..... specifies AEROSOFT_HOME directory
-v ..... prints GASPEX version
-h ..... prints help message
```

Long Options:

```
--gui ..... runs graphical interface
--solve ..... runs flow solver (auto launch)
--mpi ..... runs flow solver (manual launch)
--dbasemgr ..... runs database manager graphical interface
--decomp ..... perform auto decomp if efficiency less than target
--batchoutput .... performs output in batch mode
--replacegrid .... replaces grid (non graphical)
--replacesoln .... replaces solution (non graphical)
--updateversion... updates XML input deck to current file version
```

```
--version ..... prints GASPEX version  
--help..... prints help message  
  
--fpe ..... enable floating point exceptions (Linux only)  
--run # ..... execute given run (used with --mpi)  
--rlm ..... specifies the license file or directory location  
--var ..... specifies a global variable input value (format  
is --var NAME=#)
```

These options can be displayed from the command line by typing:

```
gaspx --help
```

A more detailed description, along with some examples of using the gaspx binary file will be given in the sections that follow. For the following examples, we will assume that an input deck has been created with the file name `deck.gsp`.

3.3.1 Running The *GASPEX* GUI

If the user wants to run the GUI, then one of the following commands can be issued from the command line:

```
gaspx  
or  
gaspx -g  
or  
gaspx --gui
```

All of the above commands for running the GUI are equivalent and will open up the *GASPEX* GUI for input deck creation. In order for the GUI to run successfully, the GUI must be able to locate a complete and correct database file. Assuming that the AEROSOFT_HOME environment variable has been set correctly, the GUI will search for the database file

`$(AEROSOFT_HOME)/share/thermochemdb.tcd` by default.

Now assume that an input deck exists (`deck.gsp`) and that the user would like to run the GUI and open the input deck simultaneously. To do this, one of the following commands would be issued:

```
gaspx -i deck.gsp  
or  
gaspx -g -i deck.gsp  
or  
gaspx --gui -i deck.gsp
```

Again, all of the above examples for running the GUI are equivalent. The GUI will be started and the input deck will be opened and displayed from the GUI. The file extension for the input deck is assumed to be `.gsp`. It is recommended that the user not use any other file extension when naming the input deck.

3.3.2 Running The *GASPEX* Flow Solver

There are two ways to run the *GASPEX* flow solver. We will refer to these two ways as auto launch and manual launch. Most often the user will use the auto launch commands to run the solver, but there are times when the manual launch must be used. The auto launch will be discussed first, followed by the manual launch option.

Auto Launch Method

GASPEX uses MPI to perform multi-processor runs for the flow solver. Because MPI commands are usually different for each platform, the auto launch method for running the flow solver hides the MPI aspect of running the binary. Auto launch works by starting up a `gaspex` process which acts as a controller. This process reads the input deck and determines what the user wants to execute (as specified in the input deck). This information includes the number of processors and the executable string, which can be unique for each run definition (for more information on Run Definitions see, Sec. 15 on pg. 239).

For example, the user may set up a problem to run on three grid sequence levels. In doing so, three run definitions would be created. The first run definition may use one processor, the second run definition may use 4 processors, and the third run definition may use 10 processors. The flow solver would be started (assuming auto launch) which would start the `gaspex` controller process. The controller would build the argument list to run the first run definition and make a system call to run the flow solver for the first run definition. When that run definition is complete, the controller would make another system call to run the second run definition and so on. With the auto launch method, a total of two `gaspex` processors would run on the root processor. One would correspond to the controller process and the second would correspond to the actual flow solver.

Now that we have discussed how the auto launch method works, it is time to give some command examples. Assuming a problem is set up and ready to run, the user executes one of the following commands to run the flow solver.

```
gaspex -s -i deck.gsp
or
gaspex --solve -i deck.gsp
```

The above commands are equivalent and will auto launch the flow solver. *GASPEX* will issue the correct MPI commands (based on the platform binary) and execute each run

definition that was specified to run in the input deck. This option for running the flow solver is recommended when available (on certain platforms it is not supported).

There are several command-line options that may be used when running the flow solver in auto launch mode. The user can view these commands by typing the following command:

```
gaspx --solve --help
```

An example of using all the available commands for the auto launch mode is given below.

```
gaspx --solve -i deck.gsp -e /Path/To/AEROSOFT_HOME --rlm portlicense.server  
--fpe
```

In the above example, the -e option specifies the location of the \$(AEROSOFT_HOME) directory. This can be used instead of setting the AEROSOFT_HOME environment variable. The --rlm option specifies a specific machine for the solver to search for the license or points to the directory or file that contains this information. This can be used instead of setting the RLM_LICENSE environment variable (which is not always required). The last option, --fpe, is used to run with floating point exceptions turned on. With this option, the flow solver will immediately stop if a division by zero is encountered (or if some other invalid arithmetic operation occurs).

There are several platforms where the auto launch mode (--solve or -s) is not available. In those cases, the user must launch the flow solver in manual mode. Another reason for using the manual launch mode is for batch jobs. There are times when the user needs to handle the MPI commands inside a batch routine. The manual launch method is discussed next.

Manual Launch Method

The manual launch method for running the *GASPx* flow solver has two main differences over the auto launch method. The first is that the user must supply the MPI arguments at the command line. These may be different depending on the platform that *GASPx* is run on. The second difference is that only one run definition can be run with a single execution of *GASPx*. So with each execution of the flow solver, the user must specify which run definition to operate on. With the manual launch of the flow solver, there will no longer be a gaspx control process running on the root processor.

The manual launch method is only recommended if there are problems with the auto launch or if using a queue system. A summary of the options available for this method can be seen by running

```
gaspx --mpi --help
```

which gives a command line help message. What follows is an example of running the flow solver in manual launch mode.

```
mpirun -n 2 gaspex --mpi -i deck.gsp --run 2
```

The above command uses the MPI command `mpirun` to start up an MPI job. The exact command to launch MPI will vary depending on the platform. In this example, `mpirun` is used for most linux platforms running OpenMPI. The `-n 2` option specifies that MPI will use 2 processes. Again this is an MPI option which will be platform dependent. The remaining arguments are specific to the `gaspex` binary and are platform independent.

The `--run` option (only used with `--mpi`) specifies which run definition to execute. In this example the second run definition is performed. If the `--run` option was not included, the solver would default to running the first run definition.

With the manual launch method, there are certain overrides in the input deck. Specifying which run definition to be executed is one of them. Even if the input deck was set up not to run the second run definition, the command `--run 2` will cause the second run definition to be executed anyway. Another override is the executable string in the input deck. This is used by the auto launch method, but not by the manual launch.

3.3.3 Running The Database GUI

The database manager GUI is used to create and/or edit the chemistry database. The database manager will create an XML file which is used by the *GASPE* GUI and flow solver. A database file comes with the *GASPE* software and is located in the `aerosoft/share` directory. The database file is called `thermochemdb.tcd`.

The database GUI is executed with the following command:

```
gaspex --dbasemgr
```

This will launch the database GUI and will contain an empty database. To run the GUI and load an existing database simultaneously, use the following command.

```
gaspex --dbasemgr -i thermochemdb.tcd
```

3.3.4 Running The Replace Grid Utility

From the GUI, *GASPE* allows the user to perform a number of operations on the grid and solution. One such operation is the ability to replace the geometric grid coordinates (x,y,z). Given the same dimensions, a zone may be replaced with another, thus allowing the geometric values to change. The user may perform this operation from inside the GUI or from the command line. In this way, the grid can be updated from within a batch job. This is sometimes beneficial for optimization problems and parametric studies.

The command line options are given by typing

```
gaspex --replacegrid --help
```

which produce the following output.

```
Usage for GASPEx Replace Grid Utility  
Command Line:  gaspex [options]
```

Long Options:

```
--replacegrid .... calls this utility  
--grid ..... specifies name of grid file (required)  
--seq ..... specifies sequence level to replace  
           (default is 1)  
--zone ..... specifies zone to replace  
           (default is to replace all zones)  
--format ..... specifies file format [ascii,binary]  
           (default is ascii)  
--system ..... specifies grid system [2d,2daxi,3d]  
           (default is 3d)  
--unit ..... specifies unit system  
           (default is Metric)  
--nomultizone .... grid format is not multi-zone  
           (default is multi-zone)  
--blank ..... grid format is I-blanking  
           (default is no blanking)  
--double ..... grid format is double precision  
           (default is single)  
--help ..... prints this help message
```

Short Options:

```
-i ..... specifies name of input deck (required)  
-e ..... specifies AEROSOFT_HOME directory  
-h ..... prints this help message
```

Note that there are three required arguments to use this utility, which are: --replacegrid, -i, and --grid. The other options have defaults which may or may not need to be changed. An example of using the replace grid utility is given below.

```
gaspex --replacegrid -i deck.gsp --grid file.p3d --seq 1 \  
--format ascii --system 3d --unit English --nomultizone --blank
```

In this example, the grid file is named `file.p3d`. The grid is used to replace all the zones for the first sequence level. The grid is ASCII Plot3D format, with units in feet. The grid file is not in Plot3D multi-zone format and there is blanking information.

At this time, the grid file must be of Plot3D format. From within the GUI, the user can replace grids using not only Plot3D format, but also CGNS format files and *GASPE* grid files.

3.3.5 Running The Replace Solution Utility

Similar to the replace grids utility is the replace solution utility. With this utility, the user may replace the solution of one or more zones in the solution file without using the GUI. The user may get command line help by typing

```
gaspx --replacesoln --help
```

which produces the following output.

```
Usage for GASPE Replace Solution Utility
Command Line: gaspx [options]
```

Long Options:

<code>--replacesoln</code> calls this utility
<code>--soln</code> specifies name of solution file (required)
<code>--seq</code> specifies sequence level to replace (default is 1)
<code>--zone</code> specifies zone to replace (default is to replace all zones)
<code>--format</code> specifies file format [ascii,binary,gaspv4] (default is Plot3D ascii)
<code>--system</code> specifies grid system [q,function] (default is q)
<code>--interp</code> specifies interpolation [node,icell,bcell] (default is node)
<code>--unit</code> specifies unit system (default is Metric)
<code>--double</code> grid format is double precision (default is single)
<code>--help</code> prints this help message

Short Options:

<code>-i</code> specifies name of input deck (required)
<code>-e</code> specifies AEROSOFT_HOME directory
<code>-h</code> prints this help message

The required arguments for this utility are: --replacesoln, -i, and --soln. An example of using this utility is given next.

```
gaspx --replacesoln -i deck.gsp --soln file.p3d --seq 1 \
--format ascii --system function --interp icell --unit English
```

The solution file is named `file.p3d` and is an ASCII Plot3D file in function format. The solution file contains information for all the interior cells (`icell`) and is in the English unit system. Note that the unit system is needed even if the input deck is in English units.

3.3.6 Running The Batch Output Utility

Solution information can be output to file in a number of ways. The most common is to set up a file output that will be performed while the solver is running. Another way to perform file output is from within the GUI. The third way to perform file output is from the command line using what is called batch output. For more information on file output, see Sec. 17 on pg. 321.

Batch output is useful when the user would like to use MPI to execute an output that was set up within the GUI. This may be necessary if the output is memory intensive such that the GUI cannot handle the memory requirements. The user may also want to perform the output across the network in which case using the GUI may be slow.

When performing batch output, file output will be performed for those output folders with the **Enable Batch** option selected (found in the Export section of File Output). The batch output command line options are given by typing

```
gaspx --batchoutput --help
```

which produce the following output.

```
Usage for GASPx batch mode output
Command Line: gaspx --batchoutput [options]
```

Long Options:

```
--run ..... specifies which run definition to execute
--help ..... prints the help message
```

Short Options:

```
-i ..... specifies name of input deck (required)
-e ..... specifies AEROSOFT_HOME directory
-h ..... prints this help message
```

The required arguments for this utility are: --batchoutput and -i. If no --run option is used, the first run definition is assumed. An example of using this utility is given next.

```
mpirun -n 8 gaspex --batchoutput -i deck.gsp --run 3
```

3.3.7 Running The Update Version Utility

When a new version of *GASPE* is released, there are times when changes to the XML file structure are made. These are considered major releases in which the version number in the second slot is changed (*e.g.*, v5.2 to v5.3). When this occurs, *GASPE* relies on the GUI to make the necessary updates to the XML files. The GUI will always be able to convert the previous version XML file to the new version. But there are times when the GUI is not available to do this (like on a remote system). In these circumstances, the user may convert the input deck from the command line. This is performed using the following command:

```
gaspex --updateversion -i deck.gsp
```

3.4 *GASPE* Environment Variables

There are a number of environment variables that can be used with *GASPE*. Of all the environment variables, only one is necessary for the user to set. This is the AEROSOFT_HOME environment variable (discussed in the installation chapter). A list of all the environment variables are given below.

AEROSOFT_HOME Specifies the `aerosoft` directory location. This environment variable is required to run *GASPE*.

GASP53_HOME Overrides the AEROSOFT_HOME environment variable. This is useful if the user has an older version of *GASPE* installed and would like to have two separate `aerosoft` directory locations, one for each version. In this case, AEROSOFT_HOME should point to the older version while GASP53_HOME points to the newer aerosoft directory.

AEROSOFT_FPE If this environment variable is set (does not require a value), *GASPE* will catch all floating point exceptions during the flow solver. When a floating point exception is encountered, *GASPE* will stop execution.

AEROSOFT_NOTMP If this environment variable is set (does not require a value), the GUI will not use temporary files when saving the grid, solution, input deck, or database. The default behavior of *GASPE* is to write files to a temporary file before overwriting the original file. This helps protect files in case of an error during the save.

AEROSOFT_ONE_HOLE_CUT If this environment variable is set (does not require a value), *GASPE*x will override the standard algorithm for directional hole filling. Normally with directional hole filling the algorithm checks how the holes are filled in all 3 directions and then combines them using all 3 directions. This overrides the behavior and sets it so that only one direction needs to have holes in order for a hole to be cut. This can be useful in instances where holes are not being cut when they should be.

AEROSOFT_TWO_HOLE_CUT If this environment variable is set (does not require a value), *GASPE*x will override the standard algorithm for directional hole filling. Normally with directional hole filling the algorithm checks how the holes are filled in all 3 directions and then combines them using all 3 directions. This overrides the behavior and sets it so that only two directions are needed to indicate a hole for a hole to be created. This can be useful in instances where holes are not being cut when they should be.

AEROSOFT_POS_VOL Normally, *GASPE*x will exit immediately upon detecting a negative volume in the grid file. In this situation, the i, j, k location of a single control volume will be printed to the standard output. If the user has inadvertently imported a left-handed grid, then all of the volumes will be negative, and the default behavior prevents excessive printing to standard output. If however, the user wishes to identify all negative volumes, then setting the environment variable before running *gaspex* will prevent the executable from exiting before printing all identified negative volumes.

RLM_LICENSE This environment variable is used to specify the directory or file name for the RLM license file. The default location for this directory is `$AEROSOFT_HOME/etc/keys`. Using this environment variable will override where the solver searches for the RLM license file.

3.5 Solver Interrupt Signals

There are times when the solver is running that the user may want to gracefully shut down the run or force the solver to update the solution file. These operations can be performed by creating a file with a key name in the same directory that the input deck is located. *GASPE*x will search for these files (listed below) each iteration. Below is the list of filenames and their corresponding action if detected.

stopgasp If this file is detected, the solver will shut down after writing the solution to file. This is the recommended way to gracefully stop a *GASPE*x execution and save the solution progress.

writegasp If this file is detected, *GASPE*x will update the solution file at the current cycle. The `writegasp` file will automatically be removed and the solver will continue running.

restartgasp If this file is detected, *GASPer* will write the current solution to file and halt execution for a period of time and then restart. The active processors will not be lost and the license will remain active (checked out). The amount of time between stopping the solver and restarting is default to 60 seconds. This amount can be overridden by setting an integer value in the **restartgasp** file. This signal is useful if the user wants to make changes to the input deck, but not check in a license or exit a job from the queue.

3.6 The units.txt File

GASPer comes with two default unit systems from which the user may select. The two default unit systems are English and Metric (SI units). The unit system is selected from within the GUI.

The **units.txt** contains any additional unit systems that the user may want to define. The format of the file is fixed, such that the user cannot add or remove unit types from the file. A sample **units.txt** file displaying the metric unit system is given below.

```

nUnits nUnitTypes
1          53
UnitSystem
'Metric'
Name      Conversion Factor   UnitStr
None:     1.                  ''
Speed:    1.                  'm/s'
Length:   1.                  'm'
Area:     1.                  'm^2'
Temperature: 1.              'K'
Density:   1.                  'kg/m^3'
Pressure:  1.                  'N/m^2'
Time:     1.                  'sec'
Energy:    1.                  'J/kg'
SpecificHeat: 1.              'J/kg-K'
ThermalConductivity: 1.      'W/m-K'
Viscosity: 1.                  'N-s/m^2'
Epsilon:   1.                  'm^2/s^3'
Omega:    1.                  '1/s'
PerLength: 1.                 '1/m'
Mass:     1.                  'kg'
MomentsInertia: 1.            'kg-m^2'
AngularSpeed: 1.              'rad/s'
Angle:    1.                  'rad'
Force:    1.                  'N'

```

Moment:	1.	'N-m'
MassFlow:	1.	'kg/s'
QDot:	1.	'W/m^2'
Enstrophy:	1.	'1/s^2'
Acceleration:	1.	'm/s^2'
KinematicViscosity:	1.	'm^2/s'
MomentumPerVol:	1.	'kg/m^2-s'
EnergyPerVol:	1.	'J/m^3'
RhoEpsilon:	1.	'kg/m-s^3'
ForcePerLength:	1.	'N/m'
PerTime:	1.	'1/sec'
RhoOmega:	1.	'kg/m^3-s'
Power:	1.	'W'
PerVolume:	1.	'1/m^3'
NucleationRate:	1.	'1/m^3-s'
EnergyPerTime:	1.	'J/kg-s'
HeatTransferCoef:	1.	'W/m^2-K'
ForcePerSpeed:	1.	'N-s/m'
PerTemperature:	1.	'1/K'
TempVariance:	1.	'K^2'
TempVarianceRate:	1.	'K^2/s'
Volume:	1.	'm^3'
RTE:	1.	'W/m^2-Sr'
ElectricFieldStrength:	1.	'N/C'
DensityGradient:	1.	'kg/m^4'
PressureGradient:	1.	'N/m^3'
MomentumResidual:	1.	'kg-m/s^2'
EnergyResidual:	1.	'kg-m^2/s^3'
SpalartResidual:	1.	'kg-m^2/s^2'
EpsilonResidual:	1.	'kg-m^2/s^4'
OmegaResidual:	1.	'kg/s^2'
TurbLResidual:	1.	'kg-m/s'

Customization of the `units.txt` file is optional. The purpose of this file is to add additional unit systems. For example, the metric unit system uses meters as the length scale. If the user needs to use inches (*e.g.*, for reading in a grid that is in inches), a new unit system will need to be defined which uses inches for the length scale. To specify that the length is inches, the user would enter the conversion from inches to meters for the unit length. In the `units.txt` file, the length would appear as:

Length:	0.0254	'in'
---------	--------	------

The conversion is always to metric units, since this is the base unit system for *GAS*Pex. Note that this is why the conversion factor for the metric unit system is always 1.

As AeroSoft continues to add features to the *GAS*Pex software package, additional unit types may need to be added. When this occurs, we will modify the template units.txt file. If you choose to modify the units.txt file, then you will need to update this file for your unit systems, each time we add additional unit types.

3.7 Overview of Running a Problem

This chapter is intended to give the user background information on running a problem with *GAS*Pex. The thought process that goes into a commercial CFD solver will be different with each code. What we hope to accomplish here is to introduce the user to the thought process of *GAS*Pex. To begin, we will introduce the following terms used throughout the manual.

***GAS*Pex GUI** The *GAS*Pex graphical user interface (GUI) is the main tool for setting up a problem. Using the GUI, the user will import grids, describe the problem physics, specify how the problem will be run, and view the resulting solution. The GUI is not used to execute the *GAS*Pex solver. This is performed in batch mode at a command line or by using a script.

The GUI saves the problem setup to a file written in XML format. This XML file contains the majority of information needed to describe and run the solver. A few things that are not stored in the problem XML file include the grid, solution, and database.

Grid A grid is really a discretization of physical space. It can represent a fluid medium such as air or water, or a solid. The grid is the entire computational domain (representing a physical space) in which the user wishes to attain a solution.

The user is responsible for creating a grid and bringing it to the *GAS*Pex solver. A grid is then imported into *GAS*Pex through the GUI. When the problem file is saved, the grid will be saved to its own file in binary format. The binary format that *GAS*Pex uses is an internal format of AeroSoft.

Sequence A sequence is identical to a grid, except that it has been reduced in size by reducing the logical grid dimensions (for structured grids) or total number of cells (for unstructured grids). Once multiple sequences are created, the user may then solve for the solution on different grid sequences (or levels). The process of solving for a solution on a coarse grid sequence and then interpolating the solution to a finer grid sequence is called mesh sequencing. Mesh sequencing helps increase numerical stability and reduce overall convergence time.

Solution Obtaining a valid solution is the ultimate goal of running a numerical solver. In *GAS*Pex, the solution is stored in its own file in binary format similar to the grid.

For time-dependent flows, multiple solution files may be stored (depending on the user setup), each representing the solution at a specific time.

The solution contains the unknowns of a problem. The number of unknowns will depend on the physics being modeled by the solver. For example, if the Navier-Stokes equations are solved for a laminar flow problem, the unknowns would include the species densities, u velocity, v velocity, w velocity, and pressure. The unknowns in *GASPer* are also referred to as the primitive variables.

Zones A grid is often created in blocks or zones. With this being said, zones are the building blocks of a grid. Zones are normally specified at the time the grid is generated.

Zonal Boundary When a grid is made up of more than one zone, and the zones are connected in a point-to-point fashion, a zonal boundary will exist. The zonal boundary is the surface that is common to two zones. Zonal boundaries are normally internal to the grid domain. Exceptions to this may be in the case of a periodic boundary.

Partitions Within *GASPer* zones may be broken up into smaller units called partitions. Partitions are normally used to help lower memory requirements and to help load balance a problem when multiple processors are being used. Partitions are used internally in *GASPer* and are mostly transparent to the user.

Physical Model In a boundary value problem such as CFD, a real-world physical problem must be represented and solved for using mathematical equations and models. A physical model in *GASPer* is used to describe the physics of a problem in computational (mathematical) terms.

Run Definition The run definition is how the user specifies the problem will be executed. This may include parameters such as time step, temporal accuracy, multiple-processor settings, and time integration.

Database A database is a collection of data that may need to be accessed by a third party. In *GASPer* the solver will need to access a database that describes the properties of such things as a gas, liquid, or solid. Like the problem setup file, the database is written to an XML file. It also has its own GUI for making changes to it. The user may have any number of database files on hand, but for each problem setup, the user must specify a database file to use.

Post-Processing Post-processing is the term used to describe what the user does with the solution after it is obtained by the solver. In *GASPer*, the user may post-process the solution by either viewing it in the GUI (in a plot domain or in the computational domain), or write the solution to a file for use with an external software.

3.8 Fluid Flow Problems

*GAS*Pex is traditionally used to solve fluid flow problems. This includes flows with any number of gas species with flow velocities ranging from subsonic to hypersonic. For these types of problems, a brief outline follows showing the steps that the user will take in order to set up and run the solver. Note that this is intended to be a general outline with the purpose of getting the user familiar with the thought process of setting up and running a *GAS*Pex problem. The user should go through the tutorials for more details of setting up a *GAS*Pex input deck.

Grid Import The user will run the *GAS*Pex GUI and import a grid file. Once the grid is imported, the user may be required to describe zonal boundaries (which can be done automatically via the GUI) and group boundary surfaces into groups (folders) for assigning boundary conditions. The CGNS format may include this information.

Sequence Grid If the user would like to use mesh sequencing, then additional grids (called sequences) will be created.

Setup Physical Model The physical model is set up next. This involves assigning boundary conditions, flow conditions, convective (inviscid) and diffusive (viscous) flux settings, thermodynamic properties, and the chemistry model.

Setup Run Definition In the run definition, the user decides which grid sequence to solve, the number of cycles to perform along with convergence criteria, the time integration scheme, and processor settings.

Save GUI and Exit Once the problem is set up using the GUI, the user must save the problem file and exit.

Run the Flow Solver The user is now ready to execute the flow solver from the command line. See the binary commands for *gaspex* described earlier in this chapter for details.

Post-Process, or Visualize the Solution With a solution in hand, the *GAS*Pex GUI can be used to Post-Process and Visualize the current solution.

3.8.1 Unstructured Grids

Version 5 has the ability to solve fluid flow problems on structured grids, unstructured grids or a combination of two. The unstructured solver supports meshes with the following cell types: tetrahedra, hexahedra, prisms, and pyramids. The mesh can contain one or more of these cell types (hybrid grid).

There are some differences in setting up and running on an unstructured mesh compared to a structured one. These are listed below, along with some key points concerning the unstructured flow solver.

Supported Grid Formats There are currently three supported grid formats for unstructured meshes. These are CGNS, FV-UNS (Fieldview) and VGrid. A CGNS grid file can be either single or multi-zone, since this format contains the zonal boundary information which is used to setup the zonal boundaries. The FV-UNS grid file supports single zone meshes.

Mesh Sequencing In order to perform mesh sequencing, the imported grid must undergo an agglomeration of cells. In doing this, cells are combined in order to reduce the total number of cells in a grid. When cells are combined to create a sequenced mesh, the discretization of node points on surfaces does not change. This allows the original number of faces on boundaries to be kept the same in order to maintain geometric integrity. Solution display has limitations for agglomerated grids.

Turbulence Models All turbulence models are supported for unstructured except the zero equation Baldwin Lomax model. This is an algebraic model which is only supported for structured grids at this time. For turbulence models that require a wall distance, the “Directed Distance” option is not available since it computes the wall distance by following grid lines emanating from the wall.

Overlapping Grids/Chimera Overlapping grids are currently not supported for unstructured grids.

Only The Navier-Stokes Physical Model is supported Problems involving physical models other than the Navier Stokes equations are currently not supported for unstructured grids.

Implicit/Explicit Solver When running a case on multiple processors, the user will perform load balancing in order to divide a zone into partitions. These partitions are then distributed among the processors such that each processor solves the governing equations on its respective partition(s). In order to reduce the required memory for the solver, the user may select to run the partitions explicitly such that only one partition is in memory at a time.

Mixed Grids The user can run the flow solver on a problem with both structured and unstructured zones. This allows structured grids to take advantage of its logical ordering.

3.9 Solid Material Problems

When performing a heat conduction problem on a solid material, the steps to take are almost identical to that of a fluid flow problem. The only difference will be in the physical model. The user must select a solid material physical model for both the zone initialization and the run group. Apart from this, the steps to running a heat conduction problem are very similar.

3.10 Conjugate Heat Transfer Problems

A conjugate heat transfer (CHT) problem is one that solves for the heat transfer across a boundary that has fluid flow on one side and a solid material on the other. This is done by coupling the fluid and solid solvers in *GASPer* such that energy is conserved when going from the solid to the fluid. In performing a CHT problem, the user will need to setup a physical model for the fluid and a second physical model for the solid material. The zonal boundaries common to the fluid and solid zones will need to have a special boundary condition assigned to them (Fluid/Solid ZB). Inside the run definition, the solid zones will have the solid material physical model assigned to it, while the fluid zones will have the fluid flow physical model assignment (using a run group for each). At this time, the zonal boundary between the solid and fluid zones must have point-to-point connectivity. In summary, the following steps are needed to perform a CHT simulation.

- Set up a Navier-Stokes physical model for the fluid zones.
- Set up a Solid Thermodynamics physical model for the solid zones.
- Create zonal boundaries for the fluid/solid surfaces. This can be done manually or automatically within the GUI using the **Compute Pt2PtZB** feature.
- Create a new folder inside the “Pt2Pt ZB” folder (which is inside of “Zonal Boundaries”). Move all the fluid/solid zonal boundaries into the new folder and toggle the folder to be a BC using the right mouse button edit menu.
- Set the boundary condition for the new folder created in the previous step to “Fluid/Solid Interface” in both physical models.
- Inside the run definition, create two run groups. The first run group should have the fluid zones selected with the Navier-Stokes physical model. The second run group will have the solid zones selected with the solid thermodynamics physical model. Note that the zones will need to be organized into sub-folders under the “All Zones” folder.

3.11 Helpful Information on Select Topics

The following sections are intended to provide guidance on select features of *GASPer* or on certain types of problems.

3.11.1 Solution Limiting Options

- Time step limiting (see Sec. 15.3.5 on pg. 252) can be very useful in maintaining numerical stability. During the initial stages of running a problem, the solution at a cell may change dramatically. This limiter will reduce the time step (either CFL or dt

based) at a cell based on how much the solution is changing. This is a local scheme in that each cell is evaluated and adjusted based on that cell's solution.

- Catastrophic limiting allows the user to apply global limits on pressure and temperature (see Sec. 15.4.2 on pg. 255). This is helpful during transient solutions or when large pressure gradients are present in the flow. This limiter can also help control stability when the grid contains areas of poor cell quality (which can be common in unstructured meshes).
- A turbulence limiter is available which controls the ratio of the eddy viscosity to laminar viscosity (see, for example, Sec. 11.4.7 on pg. 189). For boundary layers, a limiting ratio of 2000 is normally sufficient. For free shear flows, the “User Limit Ratio” should be used and adjusted as needed (free shear flows can have viscosity ratios in the 10,000+ range). When using the turbulence limiter, it is best to monitor the eddy viscosity ratio (an output variable) to ensure that the converged solution is not being limited (unless it is needed for numerical stability).

3.11.2 Plot3D Solution Import

External solutions can be imported into *GASPerx* for any zone or sequence combination. A common solution import format is Plot3D. When using this format for import, be aware of the following:

- File formats can be either ASCII or binary.
- The number of variables and their order is important. *GASPerx* expects the primitive variables to be in the same order as shown in the output variables under the filter “Primitive”. In fact, the best way to set up an export from *GASPerx* is to go to the primitive filter list and add them in that order.
- The location of data is important. Options here include nodes, interior cell centers, and cell centers with boundary values. Cell centers is normally recommended since this is where *GASPerx* solves for and stores the data. Using nodes will require some interpolation and loss of accuracy in the data. If data is exported from *GASPerx*, be sure to set the grid location under the grid range property.

3.11.3 Steady State Global Problems

- A locally based CFL number is more aggressive and tends to converge the flow field quicker, but it can also be more unstable.
- A CFL based on freestream conditions (Q Infinity) is useful to dampen out oscillations due to shocks that a local CFL may cause.

- Sometimes a combination of CFL's work best. Begin with a local CFL, and end the run with a infinity based CFL.
- Use the residual to monitor the stability of the run. For problems that involve complex physics such as shocks, expansions, combustion, separated flow, and higher order spatial accuracy with limiters, the residual can reflect more than just the solution convergence, but numerics as well. This can lead to residual ‘chatter’ or hangs. Changing the inviscid flux limiter can help in some cases.
- To determine convergence, monitor the solution directly. Use the output during a run feature to look at forces, moments, or other solution properties to determine when a solution is converged.
- In most cases, there is no advantage to running second order spatial accuracy versus third order for the inviscid flux.
- Min-Mod, Van Albeda, and Modified ENO are most commonly used spatial limiters. However, Min-Mod and ENO limiters are not continuous limiters, and their use can result in limit-cycles, and failure to converge.
- If a run keeps going unstable, trying lowering the CFL or going to first-order for the inviscid flux.
- It is always recommended to use grid sequencing. A solution on the coarse grid helps confirm that the problem is setup correctly and makes running on the fine grid easier since the physics are mostly developed by that point.

3.11.4 Time Accurate Methods

- Running with a constant time step in steady state mode is at best first order accurate (pseudo first order). For time accurate results, the user should run with either implicit dual-time stepping or explicit Runge-Kutta (see Sec. 15.2.4 on pg. 244).
- With dual-time stepping, a Newton sub-iteration approach may be taken. This is done by selecting the pseudo time step (in the time integration) to a large value like 1×10^{15} . The physical time step then controls the numerical stability. Set the number of cycles per time step to 10 initially. Monitor the solution to ensure that 10 cycles is sufficient for each physical time step. If needed, adjust cycle number up or down to optimize run.
- Another approach with dual-time stepping is the user selected time step. Select a CFL value that was used for the steady state run (or one that gives good convergence). Next set the physical time step as desired (there is no stability limit on the physical time step when using a stable CFL value). Monitor the solution to determine how many cycles are required for converging each physical time step. This approach may require 100 or more cycles per physical time step.

3.11.5 Time Accurate Simulations

- Once a time accurate simulation is started, the user may return to the steady state ($t = 0$) setting. This may be done from the GUI by pressing the **Reset(t=0)** button located in the **Time Levels** display list.
- If the user wants to change the temporal accuracy during a time accurate simulation, it is easily done if going from a high temporal accuracy to a lower accuracy. If going the other way, the user will need to create additional solution file levels. There may also be a local irregularity in the solution at or near the point in time where the time step was changed.
- If the solution is being saved (archived) at a specified time interval, remember that the storage requirements will grow, so make sure that adequate storage (disk memory) is available.

3.11.6 Turbulence Simulations

- Begin problems with the turbulence model on.
- Problems may need the turbulence limiter during initial convergence where the solution is setting up and becoming more physical.
- Monitor the eddy viscosity/laminar viscosity ratio to ensure the turbulence modeling is active.
- It is best to turn the limiter off to complete a run (but some cases require the limiter to be left on).
- Run both thin-layer and cross derivatives if unsure of when they apply. Thin-layer is always required. Cross derivatives improve viscous flux accuracy when the flow is more complex (such as flow separation).
- Use compressibility corrections when the convective Mach number is supersonic and the shear flow or mixing is present.

3.11.7 Detached Eddy Simulation

- Available for the Spalart-Allmaras and Menter's SST turbulence models.
- Grid must be 3-D. There is no grid convergence with DES. The finer the grid, the more smaller eddies will be resolved and simulated.
- Solver should use a constant time step for steady state runs or a time accurate algorithm like Dual-Time stepping. Running with steady state but using a constant Dt is a good way to start.

- Pick a physical time step such that the local CFL is near 1.
- Grid cells should be uniform (equal lengths in each direction) in the DES region. A length scale is associated with each cell and is taken as the longest side, so you want the length scale equal in all three directions in order for the grid to be efficient (otherwise the longest length scale dominates).
- In certain cases, the solution may NOT develop into DES on its own. In these cases the user must perturb the velocity field. This can be done using the **Initialize Soln** button inside the solution management section (see Sec. 8.5.7 on pg. 98). Selecting this option will bring up a dialog allowing the user to “Perturb Velocity for DES”.

3.11.8 Chemistry Simulations

- Use mesh sequencing. Fine mesh solutions will converge faster if interpolated from coarser sequences.
- Use finite-rate chemistry for most accurate results. Equilibrium chemistry is just as stiff or stiffer than finite-rate chemistry.
- For most problems, it is best to start with chemistry turned on (not off).
- When creating a chemistry model, double check the units on the reaction rate coeffs.
- In general, if problems persist, simplify the physics until the cause of the problem can be identified (*e.g.*, frozen flow, inviscid). Also check the computed temperatures since they may be out of range if using curve fits.
- For very stiff problems, the Operator Split/ODE option in the Navier-Stokes “Solver Info” section can be used.

3.11.9 Space Marching

- For two-dimensional, space-marching problems, always use the LU algorithm.
- For three-dimensional, space-marching problems, you may use any of the other time-integration schemes (but not LU).
- Use the full flux (inviscid setting) in the marching direction.
- Limiters in the marching direction are “catastrophic”.
- The flow should be supersonic in the direction normal to all cross-flow planes.
- First-order and second-order upwind are the only two accuracy options available for the inviscid flux in the marching direction.

3.11.10 Overlapping Grid Simulations

- When generating the grid, try to make overlapping cells similar in size. This yields higher quality stencils for mesh interpolation.
- A collar grid may be necessary around intersecting surfaces. Try to make the collar grid finer or at least as fine as the two body grids in that region. Ideally the collar grid should not peel (the user can set a flag that prevents peeling in a zone).
- Always view the hole cutting in the GUI to verify it was done correctly.
- View the orphan cells to identify problem areas with the grid.
- Remember that the solver will cut holes and compute the interpolation coefficients if not done in the GUI.
- Hole cutting is decomp independent, but the interpolation coefficients are not. Therefore, compute the coefficients after performing zone decomposition.

3.11.11 Grid Generation Issues

- For turbulent flows, strive for a $y+$ around 1 at no-slip walls. A good initial estimate for the first grid cell spacing is: $\Delta n = 0.7L/\sqrt{R_e L}$.
- Try to achieve at least 30-40 cells inside the boundary layer.
- Use hyperbolic tangent grid spacing in the boundary layer.
- Try to limit viscous cell aspect ratios to 1000.
- Structured grids should be orthogonal as possible with smooth volumes transitions (no abrupt changes in grid spacing).
- Structured grid lines coming off a symmetry plane should be as normal as possible.
- Try to avoid using singular lines or points.

3.11.12 Axi-Symmetric Problems

- When running an axi-symmetric problem, there are several different choices for the side wall boundary conditions. If the grid slice is 4.5 degrees and the grid singular axis is in the x-direction, use the positive and negative axi-symmetric wall BCs. This is the default setting when importing a 2-D grid as axi-symmetric into *GASPer*.
- The tangency or symmetry boundary condition may also be used for the side walls. If using the Van Leer flux option, use only solid wall BCs like tangency. The Van Leer formulation does not preserve the normal no-flow condition for split flux BCs.

Chapter 4

GASPer Menu Bar

4.1 The Pull-Down Menus

At the very top of the *GASPer* graphical user interface (GUI) is a pull-down menu bar (see Fig. 4.1 on pg. 53). The menu bar is used to perform such operations as file open and save, graphical zoom, and setting preferences. The operations performed from the menu bar are global operations in the sense that they can be done at any time while using the GUI.

There are seven pull-down menus available in *GASPer*: **File**, **Commands**, **Wizards**, **View**, **Options**, **Tabs**, and **Help**. To view any of the pull-down menus, click on its name using any of the mouse buttons.

4.2 The File Pull-Down Menu

The **File** pull-down menu is used for file input/output operations. For example, this menu selection is where you would go to open and save input decks for *GASPer*. It is also used to import grid files. When you go to quit the GUI, this is the menu item that you would need to go to. The following subsections discuss each option in the **File** pull-down menu.

4.2.1 The File/New Menu Item

The first item under the File menu is **New** (**File**→**New**). This menu item is used to start a new input deck. All the input values in the deck are set to their default values and any existing data is lost.

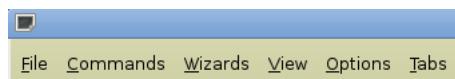


Figure 4.1: The pull-down menu bar in *GASPer*.

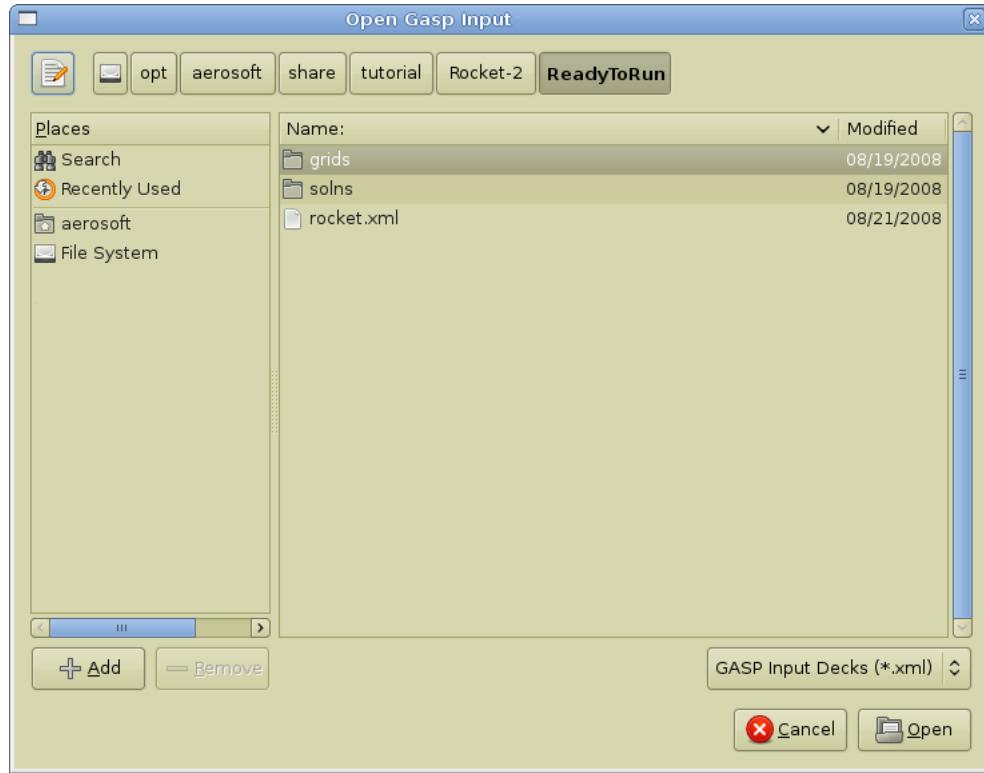


Figure 4.2: The *GASPE* file open dialog window.

4.2.2 The File/Open Menu Item

The **File→Open...** menu item allows the user to open a previously created input deck. After selecting the **Open** menu command, a File Open dialog window appears. This dialog window is shown in Fig. 4.2 on pg. 54.

At first, the dialog window displays all the .gsp and .xml files and folders listed in the current working directory. In order to display all the files in the directory, adjust the filter setting located below the name display. To open an input file, select the file name and press the **Open** button. The file name could also be double clicked to perform the same function. Use the **Cancel** button to exit the dialog window without opening an input file.

Note:

If an invalid input file is chosen, a message window will appear explaining the source of the error.

4.2.3 The File/Save Menu Item

The **File→Save** menu item is used to save the .gsp file and associated auxiliary files. If the input deck is new (never been saved before), then a **Save GASP Input** window (as shown in Fig. 4.3 on pg. 55) will appear and the user will be prompted for a name. Otherwise, the

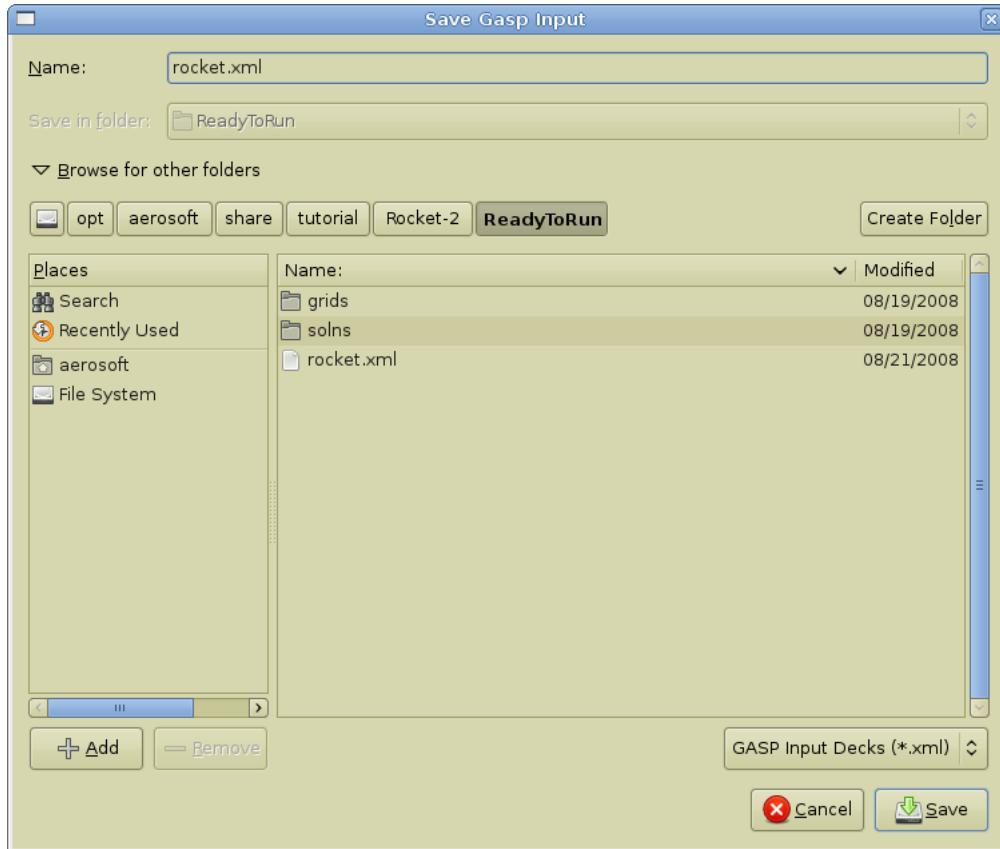


Figure 4.3: The **Save GASP Input** window.

deck will be saved to the file name already in use.

4.2.4 The File/Save As Menu Item

The **File→Save As...** menu item is used to save the .gsp file and auxiliary files with a problem name specified by the user. The **Save GASP Input** window (as shown in Fig. 4.3 on pg. 55) will always appear with this menu item.

4.2.5 The File/Close Menu Item

The **File→Close** menu item allows the user to quit an open input deck without exiting the GUI.

4.2.6 The File/Generate HostId

The **File→Generate HostId...** menu item allows the user to obtain the HostId for the current platform. The HostId is used by the RLM license manager to uniquely identify the computer

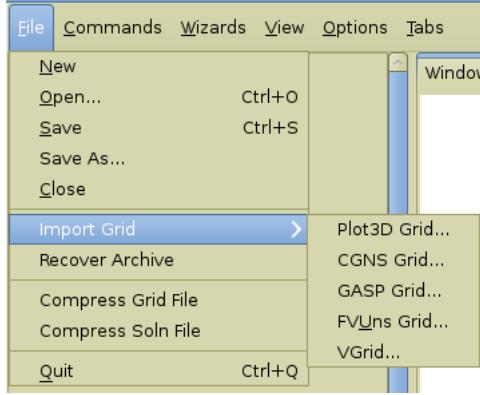


Figure 4.4: The **File**→**Import** menu options.

where the license server will run. You will need to create the HostId only on the machine where you wish to run the license manager. If you wish to obtain a node-locked license, you need to generate the hostId on that platform. Selecting the **File**→**Generate HostId** menu will display a set of dialogs where the user will provide necessary information that AeroSoft uses to generate a license key.

4.2.7 The File/Import Menu Item

The menu item allows the user to import PLOT3D, CGNS, GASP, Fieldview, and VGrid grid files. Unlike the other menu items in the **File** menu, the **Import** menu option has sub-items associated with it. This is shown in Fig. 4.4 on pg. 56. Selecting one of the sub-menus will perform different operations as described in the following sections.

The Import/Plot3D Grid Menu Item

The PLOT3D file format is one of the most common formats for structured grids. Selecting **PLOT3D Grid...** will bring up the PLOT3D grid import window as shown in Fig. 4.5 on pg. 58. There are various options to PLOT3D grid files. Therefore, the user must select various attributes such as file format, grid format, grid system, and grid units. The different options to the import window are discussed below.

1. **File Format** The PLOT3D file format has two choices: ASCII and Binary. Select the format which corresponds to the file to import.
2. **Grid Format** A PLOT3D grid can have various grid formats. *GASPer* needs to know what grid format to expect in order to read in the data correctly. The grid formats that *GASPer* supports at this time are: multi-zone, I-blanking, and double precision. Most PLOT3D grid files are generated in multi-zone format, so this option is selected by default. The I-blanking format is used to specify hole cell information that can be included in the grid file. The double precision option indicates the amount of data

used to represent each data point. Note that a grid file can have any combination of the above grid format options.

Note:

When the hole cells option is specified, the GUI will correctly read a file which contains blanking information, but it will ignore the blanking data. *GASPer* will compute blanking information itself, when required.

3. **Grid System:** The grid itself may represent either a two- or three-dimensional domain. For two-dimensional grids, *GASPer* must convert the grid into a three-dimensional grid since *GASPer* operates on 3-D control volumes. To do this, *GASPer* will take the reference length based on the grid domain and create the third dimension using five percent of the reference length value. *GASPer* assumes a 2-D grid is given in the x-y plane, so the third dimension that is created by *GASPer* will be in the z direction.

If the user has a 2-D grid and wishes to run an axi-symmetric problem with it, then they should select the **2-D axi-symmetric** option. *GASPer* will then take the 2-D grid and rotate it about the x-axis $\pi/80$ and $-\pi/80$ ($+2.25$ deg and -2.25 deg). *GASPer* assumes the 2-D grid is in the x-y plane and that the x-axis is the axis of symmetry.

For a three-dimensional grid, *GASPer* will import the grid as given in the grid file. No changes are made to the grid, but *GASPer* will check to be sure that the grid is right handed. In order for the cell volumes to be positive, the grid must be right handed, therefore *GASPer* will check for this and refuse to import the grid if a left handed grid is discovered.

4. **Grid Units:** The units of the grid file must be specified before *GASPer* imports the grid. This is done using the **Grid Units** option menu. From this menu, select the unit system that represents the grid file to import. Internally, *GASPer* stores the grid in the metric unit system. Therefore *GASPer* needs to know what unit system the grid data is in so that it can convert the grid to meters.

The default unit lengths for *GASPer* are SI meters and English feet. Users can create their own unit system (see Sec. 3.6 on pg. 40) or import the grid in meters or feet and then scale the grid inside of *GASPer* using the transform grids options (see Sec. 8.4 on pg. 93).

Once the file format, the grid format, the grid system, and grid unit system are set correctly, the file to be imported can be selected. To select the PLOT3D file, use the **Browse...** button. This will bring up a file import window which will allow the user to browse and select a grid file. The **Open** button should be pressed inside the file selection window in order to return to the PLOT3D grid import window.

Once a grid file has been selected using the **Browse...** button, basic grid information is displayed in a table located in the PLOT3D grid import window. At this point the user has two options. Either the grid file can be imported into the GUI using the **Import** button, or

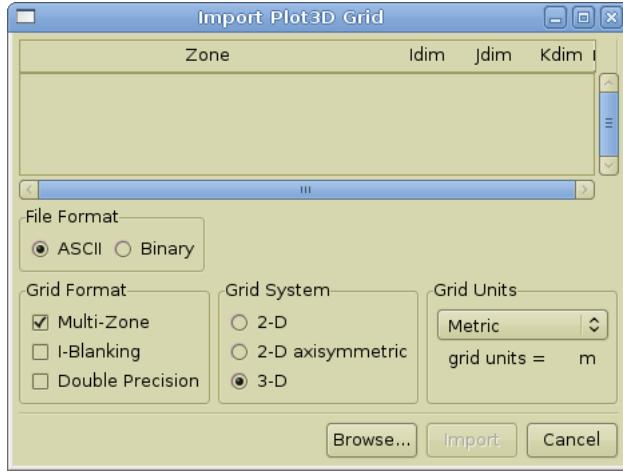


Figure 4.5: The **File→Import→Plot3D Grid...** import window.

the grid import process can be canceled using the **Cancel** button. Once the user reaches this point, all options in the PLOT3D grid import window are disabled except for the **Import** and **Cancel** buttons.

Note:

The file format, the grid format, the grid system, and grid unit system must be set correctly before selecting a grid file.

The Import/CGNS Grid Menu Item

Selecting the **File→Import→CGNS Grid...** menu item will open the CGNS grid import window as shown in Fig. 4.6 on pg. 59. There are various options to CGNS grid files. The user must select appropriate attributes such as whether or not to import zonal boundary descriptions, boundary condition family's and the unit system for the grids. The different options to the import window are discussed below.

1. **Import Options** A CGNS grid file may contain additional information beyond grid dimensions and grid coordinates. The import options may be individually selected to identify which additional information will be included at import. Selecting the **Zonal Boundaries** toggle button will include connectivity information. If boundary surfaces have been assigned to boundary condition families within the CGNS file, then selecting the **BC Families** toggle button will include that information.
2. **Name Options** If **Use CGNS Names** is selected, the surface nodes and zone nodes will be named according to the tags in the CGNS file. If **Use BC Names** is selected, the Surface Families will be assigned the CGNS BC Name, not a name corresponding to the type. Note that Pointwise correctly sets the name of the BC, while GridPro does not. When importing a cgns formatted grid created in GridPro, make sure that **Use BC Names** is not selected.



Figure 4.6: The **File**→**Import**→**CGNS Grid...** import window.

3. **Grid Units** The units of the grid file should be specified before *GASPE* imports the grid. This is done using the grid units option menu. From this menu, select the unit system that represents the grid file to import. Internally, *GASPE* stores the grid in the metric unit system. Therefore *GASPE* needs to know what unit system the grid data is in so that it can convert the grid to meters.

Once the import options, name options, and grid unit system are set correctly, the file to be imported can be selected. To select the CGNS file, press the **Browse...** button. This will bring up a file import window which will allow the user to browse and select a grid file. The **Open** should be pressed inside the file selection window in order to return to the CGNS grid import window.

Once a grid file has been selected using the **Browse...** button, basic grid information is displayed in a table located in the CGNS grid import window. At this point the user has two options. Either the grid file can be imported into the GUI using the **Import** button, or the grid import process can be canceled using the **Cancel** button. At this time, all options in the CGNS grid import window are disabled except for the **Import** and **Cancel** buttons.

Note: All grid import options must be set correctly before selecting a grid file.

The Import/GASP Grid Menu Item

If you have a *GASPE* Version 4.2 or later grid file and wish to import it into the GUI, then select the **File**→**Import**→**GASP Grid...** menu item. Selection of this item will bring up the grid import window as shown in Fig. 4.7 on pg. 60. Press the **Browse...** button to bring up an additional window used to select the grid file. After a grid has been selected, the user can adjust the **Import Sequence #** to decide which sequence level to import (only one sequence can be imported at a time). The **Import** button is used to import the grid into the GUI. The **Cancel** will close the window without importing the grid.

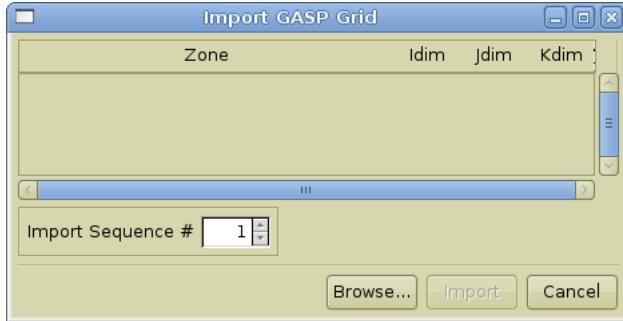


Figure 4.7: The **File**→**Import**→**GASP Grid...** import window.

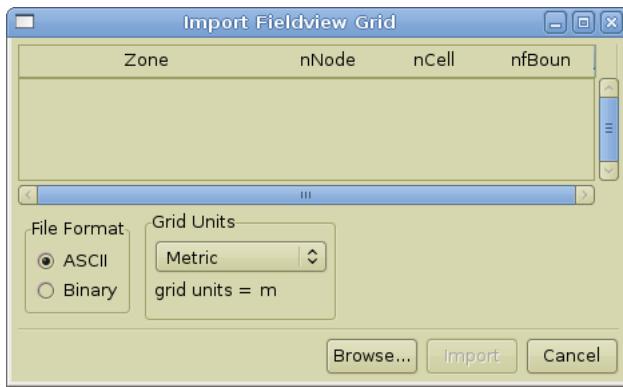


Figure 4.8: The **File**→**Import**→**FVUns Grid...** import window.

Note: If you wish to import a *GASPE* v4.0 or *GASPE* v4.1 formatted grid file, then you must first convert the entire *GASPE* input deck to v4.2, and then import the V4.2 grid file.

The Import/FVUns Grid Menu Item

If you have a Fieldview unstructured (ASCII or binary) grid file, and wish to import it into the GUI, then select the **File**→**Import**→**FVUns Grid...** menu item. Selection of this item will bring up the grid import window as shown in Fig. 4.8 on pg. 60. Select either **ASCII** or **Binary** from the **File Format** options. The units of the grid file must be specified before *GASPE* imports the grid. This is done using the **Grid Units** option menu. From this menu, select the unit system that represents the grid file to import.

Press the **Browse...** button to bring up an additional window used to select the grid file. The **Import** button is used after a grid file has been selected to import the grid into the GUI. The **Cancel** will close the window without importing the grid.

Note: Only select FV-UNS file versions are currently supported. For the ASCII file format, versions 2.4 and 3.0 are supported, while for binary versions 2.3 and 3.0 are supported.

The Import/VGrid Menu Item

GASPer supports the VGrid file format for unstructured grids. VGrid files are used to describe tetrahedra cells and can be in either ASCII or binary format. A complete VGrid mesh will have multiple files used to fully describe the grid. For binary grid files, the primary data is in a `.cogsg` file. For ASCII, the node data is in a `.grd` file along with a `.inp` file for the cell information. With both ASCII and binary grids, two additional files should also be present. These are the bc file (`.bc`) and the bc mapping file (`.mapbc`).

The units of the grid file must be specified using the **Grid Units** before *GASPer* imports the grid. From this menu, select the unit system that represents the grid file to import. Internally, *GASPer* stores the grid in the metric unit system. Therefore *GASPer* needs to know what unit system the grid data is in so that it can convert the grid to meters.

The default unit lengths for *GASPer* are SI meters and English feet. Users can create their own unit system (see Sec. 3.6 on pg. 40) or import the grid in meters or feet and then scale the grid inside of *GASPer* using the transform grids options (see Sec. 8.4 on pg. 93).

Press the **Browse...** button to bring up an additional window used to select the grid file. For binary file formats, select the `.cogsg` file. If the file format is ASCII, select either the `.grd` file or the `.inp` file. *GASPer* assumes the file prefix is the same for all the VGrid files and will automatically read in the other files.

The **Import** button is used after a grid file has been selected to import the grid into the GUI. The **Cancel** will close the window without importing the grid.

Note:

VGrid binary files are written under FORTRAN and may therefore be susceptible to big endian/little endian incompatibility. To prevent import issues, try to use *GASPer* on a similar architecture as the program which generated the VGrid files.

4.2.8 The File/Recover Archive Menu Item

For steady state problems, the user has the option of archiving or backing up solution files. Archiving is specified in the run definition (see Sec. 15.2.3 on pg. 243).

If the user has archived solutions, they can be retrieved using the **Recover Archive** option. Selecting this option will display an archive dialog which will allow the user to select an archive to make current. An archive normally consists of the following files:

- The solution file (`.sln`).
- The residual file (`resid.`).
- The input deck auxiliary file (`.aux.gsp`).
- Output files in use.

4.2.9 The File/Compress Grid File

GASPer will save imported grids in its own binary file format. Grid files are given the .grd file extension with the same file name as the input deck. The grid data is written to the binary grid file in chunks of data. As a new chunk of data is written, the data is appended to the existing file. For example, if the user imports a grid file and saves, the complete grid is written to file. If the user then creates a sequenced grid and saves, then the new sequenced data is appended to the existing grid file and the file size will grow. The user may then decide to delete the fine grid (the imported grid) and save. In this case the grid file size will not change since *GASPer* does not automatically “move” the sequenced grid location within the stored memory space.

In order to compress the grid file and remove any unused chunks of file space, the user should select the **Compress Grid File** option. This will immediately force a rewrite of the grid file. Because this action will alter the grid file pointers, this operation must not be performed while the solver is running.

4.2.10 The File/Compress Soln File

The **Compress Soln File** option performs the same operation as the grid file compression except that it is applied to the .sln file.

4.2.11 The File/Quit Menu Item

The **File→Quit** menu item allows the user to quit the *GASPer* GUI. If changes to the input deck are detected, a dialog box will be displayed prompting you to save your changes.

4.3 The Commands Pull-Down Menu

The **Commands** pull-down menu is used for some basic GUI operations with focus on the graphics. Note that most commands have a hot-key that is listed with the command. A summary of each command is given below.

Zoom Selected If a folder or node is selected in the tree view, this command will redraw the graphics window so that the object is 80% of the window. The selected object must be rendered to sparse, wire, or full in order for this command to work.

Next Selected Users can select objects graphically using the mouse. This is done using the Alt key along with the left mouse button. If the entry is common to more than one surface or zone, then the user can use this command to “jump” to the next entry in the tree view. Note that an object is selected in the tree view based on its visibility within the tree. So if a folder is closed, objects within the folder cannot be selected graphically. Instead, selecting an object within the folder would result in selecting the folder itself.

Select All Child Nodes The tree view is located in the upper left corner of the GUI. It contains a number of folders with additional folders or entries like surfaces and zones. Every entry in tree view is referred to as node, whether it is a folder (group node), a zone (zone node), surface (surface node), or other type. When a node is inside a folder, it is referred to as the child of the folder with the folder being the parent. When a folder is selected (highlighted), selecting this option will un-select the folder and select all the nodes directly inside the folder.

Render Selected None This option changes the rendering mode of the selected node or entry in the tree view. Render none will prevent any part of the node from being displayed in the graphics window.

Render Selected Sparse This option changes the rendering mode of the selected node or entry in the tree view. Render sparse will draw the outline of the selected node in the graphics window.

Render Selected Wire This option changes the rendering mode of the selected node or entry in the tree view. Render wire will draw the grid lines of the selected node in the graphics window.

Render Selected Full This option changes the rendering mode of the selected node or entry in the tree view. Render full will draw the entire node using color contouring in the graphics window.

Note:

The level of rendering will be limited to the parent folder. So if the parent is set to render sparse and the child node is set to render full, rendering will be sparse in the graphics window.

Xscreen --> +X→Yscreen --> +Y Align the positive model x axis with the screen x axis and the positive model y axis with the screen y axis.

Xscreen --> +X→Yscreen --> -Y Align the positive model x axis with the screen x axis and the negative model y axis with the screen y axis.

Xscreen --> +X→Yscreen --> +Z Align the positive model x axis with the screen x axis and the positive model z axis with the screen y axis.

Xscreen --> +X→Yscreen --> -Z Align the positive model x axis with the screen x axis and the negative model z axis with the screen y axis.

Xscreen --> -X→Yscreen --> +Y Align the negative model x axis with the screen x axis and the positive model y axis with the screen y axis.

Xscreen --> -X→Yscreen --> -Y Align the negative model x axis with the screen x axis and the negative model y axis with the screen y axis.

Xscreen --> -X→Yscreen --> +Z Align the negative model x axis with the screen x axis and the positive model z axis with the screen y axis.

Xscreen --> -X→Yscreen --> -Z Align the negative model x axis with the screen x axis and the negative model z axis with the screen y axis.

Xscreen --> +Y→Yscreen --> +X Align the positive model y axis with the screen x axis and the positive model x axis with the screen y axis.

Xscreen --> +Y→Yscreen --> -X Align the positive model y axis with the screen x axis and the negative model x axis with the screen y axis.

Xscreen --> +Y→Yscreen --> +Z Align the positive model y axis with the screen x axis and the positive model z axis with the screen y axis.

Xscreen --> +Y→Yscreen --> -Z Align the positive model y axis with the screen x axis and the negative model z axis with the screen y axis.

Xscreen --> -Y→Yscreen --> +X Align the negative model y axis with the screen x axis and the positive model x axis with the screen y axis.

Xscreen --> -Y→Yscreen --> -X Align the negative model y axis with the screen x axis and the negative model x axis with the screen y axis.

Xscreen --> -Y→Yscreen --> +Z Align the negative model y axis with the screen x axis and the positive model z axis with the screen y axis.

Xscreen --> -Y→Yscreen --> -Z Align the negative model y axis with the screen x axis and the negative model z axis with the screen y axis.

Xscreen --> +Z→Yscreen --> +X Align the positive model z axis with the screen x axis and the positive model x axis with the screen y axis.

Xscreen --> +Z→Yscreen --> -X Align the positive model z axis with the screen x axis and the negative model x axis with the screen y axis.

Xscreen --> +Z→Yscreen --> +Y Align the positive model z axis with the screen x axis and the positive model y axis with the screen y axis.

Xscreen --> +Z→Yscreen --> -Y Align the positive model z axis with the screen x axis and the negative model y axis with the screen y axis.

Xscreen --> -Z→Yscreen --> +X Align the negative model z axis with the screen x axis and the positive model x axis with the screen y axis.

Xscreen --> -Z→Yscreen --> -X Align the negative model z axis with the screen x axis and the negative model x axis with the screen y axis.

Xscreen --> -Z→Yscreen --> +Y Align the negative model z axis with the screen x axis and the positive model y axis with the screen y axis.

Xscreen --> -Z→Yscreen --> -Y Align the negative model z axis with the screen x axis and the negative model y axis with the screen y axis.

4.4 The Wizards Pull-Down Menu

The **Wizards** pull-down menu contains operations that assist the user in either the graphical display or basic grid generation. Each wizard presents the user with a step by step display to help accomplish an operation. A summary of each wizard is given below.

Contours Window This wizard assists the user in setting up a Contours solution visualization node. This same operation can be done manually by selecting on the Solution Visualization folder in the tree view and using the right mouse button edit menu.

Cartesian Zone This wizard assists the user in creating a single Cartesian zone. At this time, this is the only type of three-dimensional grid that *GASPer* can generate.

Wavefront Zone This wizard assists the user in creating a two-dimensional wavefront zone. Wavefront zones can be planar or spherical, and are used with add-on modules of *GASPer* (such as the wave-optics solver).

4.5 The View Pull-Down Menu

The **View** pull-down menu contains options related to the graphical display. A summary of each command is given below.

New Window When the GUI is first run, there is a single window for the graphics display. Users can create additional windows using this option.

Duplicate Window Use this option to create a new window that is a duplicate of the currently selected window.

Delete Window This option will remove or delete the currently selected window. Note that there must always be at least one window.

Toggle Window Location When the GUI is first started, the graphics window is integrated into the main GUI display. The user may detach the graphics window from the main display using this option. Selecting the option again will place the graphical window back into the main display.

Next Window Selecting this option will change the display to the next window. This option is only valid when there are multiple windows.

Toggle Window Tabs By default, tabs showing which window is being displayed are located directly above the graphical window. These tabs can be hidden using this option. Selecting this option again will cause the tabs to reappear.

Undo View Change *GASPE* automatically saves the last 10 views as the graphical display is used. The user can then cycle through these views using this command.

Save View Selecting this option will save the current view. The view is then placed at the bottom of the **View** pull-down menu.

Edit View List Once views are saved (see above item), the user may edit saved views using this option. Views are edited with a pop-up dialog which allows the user to rename or remove saved views.

4.6 The Options Pull-Down Menu

The options menu allows the user to select preferences for GUI behavior. A summary of each command is given below.

Tool Tips Select data entries and buttons have tool tips associated with them. Tool tips are on by default and are activated when the mouse pointer resides over the entry. Selecting this option will toggle the tool tips on or off.

Include All XML The input deck is written in XML format and contains all the parameters associated with the GUI. This includes the tree view layout, solution display, and all the run parameters needed for the solver. Every entry in the XML file has a default setting, which is applied with the GUI is run for the first time. When the input deck is saved to file, only those entries that are different from the default values are written. If the user would like all the parameters to be written to the XML file, then selecting this option will force all the parameters to be written when the GUI is saved.

Auto Solution File Update The GASP user interface can detect changes in the solution file and automatically re-read/update the solution display within the solution visualization window. With this feature, the user may dynamically monitor the changes in a solution as it converges without having to re-start the GASP user interface. Select this option to enable this feature. Note that older operating systems may not support this feature, in which case this option will not appear in the options list.

Update Solution File(s) The GASP user may force an update of the solution file for steady state solutions or refresh the **Time Levels** list for time accurate simulations. For example, this is beneficial if the user is running a time accurate simulation and would like to update the time level list to include recently saved time dependent solutions.

Global Variable List... GASP maintains a list of global variables which may be used to alter certain data quantities at run-time. Selecting the **Options→Global Variable List...** menu item will present the user with a dialog window containing a list of all active global variables defined within the current input deck, as shown in Fig. 4.9 on pg. 68. Using the new or cut buttons will add or remove variables to the list respectively.

Global variables can be static or varying. A static variable means the value will not change throughout the simulation. By selecting the **Constant Value** option, a single value input allows the user to set the global variable.

For variables that vary, the user may select from either **Time Varying** or **Cycle Varying** options. This allows the user to vary most any input parameter by time (for time accurate simulations) or cycle number (for steady state simulations). In either case, the user can manually enter the values by first setting the number of data points and then editing the entries in the table that appears. The user may also provide data by importing an ASCII file. If a file is used to create the data set, each line of the file should contain the time (or cycle depending on the input option) and the value. A new line of data should be used for each data pair. When importing data via a file, the number of data points is set based on the input file.

For the current example, two static global variables are defined, *UINF* and *PINF*, which are meant to allow alteration of freestream values of velocity and pressure respectively. The **Value** input represents the local GUI value when the global variable is substituted for real data in the input deck. The units on the global variable will be set based on the selected input deck unit system. For example, if the input deck is using the default metric units, then values for velocity will be *m/s* and values for pressure will be *N/m²*.

Global variables are used by substituting the global variable name in place of an input value. If a global variable substitution is not supported, an error message will be printed to the terminal, and the original value will be restored.

4.7 The Tabs Pull-Down Menu

Directly below the graphics window (lower right area in the GUI) is the data input. Data is grouped into sections using tabs such as File, Zones, Physical Models, etc. If a tab is not being used for the current problem, the user may hide the tab by selecting the corresponding name from the tabs pull-down menu. Note that some tabs may be hidden by default such as **Motion Models**.

4.8 The Help Pull-Down Menu

The help menu has the following options.



Figure 4.9: The Options→Global Variable List... dialog window.

User Manual Selecting this option will display the *GASPE* User's Manual using a PDF viewer. Recall that the user's manual installed with *GASPE* is located at:

`AEROSOFT_HOME/share/doc/gasp_usersguide.pdf`

where `AEROSOFT_HOME` is the install directory location. For linux platforms, the GUI will try to use one of the following commands to display the manual: acroread, evince, or okular. On a windows machine, the Adobe Acrobat Reader DC will try to be used.

If none of the default programs are available, the user may specify a PDF viewer by setting the environment variable `AEROSOFT_PDF_BIN`. If the binary is not in the machine path, then the environment variable should consist of the full path.

Restrictions This option displays any U.S. government restrictions that might be associated with the *GASPE* software version.

About GASP This option produces a pop-up window with the *GASPE* version number, the date the binary was built, and contact information for AeroSoft.

Chapter 5

The Main Frame

The **Main** frame of the *GASPerx* GUI will appear in the lower right hand corner when the GUI is started. The default **Main** frame will appear as in Fig. 5.1 on pg. 69. The settings for **Main** will be described in the following sections.

5.1 Chemistry and Thermodynamics Database

GASPerx accesses species and reaction data from a database of chemistry models. When a chemistry model is specified in a user's input, *GASPerx* reads the database file to retrieve quantities such as molecular weight, specific heat, and characteristic temperature of vibration. This chemistry database file is created using the database manager component of the *GASPerx* GUI. A database XML file is supplied with *GASPerx* and is located in `aerosoft/share/thermochemdb.tcd`. However, if you have created a new chemistry model

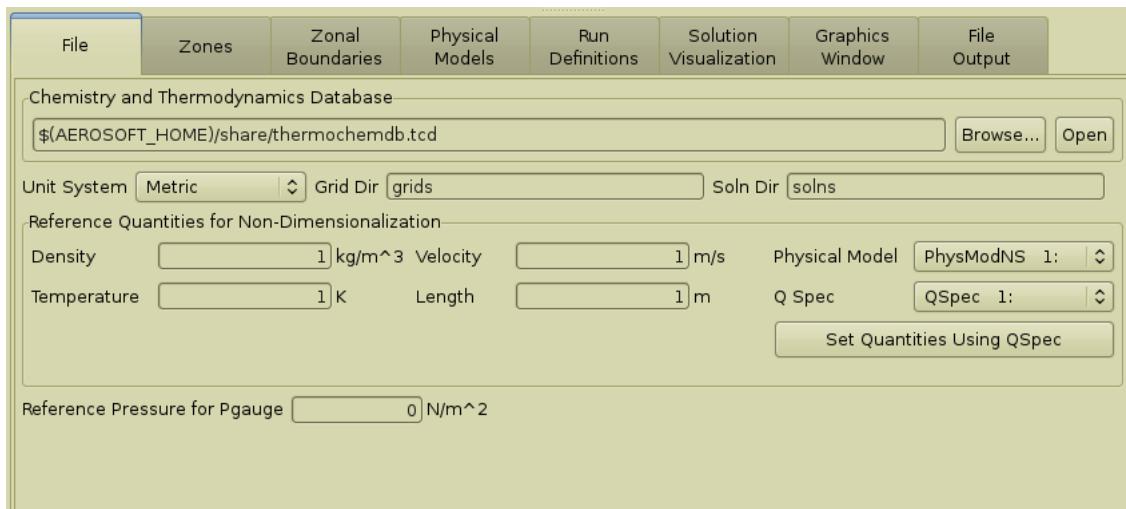


Figure 5.1: The Main interface frame of the *GASPerx* GUI.

using *DBASEMGR*, then your new database file can be specified in the **Chemistry and Thermodynamics Database** input. You may also browse the file system through a dialog to choose a database file by pressing the **Browse** button. A database file is necessary in order to use *GASPEX*.

The database file can be viewed using a stand-alone GUI or by pressing the **Open** button next to **browse**. This will allow the database to be opened in a new window for viewing or editing. In order to save any changes made to the database, the user must use the **File→Save** menu item located in the database GUI window. For more information on the database GUI, see Chap. 18 beginning on pg. 365.

Note:

Commencing with Version 5.1, the default chemistry and thermodynamics database file has been renamed from `db.xml` to `thermochemdb.tcd`. This change was made to facilitate interface interaction in the Microsoft Windows environment. In addition, the suffix for all thermo chemistry database files should be changed from “`.xml`” to “`.tcd`”. A version number has been added to the `xml` file to detect compatibility with earlier versions of the database file. The current *DBASEMGR* can read earlier version files and save to the new format. You will need to perform this operation before using Version 5 and earlier database files.

5.2 Unit System

The **Unit System** option menu specifies the units of your problem. All the quantities which have units in the input deck will be presented in terms of the selected units. Examples include the Q specification, reference quantities, and grid coordinates.

GASPEX comes with two unit systems built in. These are **Metric** and **English**. Upon running *GASPEX*, the unit system will default to the **SI Metric** system. Changing the **Unit System** option menu will cause *GASPEX* to convert all the dimensional quantities in the input deck to the newly selected unit system. Because of this, be sure to select the correct unit system for the input deck before proceeding with the set up.

Additional unit systems can be added to the option menu by editing the `units.txt` file. *GASPEX* will search for this file in the `$(AEROSOFT_HOME)/share` directory. Any unit systems found in the `units.txt` file will be added to the list of unit systems.

5.3 Grids Directory

To help organize a problem being simulated, *GASPEX* allows the grid files to be placed in a separate directory. By default, the name of this directory is `grids`. The user may change this name or set a different location for the grid files.

For steady state calculations, there will only be a single grid file. For moving, time-dependent problems, the user has the option of saving the grid and solution files on an incremental basis. It is in these situations that organization is important. There will be a separate grid file (or logical link) in the **Grid Dir** directory for each saved physical time step. Likewise, there will be a unique motion model XML file for each physical time-step.

5.4 Solution Directory

Similar to how the grid files have their own directory, the same is true for the solution files. For a steady-state simulation there will only be one solution file generated by *GASPer*. For time accurate moving body simulations, there is the potential of having multiple solution files depending on how the user sets up the problem. By default the solution directory is named **solns**.

Solutions may also be archived for future reference or for recovery purposes. Archived data is also stored in the solutions directory. See Sec. 15.2.3 on pg. 243 for how to archive solutions.

5.5 Reference Quantities

The reference quantities are used to non-dimensionalize data in *GASPer*. This helps control round-off error during the computation. If you set the reference quantities all to 1.0, then the non-dimensional variables used in the flow solver will equal the dimensional values. Traditionally, the reference quantities are chosen such that the non-dimensional free-stream variables are Order(1). Most often the reference values are set equal to the freestream data.

The user specifies four reference quantities (**Density**, **Temperature**, **Velocity**, and **Length**), which can then be used to compute all other reference quantities. For example, we denote the four reference quantities entered through the input deck as ρ_{ref} , V_{ref} , T_{ref} , and L_{ref} . From these four values, the following reference quantities are computed as

p_{ref}	$= \rho_{\text{ref}} V_{\text{ref}}^2$	pressure
e_{ref}	$= V_{\text{ref}}^2$	energy
t_{ref}	$= L_{\text{ref}}/V_{\text{ref}}$	time
\mathcal{M}_{ref}	$= 1$	molecular weight
\mathcal{R}_{ref}	$= V_{\text{ref}}^2/T_{\text{ref}}$	universal gas constant
μ_{ref}	$= \rho_{\text{ref}} V_{\text{ref}} L_{\text{ref}}$	molecular viscosity
k_{ref}	$= \rho_{\text{ref}} V_{\text{ref}}^2 L_{\text{ref}}/T_{\text{ref}}$	molecular conductivity
$(D_{12})_{\text{ref}}$	$= V_{\text{ref}} L_{\text{ref}}$	binary diffusion coefficient
K_{ref}	$= V_{\text{ref}}^2$	turbulence kinetic energy
ϵ_{ref}	$= V_{\text{ref}}^3/L_{\text{ref}}$	turbulence dissipation rate
ω_{ref}	$= V_{\text{ref}}/L_{\text{ref}}$	turbulence frequency

If the user wishes to set the reference quantities based upon flow conditions specified in a physical model, this can be done by pressing the **Set Quantities Using QSpec** button. By pressing this button, the reference density, temperature, and velocity will be set according to the values corresponding to the **Physical Model** and **Q Spec** option menu settings.

5.6 Reference Pressure

GASPE uses a gauge pressure for all computations involving the Navier-Stokes physical model because round-off errors can occur in the momentum fluxes of low-speed flows. Specifically, the ratio of the absolute pressure, p , to ρu^2 is $1/(\gamma M^2)$. To prevent inconsistencies across physical models, a single reference pressure is specified using the **Reference Pressure for Pgauge** input. The code stores a gauge pressure which is the absolute pressure minus reference pressure. For most applications, the reference pressure may be set to zero, allowing the code to perform all calculations using absolute pressure.

As a final note about the reference pressure, the user should not change the reference value after a problem has started. The value of the pressure stored in the solution file takes into account the reference pressure (true pressure - reference pressure). Once the problem has started, changing the reference pressure will result in an inconsistency being introduced into the problem setup, causing an error in the pressure value.

Chapter 6

The Graphics Window

6.1 Introduction

The graphics window is used to visualize the three-dimensional space associated with the current problem. From within the graphics window the user can manipulate the viewpoint (point of view), and interactively pick and modify the different graphical objects associated with the solution.

6.2 Manipulating The View

The objects associated with a *GASPer* input deck (zones, boundary surfaces, solution visualization, and output nodes) all exist in three-dimensional space. When the user manipulates the graphics window, they do not change the coordinates of the objects, they change the user's position relative to the objects. Rotating, translating and zooming in and out only change the user's view, and thus do not affect the solution itself.

The user's view of the three-dimensional object world can be described by two points, the eye location (i.e. the user's location) and the point of focus (i.e. the direction the user is looking). View manipulations change those two points in one way or another.

6.2.1 Rotation

Rotating the view changes the eye location only. Therefore, when you rotate a view, you rotate about the focal point. The distance between the eye and the focal point does not change, so that as you rotate, you do not zoom in or out. Because the focal point always appears in the center of the graphics window. Rotations will appear to be performed about the center of the screen. The user may change the center of rotation by changing the point of focus. Moving an object to the center of the view window will result in rotation about that point.

The rotate view feature employs a track-ball analogy. The track-ball has a diameter roughly 80% of the screen. Pressing the left mouse button in the track-ball will rotate the

view as if you grabbed the imaginary track-ball and moved it. Pressing the left mouse button outside of the track-ball will rotate the view about the viewer (spinning the view about the graphics window). By default the rotate view command is a rate controlled function. The more the mouse is moved after pressing the left mouse button, the faster the view will spin.

6.2.2 Translation

Translating the view changes both the eye position, and the focus position. Both points are moved in a plane which is normal to the vector pointing from the eye position to the focus position.

The translate function is rate controlled by default. Pressing the right mouse button and moving it in a given direction will translate the view in that direction. The further the user moves the mouse point, the faster the translation rate.

6.2.3 Zooming

Zooming the view will change the distance between the eye and the focus point. Only the coordinates of the eye are changed.

The zooming function moves the viewpoint further or closer to the objects in the graphics window. Pressing the middle mouse button and moving the cursor up will move the viewpoint away from the objects, affectively zooming out, while pressing the middle mouse button and moving the cursor down will move the viewpoint toward the objects, affectively zooming in. By default the zoom view command is a rate controlled function. The more the mouse is moved after pressing the middle mouse button, the faster the view will zoom.

Zoom To Object

The user can quickly manipulate the view point to find objects of interest with the zoom to object function. This function is activated by selecting **Commands**→**Zoom Selected** from the menu, or by using **[Ctrl]** - **[Z]** from the keyboard. Once activated, the zoom to object function will reset the view point such that the selected object fills the screen, and the center of rotation will be approximately the center of the selected object.

6.3 Interactive Picking Objects

The user can select objects within the graphics window by positioning the mouse button over the appropriate object in the graphics window, and pressing a modifier key (*i.e.*, **[Alt]** , **[Alt]** - **[Shift ↑]**) in combination with the left mouse button. Picking objects is limited to those elements which are expanded within the tree view. For example, if only the “All Objects” object is visible in the tree view, then selecting an object within the graphics window, will only select “All Objects”, not any of its children. Therefore, it will be

necessary to expand the Surfaces folder before individual surfaces can be selected for moving into appropriate surface groups.

6.3.1 Single Select

The single select function will de-select all objects, and select a single object. To do so, position the cursor over an object within the graphics window. Pressing the **[Alt]** - left mouse will clear all other selected objects, and select the single object.

6.3.2 Multiple Select

If the cursor is not over an object when the **[Alt]** - left mouse is pressed, then the user can hold the button down, and drag the mouse over a region. When the button is released, all objects within the drag region will be selected.

6.3.3 Toggle Select

Often the user will want to select multiple objects, but the drag select method will be inappropriate. To interactively select multiple objects one at a time, position the cursor over an object within the graphics window. Pressing the **[Alt]** - **[Shift ↑]** - left mouse will toggle the selection of the object. If the object is not selected, toggling will add this object to the selected list. If the object is currently selected, toggling will remove it from the selection list.

If the cursor is not over an object when the button is pressed, then the user can hold the button down, and drag the mouse over a region. When the button is released, a toggle select action will occur for each object within the drag region.

6.3.4 Cycle Select

The pick algorithm will return all objects that are within six pixels of the mouse button when the select function is called. However, depending upon the resolution of the view, and the number of visible objects, there may be more than one object within the pick region. If a single select is to occur, the pick algorithm must select just one of the objects within the pick list. The cycle select function, (pressing **[Ctrl]** - **[N]**) will unselect the previously selected object, and select the next object found within the pick region. Repeated key presses will cycle through all of the objects within the pick region in a circular fashion.

Chapter 7

The Tree View

7.1 Introduction

The tree view, Fig. 7.1 on pg. 78, located in the upper left corner of the main window provides an hierarchical representation of all of the elements of the current solution which may be graphically viewed. This includes not only individual zones and boundary surfaces, but also solution visualization entities such as solution contours, vectors, and stream tracers. Within the tree view, these objects are organized within folders, which may be expanded and minimized as the user requires.

All objects within the tree view must reside beneath the main folder “Flow Solver”. Immediately beneath the “Flow Solver” folder are five children; “Zones”, “Zonal Boundaries”, “Surfaces”, “Solution Visualization” and “File Output”. No other tree items may be placed directly within the “Flow Solver” folder.

7.1.1 Zones

Within the “Zones” folder, the user will find entries or nodes which represent the individual zones of the current problem. When a grid is imported into the input deck, a zone node is added for each zone of the imported grid. Each zone will be given a name corresponding to the original grid file name, and the zone number within that individual file. The names of the zone nodes assist the user in identifying the geometry, and may be altered as needed.

The zone nodes may also be organized within the “Zones” folder to suit the current problem. For example, the user must identify which zones to iterate upon within the “Run Group” section of a run definition. These zones are identified by specifying a zone folder. All zones within a specified zone folder will be integrated in time for a given run group. By default, the run group uses the “Zones” folder, so that every zone within a given problem is integrated in time. However, the user may choose to run a subset of the zones. This is done by creating a new folder within the “Zones” folder, and then placing the zones for a given run group within that folder.

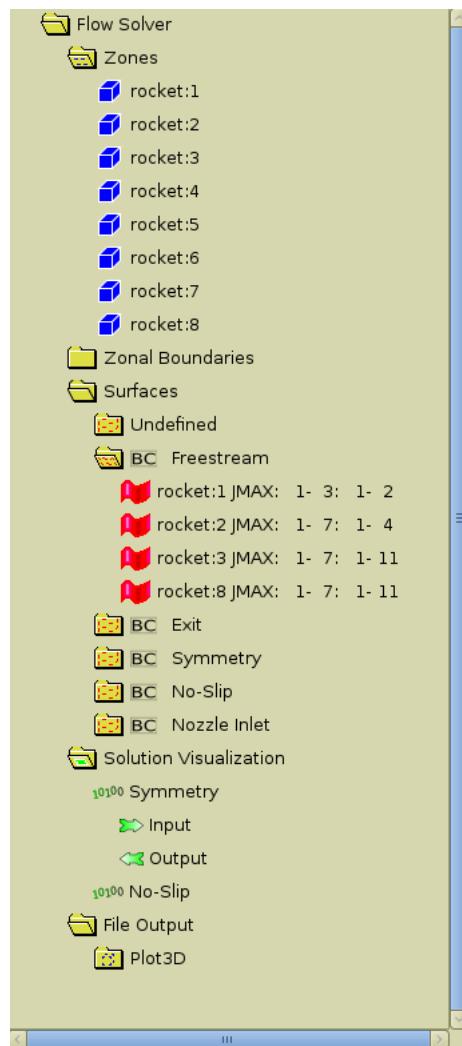


Figure 7.1: The Tree View

7.1.2 Zonal Boundaries

Within the “Zonal Boundaries” folder, the user will find nodes of the tree which represent the surfaces used to communicate from one zone to another. Zonal Boundaries may be further categorized into point-to-point boundaries and chimera boundaries. Several specialty folders are created within the “Zonal Boundaries” folder by default; “Pt2Pt ZB”, “Chimera ZB”, “Cutting Overrides”, “Chimera Overrides”, and “Zipper Grids”. The “Pt2Pt ZB”, folder will contain all surfaces which define point-to-point zonal boundaries. The point-to-point zonal boundaries are detected automatically in the **Pt-2-Pt ZB** tab of the **Zonal Boun** section. All surfaces which represent overset grid interfaces should be placed in the “Chimera ZB” folder. The “Cutting Overrides”, “Chimera Overrides”, and “Zipper Grids” folders are discussed in the Chimera section of the reference manual.

7.1.3 Surfaces

Within the “Surfaces” folder, the user will find nodes of the tree which represent the individual boundary surfaces of the current problem. An “Undefined” surface folder is created within the “Surfaces” folder by default. All imported surfaces which have not been assigned to a specific folder are placed within the undefined folder. Before the *GASPE* flow solver is run, surfaces should be moved from the “Undefined” folder to an appropriate surface folder, which is usually created by the user.

A “Singular” folder will be created if the automatic singular line detection algorithm identifies any singular lines. Singular lines and points may be automatically detected within the **Pt-2-Pt ZB** tab of the **Zonal Boun** section.

When a structured grid is imported into the GUI, six surface nodes are added for each zone of the imported grid, corresponding to the six logical boundaries of the structured zone ($i=0$, $i=i\text{Max}$, $j=0$, $j=j\text{Max}$, $k=0$ and $k=k\text{Max}$). Each surface will be given a name corresponding to the original grid file name, the zone number within that individual file, and the logical surface it represents. The names of the surface nodes assist the user in identifying the geometry, and may be renamed as needed.

When an unstructured grid is imported into the GUI, surfaces defined in the grid file are set as surface folders in the tree view, and BC's are activated for each folder and set to the default values. Zonal boundaries defined in the grid file are also automatically imported.

As part of the problem setup, the user must group the surfaces into folders which describe the topology and organization of the problem. As mentioned above, this is typically already done in the grid file for unstructured grids, but for structured grids, it is typically done by the user after importing the grid. For example, the user might create a surface folder which is called “Aircraft Surface”. This folder may be used to hold all boundary surfaces which are on the surface of the aircraft. Likewise, the user may create a surface folder which is called “Far-Field”, which will contain all surfaces on the far-field boundary. Once all the boundary surfaces are grouped into appropriate folders, the user may use these folders to perform additional operations, such as describing overset grid cutting surfaces, and applying boundary conditions.

Note: *GASPer* supports the import of CGNS grid files. With CGNS, if surfaces and boundary conditions are set up in the grid generator, the information can be imported along with the grid. This can help assist the user during the input deck set up process.

Creating a BC Folder

Once a surface folder has been created, the user may specify the folder as a boundary condition. This is done using the right mouse button and selecting **Toggle Folder BC On/Off** under the **Edit** option. To indicate that a BC is being applied to the folder, a “BC” icon will appear next to the folder and before the folder name. The boundary condition value is set inside the physical model (see Sec. 10.9 on pg. 159).

7.1.4 Solution Visualization

All objects associated with on-screen solution visualization are placed within the “**Solution Visualization**” folder. By default, no visualization nodes are present within the folder initially. The user must explicitly create visualization nodes for a given problem.

For details on how to set up solution visualization nodes, the user is encouraged to go through the visualization tutorial located in the `aerosoft/share/tutorial` directory.

7.1.5 File Output

All objects associated with exporting post-processing solution data, and solution monitoring are placed within the “**File Output**” folder. By default, no output nodes are present within the parent folder initially. The user must explicitly create any output nodes required for a given problem.

7.2 Tree Node Properties

Every node in a tree has properties which are associated with it. These properties affect the graphical representation of that node, as well as its children. The windows that represent the display values of the properties are consistent with the currently selected tree nodes. In particular, the render property window, Fig. 7.2 on pg. 81, represents the node properties associated with basic rendering for the currently selected nodes. If more than one node is selected, the widget represents a composite view of all selected nodes. Changing a value for the property will set that property for all selected nodes. For example, to change the line thickness used to render the boundary surfaces, one would select all the boundary surfaces, and then change the **Line Width** from the render property window.

Node properties are recursive. In other words, child nodes inherit certain properties from their parents. For example, it is possible to set the line color for all children of a single folder, simply by setting the line color for that folder.



Figure 7.2: The render property window.



Figure 7.3: Tree node icons showing different render modes.

7.2.1 The Render Property

The render properties are used to control the level of detail for the graphical display of a tree node and all of it's children. In particular, there are four levels of display detail which may be selected; “None”, “Sparse”, “Wire”, and “Full”. If “None” is selected, an object is not rendered in the 3-D viewing window. This mode is appropriate for temporarily or permanently disabling the display of a given object and its children. “Sparse” display mode should provide sufficient detail to identify or graphically pick an object, yet still require minimal display time. “Sparse” display mode is appropriate for interactive object manipulation on slower workstations, and for editing very complex scenes. “Wire” display mode will render solid surfaces with a wire frame representation. In particular, surface grids will be rendered as wire frame grids, appropriate for visualization of grid point distribution. Finally, “Full” display mode will render solid surfaces with a fully shaded representation with a lighting model.

The interaction of the render property between a node and it's children is different from other properties. In particular, the render property will act like a filter to all of a node's children. If a parent node has a more restrictive drawing mode than a child, then the child will be rendered with that restrictive render mode, even though its render mode is less restrictive. As an example, if you set the render mode for the “Flow Solver” to “None”, then no graphics will be rendered, since all nodes are children of the top “Flow Solver” folder. By limiting the display mode for the top level folder/node, the display of a node and it's children will be limited as well.

Note:

If you change the render mode of a node, and there is no effect, then it is likely that a parent node has a more restrictive render mode. However, this parent may be more than one level up the tree hierarchy.

The icons used to identify the node types change according to the render mode. The icon types should assist the user in identifying the current render mode. Icons for the different render modes are shown in Fig. 7.3 on pg. 81.

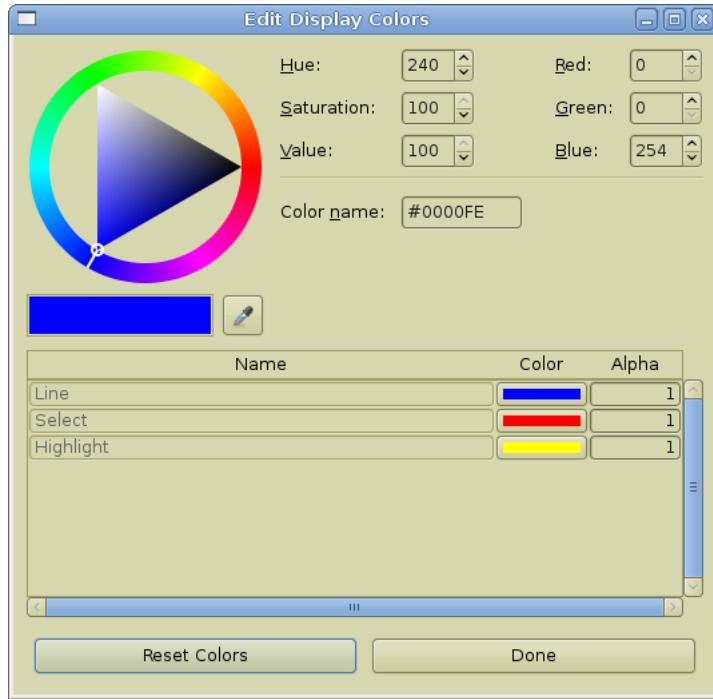


Figure 7.4: The Node Color Selection Dialog

7.2.2 The Color Property

The color properties are used to specify the colors used to render the current object. The color properties are now presented in a pop-up dialog which is displayed as in Fig. 7.4 on pg. 82 when the **Colors...** button is pressed. The types of colors are presented in a list at the bottom of the dialog. For each color type, the user can control the color and the transparency value. In order to change a given color, press on the appropriate color button and the color selection widget at the top of the dialog will be set for editing that particular color. Any changes to that color should be immediately visible. The **alpha** value for each color type should be a real value in the range of 0 to 1, with 0 representing a completely transparent rendering and 1 representing a completely opaque rendering.

By default, all nodes have the following colors

1. “Line”: The “Line” color is used when “Sparse” and “Wire” render modes are selected. When a lighting model is used for fully shaded objects, *i.e.*, “Full” rendering, The material property is also constructed from the “Line” color.
2. “Select”: The “Select” color is used to identify selected nodes in the display window.
3. “Highlight”: The “Highlight” color is used to highlight certain nodes in the display window. Highlighting is a visual feedback to assist in the drag-n-drop process. When the user attempts to drag-drop a selected node into a new folder, the folder which will accept the drop is highlighted on the display.

As with most other properties, it is possible to simultaneously change the color values of more than one node. This is done by using the multiple selection option within the tree view widget.

If after changing color values in the color dialog, the user wishes to revert to the original colors, they should press the **Reset Colors** button.

7.2.3 The Line Width Property

The **Line Width** option menu is used to select the pixel width used for wire frame and sparse rendering of the given object. Integer pixel widths from one to four points may be selected.

7.2.4 Use Parent

The **Use Parent** toggle button is used when the tree node should set the colors and line width based upon the parent values. By default, all base nodes (Surface, Zone, and Visualization nodes) have the **Use Parent** flag set to true. Setting the color for the parent folder will recursively set the colors for all of the child nodes.

In order to avoid confusion, when the **Use Parent** toggle button is set for a given node, the color selection widgets are disabled. It will therefore not be possible to change the node colors unless the **Use Parent** toggle button is off.

7.3 Manipulating the Tree

7.3.1 Expanding and Collapsing Tree Nodes

The children of a given node of the tree are visible only when the parent node is in the expanded (or open folder) mode. If a given node is collapsed, then none of the individual child nodes are visible in the tree view, and the user cannot individually select those children. If the user interactively selects a child element through the 3-D graphics window, Sec. 6.3 on pg. 74, the first parent node which is visible within the tree view will be selected.

7.3.2 Renaming a Tree Node

All tree nodes (except the default folders, such as “Flow Solver”) can be re-named according to user preference. To do so, simply position the cursor over the tree node text and double click with the left mouse button. If it is possible to edit the text of the current node, then the display will change to a text type-in widget, as shown in Fig. 7.5 on pg. 84. Edit the text as you normally would in a text widget, and press the **Enter** key to accept the changes. Note that if you make changes to the text and lose focus, then those changes will also be registered; using the **Enter** key is not required.

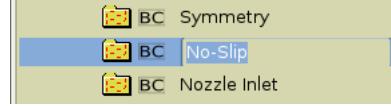


Figure 7.5: Editing Text In A Tree Node

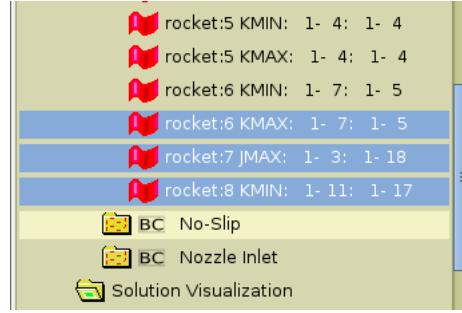


Figure 7.6: Moving three zone nodes into the “No-Slip” surface folders.

7.3.3 Selecting Tree Nodes

Many operations within the user interface require one or more elements of the object tree to be selected. Selection of tree elements occurs either through interactive picking with the 3-D viewing window, as described in Sec. 6.3 on pg. 74, or by selecting the elements directly from the tree view. Selection from the tree view widget is similar to standard selection of a multiple select list widget. To select a single node, position the cursor over the appropriate node and click the left mouse button. To select a range of elements, position the cursor over the first node. Depress the left mouse button and drag the cursor over the range of nodes to be selected. Upon reaching the last node, release the button. In order to toggle the selection of a node, depress the **[Ctrl]** key while selecting a single node with the left mouse button.

7.3.4 Moving Tree Nodes

The tree view can be re-organized by creating new nodes, deleting existing nodes, and moving existing nodes to alternate locations. In order to move an existing tree node or nodes, the user must first select the nodes they wish to move, as described in Sec. 7.3.3 on pg. 84. To move a node, first position the cursor over any one of the selected nodes and press the middle mouse button. While holding the middle mouse button, move the cursor to the location to which you wish to move the selected items. If it is possible to move the selected items to the desired location, the cursor will change from an arrow to a finger. If you are moving into a different folder, a folder which will accept a drop will change color to a darker shade as in Fig. 7.6 on pg. 84, signifying an acceptable move. Some nodes may not be moved. For example, it is not possible to move a folder node into one of its children. It is also not possible to move zone nodes into surface folders, and vice versa. The user interface will prevent these types of operations.

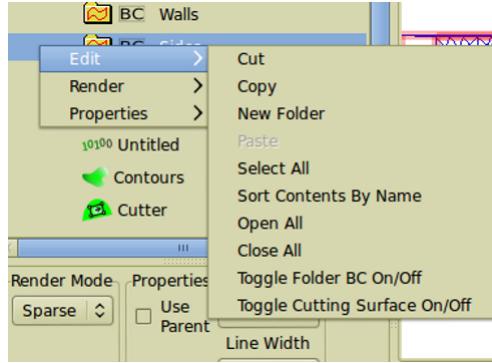


Figure 7.7: The Tree view widget **Edit** Menu.

A second method for moving selected objects into an acceptable folder can be simpler than the method described above. Once a group of tree nodes is selected, position the cursor over the folder which will receive the selected nodes. If moving the selected nodes into that folder is possible, the receiving folder will be highlighted. Pressing the middle mouse button over the receiving folder will pop up a menu entitled **Drop Selected**. Selecting the **Drop Selected** menu will drop the selected items into that folder.

7.4 Pop-Up Menus

Pressing the middle or right mouse buttons while in the Tree View window may display a context sensitive pop-up menu. Pressing the right mouse button will always display a pop-up. However, the middle mouse button will only display a pop up when the cursor is over an acceptable donor node for the drag drop operation.

7.4.1 The Edit Menu

The **Edit** Menu, Fig. 7.7 on pg. 85, will be displayed whenever the right mouse button is depressed in the Tree View window. The only exception to this is when the cursor is positioned directly over a node icon (*i.e.*, a folder icon). From the **Edit** Menu, the user will be allowed to create new nodes, cut selected nodes, copy selected nodes, or paste previously copied nodes. These operations are with respect to the currently selected nodes. However, each of these operations is context sensitive. For example, new nodes cannot be created when a non-folder node is selected.

When the **New Folder** button is pressed while a folder is selected, a new folder of an appropriate type is created within the selected folder. Each new folder will be assigned the default name “**Untitled**”.

When the right mouse button is pressed while one or more tree nodes are selected, the **Cut** button found in the **Edit** pop-up menu will be enabled when it is possible to cut or delete each of the selected nodes. If any or all of the selected nodes are not able to be cut, then

the **Cut** button will be disabled. Selecting the **Cut** when enabled will remove all selected nodes from the tree, and permanently delete these entities. All nodes that are children of the selected nodes will also be cut.

If, in the process of a cutting a node, a surface node is deleted, this entry will be moved to the “**Undefined**” folder within the “**Surfaces**” folder. Surface nodes are only deleted from a given geometry when the corresponding zone is also deleted. Otherwise, surface nodes must be conserved.

Copying and pasting of nodes has similar semantics to the Cut operation

7.4.2 The Drop Selected Menu

The last possible pop-up menu that you might encounter is the drop selected menu. This was already discussed in Sec. 7.3.4 on pg. 84. In that section, it was explained how to move a selected node (tree object) from one folder to another. This was accomplished by selecting a tree node (tree object), and then positioning the cursor over the folder which will receive the selected node. Pressing the middle mouse button over the receiving folder will pop up a menu entitled **Drop Selected**. Selecting the **Drop Selected** menu will drop the selected items into that folder.

Chapter 8

The Zones Frame

8.1 Introduction

The **Zones** frame is comprised of tabs which allow the user to manipulate and set parameters for the zones (including the grid, solution, surfaces, etc.). The data in the tabs is tied directly to the current time level and sequence level selected in the **Time Level** and **Sequences** lists (see Fig. 8.1 on pg. 87). Several tabs are also dependent upon which zone nodes are selected in the tree view.

Here is a brief description of each of the Zone tabs:

Initialization This allows the user to specify input parameters for each zone's solution initialization.

Sequencing This is used to create and perform mesh sequencing. Sequencing is the process of creating a coarse grid from a fine grid.

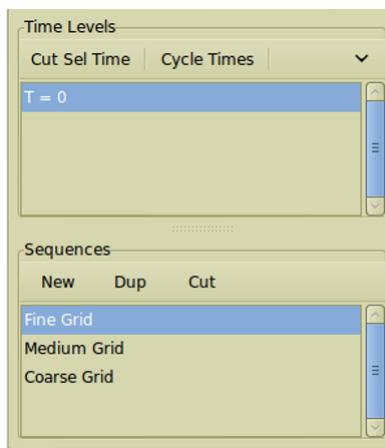


Figure 8.1: Time Level and Sequences lists

Zone	Soln For PhysMod 1: ▾	Idim	Jdim	Kdim	nSoln	PhysMod	QSpec
Nose 1		9	49	13	5	PhysMod 1: ▾	Initial Q ▾
Nose 2		17	49	29	5	PhysMod 1: ▾	Initial Q ▾
Body		45	49	29	5	PhysMod 1: ▾	Initial Q ▾
Outer Nozzle 1		17	33	29	5	PhysMod 1: ▾	Initial Q ▾
Outer Nozzle 2		17	17	29	5	PhysMod 1: ▾	Initial Q ▾
Inner Nozzle 1		29	21	29	5	PhysMod 1: ▾	Initial Q ▾
Inner Nozzle 2		73	9	13	5	PhysMod 1: ▾	Initial Q ▾

Set all zones in table based on first entry Set All Zones In Table

Figure 8.2: The **Initialization** tab

Grid Mgt This is the grid management tab which allows the user to manipulate the grid data. This includes deleting the grid data from file, replacing the grid data, transforming the grid, and deleting wall distance data.

Soln Mgt This is the solution management tab which allows the user to manipulate the solution data. This includes interpolating the solution between sequences, initializing the solution, deleting the solution from file, importing a new solution, removing particle data from file.

Edit Surface This allows the user to manually split or combine surfaces along a grid line. This is applicable to structured meshes.

Interp Coefs This computes the overlap coefficients used with the conservative volume overlap solution interpolation located in the Zone Algorithms tab.

Zone Algorithms Performs a number of zone operations such as unstructured grid adaptation, conservative solution interpolation, and structured mesh shock fitting.

The following sections will describe in detail the function of each of these tabs.

8.2 Zone Initialization

Parameters relevant to a zone's solution initialization are edited by selecting the **Initialization** tab, shown in Fig. 8.2 on pg. 88. In order to enable a zone or group of zones for editing under the **Initialization** tab, the user must first select the zone or group of zones in the tree view. Only zones selected in the tree view will appear in the zone initialization table. If no zones are selected, then the table will display all zones.

8.2.1 Zone Name and Dimensions

The first entry in the zone initialization table is the zone name. The zone name table entry is not editable. It is a reflection of the name given the zone in the tree view. If the user changes the name of the zone in the tree view, it will be updated in the table. Another characteristic of the zone name is that it is independent of the sequence level.

The 2nd, 3rd and 4th entries in the table are the dimensions for the zone. These are also un-editable values. For the fine sequence, they are set when the grid is imported. For all other sequences, they are a function of the fine sequence dimensions and the sequencing levels which are set in the **Sequencing** tab. If we have more than one sequence defined, and we switch sequences by selecting a different sequence from the **Sequences** list in the lower left corner of the GUI, then the dimensions in the table will be updated to reflect the values of the new current sequence level.

8.2.2 Soln For and Initial Physical Model and Q Specification

For each parent (black) physical model listed in the **Soln For PhysMod** option menu, the user can set a compatible physical model with which to initialize the solution for each zone. The compatible physical models in the table will show up as black in the individual zone init **PhysMod** option menus, and the incompatible ones will be in grey. Selecting any of the children (red) physical models in the **Soln For** option menu is the same as selecting its parent (the first black one above it) since only one solution exists for a parent physical model and all of its children.

The **PhysMod** option menu for each zone in the table selects which compatible physical model is used to initialize that zone's solution for the current sequence level. The **Q Spec** option menu determines which Q specification within the chosen physical model is used to initialize the values of the solution vector for each zone.

If the term “N/A” appears in the **nSoln** column of the table, it means the solution is not available. In other words, the physical selected under **Soln For** does not support a solution on those zones. Solution management is performed in the **Soln Mgt** section.

The physical models selected for a given zone must be consistent across all sequence levels. In other words, if zone one of the fine sequence is initialized with a five species air chemistry model with a k -omega turbulence model, then so must zone one for all of the other sequences. This requirement is left to the user to enforce.

8.2.3 When Does Initialization Occur

Initialization of the solution may occur at two different times. First, the user may elect to initialize the solution from the **Soln Mgt** tab (see Sec. 8.5 on pg. 96 for more information). The initialization would then occur within the GUI and the solution would be saved to file

when the GUI save command is performed. The user may initialize the solution regardless of whether a solution already exists or not.

The second way for a solution to be initialized is through the flow solver. When the flow solver starts, the first thing it does is check to see if a solution exists in file. If a solution does not exist for the sequence being simulated, the flow solver will initialize the solution according to the settings in this tab. Because of this behavior, the user will never need to be concerned about creating (or initializing) a solution before running the flow solver.

One final comment will be made about solution initialization. If a solution has been created (either by the GUI or by the flow solver), and the user wishes to reset the solution back to the initial **Q Spec** values, then one of two things can be done. The user can do this using the GUI as described above, or use the **Re-Initialize Solution** flag in the run definition to force the flow solver to do it when the run definition is executed.

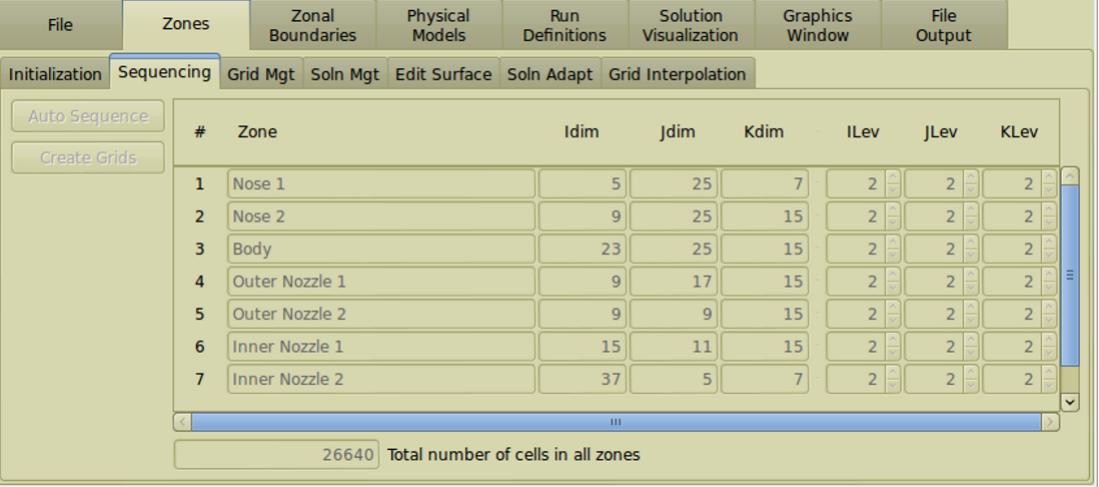
8.3 Sequencing

Mesh sequencing is a powerful tool that can help reduce the required CPU time needed to obtain the final solution to a problem. This feature should be exercised whenever possible. Sequencing operations are always performed on all zones in order to maintain consistency across point-to-point zonal boundaries within a sequence level. Therefore, in the **Sequencing** tab, all the zones are always displayed in table format with their sequencing information. This differs from the **Initialization** tab which only displays those zones which are currently selected in the tree view. In addition to the zone table, the **Sequencing** tab also contains two buttons for setting up and creating the sequenced grids. All of the parameters in the **Sequencing** tab are a function of the current sequence level as displayed in the **Sequences** list located in the lower left corner of Fig. 8.3 on pg. 91. For this example, there are three sequences: fine, medium, and coarse, and we are currently viewing the medium sequence level.

8.3.1 Zone Name and Dimensions

The first entry in the zone sequencing table is the zone name. The zone name table entry is not editable. It is a reflection of the name given the zone in the tree view. If the user changes the name of the zone in the tree view, it will be updated in the table. Another characteristic of the zone name is that it is independent of the sequence level.

The 2nd, 3rd and 4th entries in the table are the dimensions for the zone (IDim, JDim, KDim for structured grids and Cells, Nodes, Faces for unstructured grids). These are also un-editable values. For the fine sequence, they are determined when the grid is imported. For all other sequences, they are a function of the fine sequence dimensions and the sequencing levels discussed in the next section. If the sequence level is changed, then the dimensions in the table will be updated to reflect the new sequencing level values.



The screenshot shows a software interface with a menu bar at the top containing 'File', 'Zones' (which is highlighted in blue), 'Zonal Boundaries', 'Physical Models', 'Run Definitions', 'Solution Visualization', 'Graphics Window', and 'File Output'. Below the menu is a toolbar with buttons for 'Initialization', 'Sequencing' (which is selected and highlighted in blue), 'Grid Mgt', 'Soln Mgt', 'Edit Surface', 'Soln Adapt', and 'Grid Interpolation'. On the left, there are two buttons: 'Auto Sequence' (disabled) and 'Create Grids'. The main area is a table titled 'Sequencing' with columns for '#', 'Zone', 'Idim', 'Jdim', 'Kdim', 'ILev', 'JLev', and 'KLev'. The table contains seven rows of data:

#	Zone	Idim	Jdim	Kdim	ILev	JLev	KLev
1	Nose 1	5	25	7	2	2	2
2	Nose 2	9	25	15	2	2	2
3	Body	23	25	15	2	2	2
4	Outer Nozzle 1	9	17	15	2	2	2
5	Outer Nozzle 2	9	9	15	2	2	2
6	Inner Nozzle 1	15	11	15	2	2	2
7	Inner Nozzle 2	37	5	7	2	2	2

At the bottom of the table, a status bar displays '26640 Total number of cells in all zones'.

Figure 8.3: The sequence information for the medium structured grid.

8.3.2 Structured Mesh Sequencing Levels

When a structured mesh has been imported, the last three entries in the table are the sequencing levels for each of the i , j and k grid coordinate directions. For the original grid (*i.e.*, the finest grid), the values for the “ILev”, “JLev”, and “KLev” input boxes are all set to 1, meaning that every grid line is kept. Since the finest grid exists either in memory (after importing) or in the .grd file, the sequence level input boxes are not editable for this sequence level. When a new sequence is created in the **Sequences** list, the values for the “ILev”, “JLev”, and “KLev” input boxes are initially set to those of the next finer sequence level. So if we started with the finest grid and created a second sequence level, these values would initially all be set to 1. The “ILev”, “JLev”, and “KLev” values can be change either globally by selecting the **Auto Sequence** button or by changing the individual values in the input boxes.

The **Sequencing** tab shown in Fig. 8.3 on pg. 91 is for a medium sequence grid system where each of the coordinate directions for all of the grids have been sequenced by 2. Notice that the number 2 appears in all of the “ILev”, “JLev”, and “KLev” input boxes. A value of 2 in a specific direction indicates that every other grid line will be retained for this grid level, thus reducing the dimension in that direction by a factor of two. This sequence was created by selecting the **Auto Sequence** button followed by the **Create Grids** button.

Note that the “ILev”, “JLev”, and “KLev” input boxes in Fig. 8.3 on pg. 91 are non-editable. Also the **Auto Sequence** and **Create Grids** buttons are disabled. This is because the grids have already been created for this sequence. Had we just created a new sequence, but not yet created the grids, then the coordinate sequence levels would be editable, and the buttons would be selectable. Because the finest grid sequence levels cannot be changed, it is always un-editable in the **Sequencing** tab and thus for viewing purposes only.

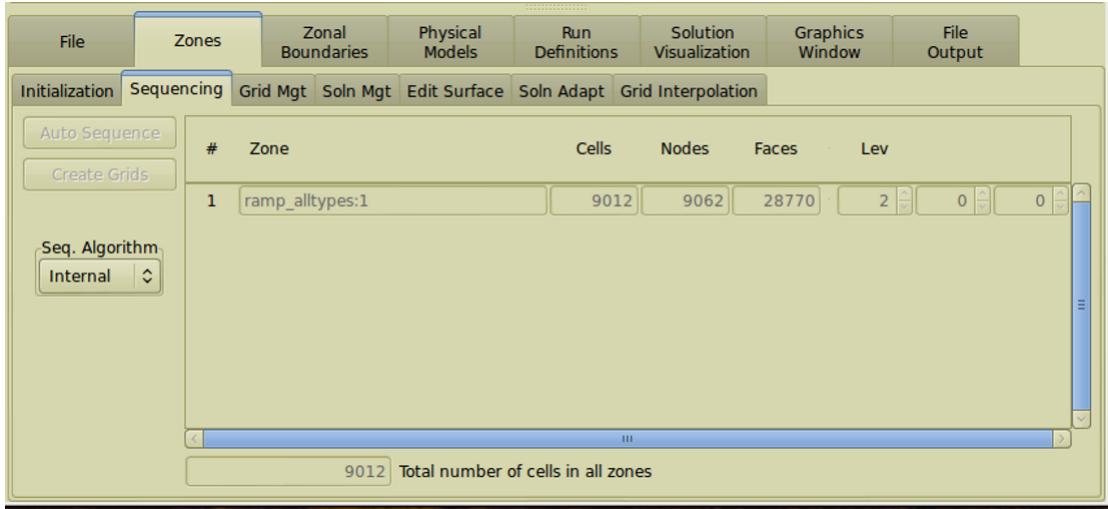


Figure 8.4: The sequence information for the medium unstructured grid.

8.3.3 Unstructured Mesh Sequencing Levels

When an unstructured mesh has been imported, only the first of the last three entries in the table is used as the sequencing level.

For the original grid (*i.e.*, the finest grid), the value for the “Lev” input box is set to 1, meaning that every cell is kept. Since the finest grid exists either in memory (after importing) or in the .grd file, the sequence level input box is not editable for this sequence level. When a new sequence is created in the **Sequences** list, the value for the “Lev” input box is initially set to those of the next finer sequence level. So if we started with the finest grid and created a second sequence level, the value would initially be set to 1. The “Lev” value can be changed either globally by selecting the **Auto Sequence** button or by changing the individual value in the input box.

The **Sequencing** tab shown in Fig. 8.4 on pg. 92 is for a medium sequence grid system where the grid has been sequenced by 2. Notice that the number 2 appears in the “Lev” input box. A value of 2 indicates that the sequenced grid will have approximately 1/2 the number of cells. This sequence was created by selecting the **Auto Sequence** button followed by the **Create Grids** button.

Note that the “Lev” input box in Fig. 8.4 on pg. 92 is non-editable. Also the **Auto Sequence** and **Create Grids** buttons are disabled. This is because the grids have already been created for this sequence. Had we just created a new sequence, but not yet created the grids, then the sequence level would be editable, and the buttons would be selectable. Because the finest grid sequence levels cannot be changed, it is always un-editable in the **Sequencing** tab and thus for viewing purposes only.

The unstructured **Sequencing** tab has one additional option menu, the **Sequence Algorithm** which gives the user a choice between two agglomeration algorithms to form the sequenced grid. The default choice is **Internal** which is an algorithm written specifically for

GASP, and the other choice is the **MGridGen** algorithm.

Auto Sequencing and Create Grids

The **Auto Sequence** button is used to automatically set up the next “logical” sequence for a user. For structured grids, the algorithm attempts to increment each of the “ILev”, “JLev”, and “KLev” values for all of the grids to the next allowable values. Starting from the fine grid in which all of the values are 1, the **Auto Sequence** algorithm would first check to see if a value of 2 for each of the coordinate levels is allowable. Whether or not a value is allowable is a function of many parameters including grid dimensions, surface dimensions, zonal boundary dependencies, and the “ILev”, “JLev”, and “KLev” values of adjacent sequences. If a value of 2 is not allowable, then the algorithm will check a value of 3, and so on up to a limit of either the values of the next coarser sequence level or $3^*(\text{current sequence level number})$ - in this case, it’s the second sequence and assuming there is no coarser sequence, so that would be $3^*(2)=6$. If we were creating a third sequence and the second sequence had values of 2, then the difference would be that the algorithm would start by checking values of 4, and then 6, and so on up to a maximum value $3^*(3)=9$, but really 8 since 9 is not a multiple of 2. If the second sequence had values of 3 instead of 2 for “ILev”, “JLev”, and “KLev”, then the **Auto Sequence** would check for allowable values of 6 and 9. For unstructured grids, the **Auto Sequence** algorithm simply divides the next finer sequence by a factor of 2.

After we are done editing the “ILev”, “JLev”, and “KLev” or “Lev” values either through use of the **Auto Sequence** button or by manually editing the input boxes, we are ready to create the grid for the new sequence level. This is done by pressing the **Create Grids** button. After this has been done, the new sequence level is no longer editable, and the new grid is in memory. When the input deck is saved, the new grids will be saved to the binary grid file.

For structured grids, when manually editing sequence levels, values for topologically connected zones will be maintained in a consistent state so as to always represent a valid sequence. For example, consider a two zone problem with a zonal boundary connecting the iDim boundary of the first zone with the i0 boundary of the second zone. If the user changed the “JLev” or “KLev” of zone one, the “JLev” or “KLev” values in zone two will automatically be updated to the same value.

Please note that in *GASPex*, it is necessary to create the sequenced grids before running the flow solver. The flow solver is unable to create the coarser grids. Instead, they must be created from within the GUI before running on that sequence level. A sequenced grid is only created after pressing the **Create Grids** button.

8.4 Grid Management

Manipulation of a zone’s grid is done by selecting the **Grid Mgt** tab, (see Fig. 8.5 on pg. 94). In order to enable a zone or group of zones for editing under the **Grid Mgt** tab, the user

#	Zone	Sort By			Grid Mem	Grid File	Wall Dist
		Tree Order					
1	rocket:1	38	17	8	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
2	rocket:2	17	17	22	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
3	rocket:3	15	11	17	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
4	rocket:4	19	17	9	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
5	rocket:5	21	11	3	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
6	rocket:6	19	11	17	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
7	rocket:7	10	13	17	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
8	rocket:8	19	17	10	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Figure 8.5: The Grid Mgt tab.

must first select the zone or group of zones in the tree view. Only zones selected in the tree view will appear in the zone management table. If no zones are selected, then all zones will appear in the table.

8.4.1 Table Parameters

The first entry in the zone grid management table is the zone name. The zone name entry is not editable. It is a reflection of the name given the zone in the tree view. If the user changes the name of the zone in the tree view, it will be updated in the table. Another characteristic of the zone name is that it is independent of the sequence level.

There are several options for selecting the order in which zones appear in the table. The **Sort by ...** option menu allows the user to change the display order based on the following criteria.

Tree Order The order in which zones appear in the tree view. Selecting this option allows the user to arrange the order in the table by moving zones around in the tree view (under the **All Zones** folder).

Import Order The order in which zones were imported into the GUI. This is a fixed order which may sometimes be useful if replacing the grid of a large number of zones.

Name The alphabetical order based on zone name.

After the zone name, specific zone information is given in the table. For structured grids, this comprises the I, J, K grid coordinate dimensions. For unstructured grids, the information displayed is the number of cells, nodes and boundary faces for the zone. The grid information is dependent upon which sequence is selected. Changing the sequence (by selecting a different sequence from the **Sequences** list) will cause information in the table to be updated.

The **Grid Mem** and **File** headers are used to display the current state of the grid. This comes in the form of non-editable check boxes which informs the user which zones are

currently in memory and/or the file. This can be useful for several reason. For example, a grid that is in memory but not in the file indicates that there has been a change, and that the file is not current with that change. Therefore we need to save before exiting if we want that change to be made to the file. It also gives the user an idea of which grids are currently in use (*i.e.*,in memory) in the GUI which can be useful when memory limits are a concern.

8.4.2 Delete Grids Option

The **Delete Grids** option is used for deleting the existing grids in the selected zones for the current sequence. Only those zones displayed in the table will have their grids deleted.

This command does not remove the zone from the GUI, but only the X,Y,Z grid values from both the memory and file locations. This is most useful for deleting unneeded grids, and thus reducing the file size. File size will only decrease if the grid being deleted is stored last in the grid file or by using the **Compress Grid File** option under **File** in the menu bar.

Note: In order to completely remove a zone from the input deck, the user needs to select the zone inside the tree view and then select **Edit→Cut**.

8.4.3 Replace Grids

The **Replace Grids** option is used for replacing existing grid values in the selected zones for the current sequence. This is useful for changing the distribution of points in a grid (*i.e.*,the X, Y, and Z values). For example, the user may import a grid and later decide to change some feature of the grid. As long as the grid dimensions and zone connectivity have not changed, then this allows an easy way to update the grid.

If the user wants to change the grid dimensions, then the **File→Import** must be used to add a completely new grid. Then, the **General Interp Solns** option can be used to interpolate the solution onto the new grid. Finally, the old zone can be deleted by selecting it in the tree view and using **Edit→Cut**. The new zone may then be set up (*i.e.*,pt2pt ZB's, assign the surfaces to folders, etc).

8.4.4 Transform Grids

The **Transform Grids** feature is used to perform operations on the grid. For example, the user may scale the grid, translate or rotate the grid, or flip the grid about one of the three major axis. When this button is pressed, a pop-up dialog will appear giving the user various options for manipulating the zones. Note that operations will only be performed on the zones displayed in the grid table.

8.4.5 Delete Wall Distance

There are a number of turbulence models in **GASPer** that require a wall distance parameter. Because of this, **GASPer** offers several methods of computing the distance to the closest

#	Zone	Soln For	Sort By			Idim	Jdim	Kdim	nSoln	Soln
		PhysMod 1: Tree Order	1:	Tree Order						Mem File
1	Nose 1					5	25	7	5	<input type="checkbox"/> <input checked="" type="checkbox"/>
2	Nose 2					9	25	15	5	<input type="checkbox"/> <input checked="" type="checkbox"/>
3	Body					23	25	15	5	<input type="checkbox"/> <input checked="" type="checkbox"/>
4	Outer Nozzle 1					9	17	15	5	<input type="checkbox"/> <input checked="" type="checkbox"/>
5	Outer Nozzle 2					9	9	15	5	<input type="checkbox"/> <input checked="" type="checkbox"/>
6	Inner Nozzle 1					15	11	15	5	<input type="checkbox"/> <input checked="" type="checkbox"/>
7	Inner Nozzle 2					37	5	7	5	<input type="checkbox"/> <input checked="" type="checkbox"/>
8	Wake					23	35	15	5	<input type="checkbox"/> <input checked="" type="checkbox"/>

Figure 8.6: The **Soln Mgt** tab.

wall for each cell in the grid domain. Some of these algorithms (see Sec. 11.4.7 on pg. 187) can be very computationally expensive, and so an option is available that will save the wall distance to the grid file. If the user would like to remove the saved wall distance values, the **Delete Wall Distance** button should be used. This will remove the wall distance values from the grid file upon saving the input deck.

8.4.6 Display Grid Info

This option brings up a window that displays information about the mesh such as volume and x,y,z ranges. For unstructured grids, the number of cells for each cell type is given. The displayed information only applies to the zones listed in the grid table. If individual surfaces are selected, then information about those surfaces are also reported.

8.5 Solution Management

Manipulation of a zone's solution is done by selecting the **Soln Mgt** tab, (see Fig. 8.6 on pg. 96). In order to enable a zone or group of zones for editing under **Soln Mgt**, the user must first select the zone or group of zones in the tree view. Only zones selected in the tree view will appear in the zone management table. If no zones are selected, then all zones will appear in the table.

8.5.1 Soln For Physical Model

For each parent (black) physical model listed in the **Soln For** option menu, the user can perform the various solution management tasks listed below. Selecting any of the children (red) physical models in the **Soln For** option menu is the same as selecting its parent (the

first black one above it) since only one solution exists for a parent physical model and all of its children.

8.5.2 Table Parameters

The table parameters for the solution management tab are very similar to those for the grid management. The only exception is that the table under **Soln Mgt** also displays the number of solution variables stored in the solution file. This is equivalent to the number of primitive variables used in the solver.

8.5.3 Average From Finest Sequence

The **Average From Finest Sequence** option is used for averaging the solution from the fine sequence onto the current sequence. This operation applies to only those zones in the table and for the selected **Soln For**. This is useful in a case where iterations have been performed on a fine grid, and the user wants to create a coarser grid and compute a solution on it. This allows the user to initialize the coarser grid solution from the fine grid.

Note that the default behavior in *GASPer* is to interpolate the solution up to the next sequence level at the end of a run. The user can prevent this behavior using the **Interp Solution Up** option in the run definition's main tab. In other words, it is not "normal" practice to go from running on a fine grid to running on a coarse grid, so one must use care when doing so.

8.5.4 Interp from Coarser Sequence

The **Interp from Coarser Sequence** option is used for interpolating the solution from the next coarser sequence onto the current sequence selected in the sequence list. This applies only to those zones displayed in the table and for the selected **Soln For** physical model. Typically at the end of a run on a given sequence, the solution is interpolated up to the next sequence level. However, in some cases this is not done, such as when a run is interrupted before completing, or the default interpolation behavior is overridden. In these cases, the user can perform the interpolation command from the GUI without having to run the flow solver to perform the interpolation.

8.5.5 General Interp

The **General Interp** selection is used to interpolate the solution from the non-selected set of zones in the current sequence to the selected set of zones (those zones which are in the table). This button is useful when adding zones to or changing zones for a problem that has a solution. Suppose we want to add a finer grid in a region of interest. This utility allows us to interpolate the current solution from the current grids onto the newly added grid. To do so, we would select the new grid, and then select this option.

When using this option, a pop-up dialog will appear giving the user the following three options:

Structured Grids, High Memory, Overlap Cell Scheme This will perform the general interpolation but in a high-memory mode. All zones will be loaded into memory at one time, making the operation faster, but more memory intensive. This option is only valid for structured grids.

Structured Grids, Low Memory, Overlap Cell Scheme This will perform the general interpolation but in a low-memory mode. The new zones will be interpolated to one at a time, allowing less memory to be utilized. This option is only valid for structured grids.

All Grid Types, closest Point Scheme This option is valid for any grid and performs a general interpolation based on the closest overlapping cell center value.

Note: Note that the general interpolation is done in such a way that it destroys the current zonal boundary coefficients stored for the sequence, so the user must recompute these (if they are needed) after performing the interpolation. Zonal boundary coefficients are used with the Chimera (overlapping grids) feature.

8.5.6 Insert Solution

The **Insert Soln** button is used for adding (or associating) a physical model solution to the selected zones in the solution table. The physical model to be inserted is selected from the **Soln For** option menu.

Inserting a solution allows *GASPE* to know which physical models are valid for a given zone. A solution can be removed from a zone using the **Delete Soln** button and selecting the “Completely Remove this solution from all sequences” option.

If a zone already has the physical model solution associated with it, pressing the insert button will have no impact. The user can verify that a solution is added by looking at the **nSoln** column in the table. If the nSoln value is “N/A”, then the physical model is NOT associated with the zone.

8.5.7 Initialize Solution

A user may initialize the solution of one or more zones (and sequences) in one of several ways. The first way can be done here in the **Soln Mgt** tab using the **Initialize Soln** option. Using this, the solution for the selected **Soln For** physical model corresponding to the zone or zones displayed in the zone table (list of zones displayed to the right of the button) will be initialized. If the solution does not exist, then the **Initialize Soln** will create the solution and initialize the values according to the settings in the **Initialization** tab (see Sec. 8.2 on pg. 88). If a solution already exists, a pop-up menu will appear to allow the user to select

which part of the solution to initialize (specie densities, velocities, pressure, etc...). Using this feature, a user may initialize or re-initialize the solution for a specific physical model of a zone.

The second way a solution can be initialized is by the flow solver. When the flow solver is started, it first checks to see if a solution exists for the zones it will be computing on. If a solution does not exist in the solution file (*.sln*), the flow solver will initialize the solution. Due to this behavior, the user is never required to initialize the solution before starting the flow solver for the first time.

8.5.8 Delete Solns

The **Delete Solns** button is used for deleting the existing solutions for the selected **Soln For** physical model. This is most useful for deleting unneeded solutions, and thus reducing the solution file size.

When this option is selected, a dialog will appear with the following two options:

Delete Solution on current sequence This will free the solution from memory for the selected zones and sequence level. If the input deck is saved after deleting the solution, the solution is also deleted from the solution file. Running the solver will then re-initialize the solution in this case.

Completely Remove this solution from all sequences This will delete the solution from the selected zones on all the sequence levels. It will also prevent the selected physical model from being associated with the zone, thus not allowing the solver to use the physical model. This option is opposite to the **Insert Soln** command. Therefore, to add the solution for the physical model back, use the **Insert Soln** button.

8.5.9 Import Solns

The **Import Solns** button is used for importing new and/or replacing the existing solutions for the selected **Soln For** physical model. This is useful for initializing the solutions for specific zones with solutions from Plot3D, CGNS, or *GASPEX* *.sln* files.

When this option is selected, a dialog will appear allowing the user to select the import file format. For Plot3D format, the solution may be at either the nodes or cell centers. For cell centers, the solution file may consist of the interior cells only, or include the first layer of boundary cells. The solution variables should correspond to the primitive variables for the physical model selected.

8.6 Editing Surfaces

The **Edit Surface** tab allows the user to split or combine surfaces for structured grids. For unstructured grids, this tab is not applicable.

The edit operations are performed on the surfaces listed in the **Surface Description** table (located inside the **Edit Surface** tab). Surfaces selected from the tree view will be displayed in the table. So the user will need to select one or more surfaces in the tree view in order to split or combine surfaces. The user can select surfaces individually or an entire surface folder.

Any surface can be edited except for surfaces in the **Pt2Pt ZB** folder. Once a surface has been labeled as a point-2-point zonal boundary surface, that surface cannot be edited.

The total number of surfaces in the table will be given in the **# of Selected Surfaces** display box. This can be useful if the user wants to count the number of surfaces in a folder or all the surfaces. For example, if the user selects the “Surfaces” folder, the total number of surfaces will be given in the display box (as well as listed in the table).

8.6.1 Combine Surfaces

The **Combine Surfaces** button allows the user to combine surfaces that have been split. This operation will only operate on surfaces listed in the table (selected surfaces). The algorithm tries to combine surfaces in the table that belong to the same zone and same min/max category (*i.e.*,IMIN, IMAX, JMIN, JMAX, KMIN, KMAX). Surfaces that do not belong to the same zone cannot be combined.

The combine surface feature can only be performed on non-decomped surfaces. A user may have to recompose the zones if a decomposition exists (this is performed in the **Decomposition** tab of the run definition).

To combine surfaces, simply select the surfaces (two or more) and press the **Combine Surfaces** button. If the surfaces can be combined, the old surfaces are removed and a new surface will appear in the tree view representing the combined surfaces.

If more than one sequence level exists, the **Combine Surfaces** button will not work. A pop-up dialog will appear to inform the user that surfaces must be combined before creating sequence levels. In other words, all of the splitting and combining operations must occur before any additional sequences are created.

8.6.2 Splitting Surfaces

In order to fully understand the use of the splitting surface feature, we first need to understand the function and manipulations of surfaces within *GASPer*. When a new structured zone is imported into *GASPer*, the six basic surfaces (*i.e.*,IMIN, IMAX, JMIN, JMAX, KMIN, KMAX) are formed, and placed in the **Undefined** surface group folder. Typically the first manipulation of these surfaces occurs when the user applies the automated point-to-point zonal boundary algorithm from the **Zonal Boundaries** section. In this case, *GASPer* will split surfaces automatically, and form point-to-point zonal boundaries by moving these surfaces into newly formed point-to-point ZB folders inside the permanent **Pt2Pt ZB** surface group folder. The surfaces which have been created through splitting which are not point-to-point zonal boundaries will be placed in the **Undefined** surface group folder. The

remaining undefined surfaces are then typically moved into surface folders in the tree view so that properties such as boundary conditions, pointwise Q specification, and surface cutting can be applied to them.

Depending upon the grid topology, the six logical boundary surfaces for a given zone may or may not require splitting. A surface will require splitting whenever more than one boundary condition needs to be applied to that surface. For example, if half of a surface is described by a point-to-point zonal boundary, while the other half is described by a solid wall boundary condition, then that single surface will require splitting. The situation described here will be handled automatically by the automated point-to-point zonal boundary detection algorithm. However, there are other situations which will require manual splitting of the surfaces by the user. The two most common reasons for manually splitting a surface are:

1. The surface is a point-to-point zonal boundary that folds onto itself.
2. The user wants to apply different properties (e.g. boundary condition, pointwise Q specification, or surface cutting) to different portions of the current surface.

A more detailed description of these uses will be given in the next two sections along with a few specific examples, but first we will give a brief explanation of the function of the widgets in the **Edit Surface** tab, shown in Fig. 8.7 on pg. 102;

The first step in splitting a surface (or surfaces) is to select the surface (or surfaces) from the tree view. The selected surfaces will then be displayed in the table of the **Edit Surface** tab. The next step is to select the I1 and I2 grid line locations where the surface(s) are to be split. Notice that the name for every surface contains a surface type (IMIN, IMAX, etc) followed by two ranges (*e.g.*, 1-10, 1-1). The first range corresponds to the I1 splitting value, and the second range corresponds to the I2 splitting value. For an IMIN or IMAX surface, I1 corresponds to J, and I2 corresponds to K. For a JMIN or JMAX surface, I1 corresponds to K, and I2 corresponds to I. Finally, for a KMIN or KMAX surface, I1 corresponds to I, and I2 corresponds to J. The final step is to press the **Split Surface** button.

To help the user identify the split location, the I1 and I2 grid lines will be shown in the graphical display window. As the user changes the I1 or I2 index, the graphical split line will change as well.

If more than one sequence level exists, the **Split Surface** button will not work, thus preventing the use of the split surface feature. All of the splitting of surfaces (whether through the point-to-point automated zonal boundaries or the manual split surface feature) must occur before any additional sequences are created.

Point-to-Point Zonal Boundaries

As stated above, *GASpex* will split surfaces automatically to form zonal boundaries, but there is one special case when the user must split a surface manually in order to enable the automated zonal boundary detection feature for that surface. When a surface folds onto itself, the user will need to split the surface at the fold.

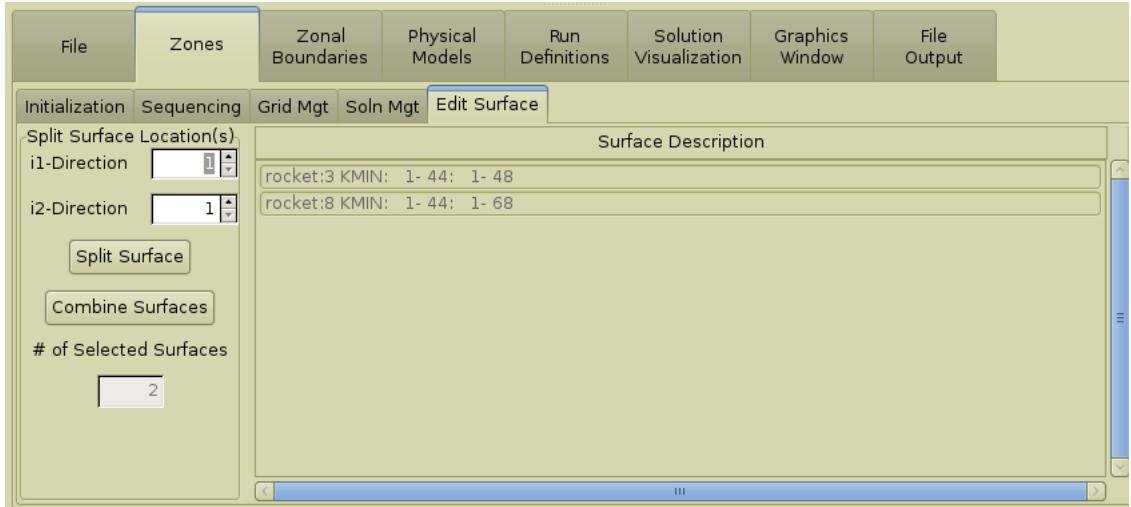


Figure 8.7: The Edit Surface tab after splitting a JMAX surface in half.

Varying Surface Properties

The second case is the more typical application of the split surface feature. Often times a surface will span a region in the domain that needs to describe more than one value for a given surface property. For example, a simple rectangular domain describing a flat plate with an injection hole will originally have a surface which spans both the plate and the injection hole. In order to apply a solid surface boundary condition to the plate, and an injection condition (*i.e.*, a Q Spec) to the injection hole, the original surface must be split.

An example of this occurs for the multi-zone Biconic problem, Fig. 8.8 on pg. 103, which has a normal injection hole near the top symmetry plane, in the last zone on the KMIN body surface (shown in red). The “grid:4 IMIN: 1-48: 1-60” surface has been selected from the tree view and is displayed in the split surface table, and is ready to be split.

The injection hole for this problem has the following cell face ranges: $J=1-4$, $K=29-32$. This corresponds to grid line ranges of: $J=1-5$, $K=29-33$. Note that the I1 and I2 split surface widgets are given in terms of grid line locations while the surface names are shown in terms of cell face locations. In order to create the desired injection hole, we will need to make two different surface splits. We will start by splitting the original surface into 4 surfaces by splitting along the $J=5$ and $K=29$ grid lines. After entering 5 for I1, and 29 for I2, and pressing the **Split Surface** button, the resulting 4 surfaces are shown in Fig. 8.9 on pg. 104.

In order to finish making the injection hole surface, we need to split the “grid:4 IMIN: 1-4: 29-60” surface along the $I=33$ grid line. After selecting this surface from the tree view, we set I1=1 and I2=33 (if the minimum value in a given direction is set, then no splitting will occur in that direction), and press the **Split Surface** button. The final result is shown in Fig. 8.10 on pg. 105 where the “grid:4 IMIN: 1-4: 29-32” surface can now be used to describe the injection hole boundary condition. The other 4 IMIN surfaces can easily be grouped into

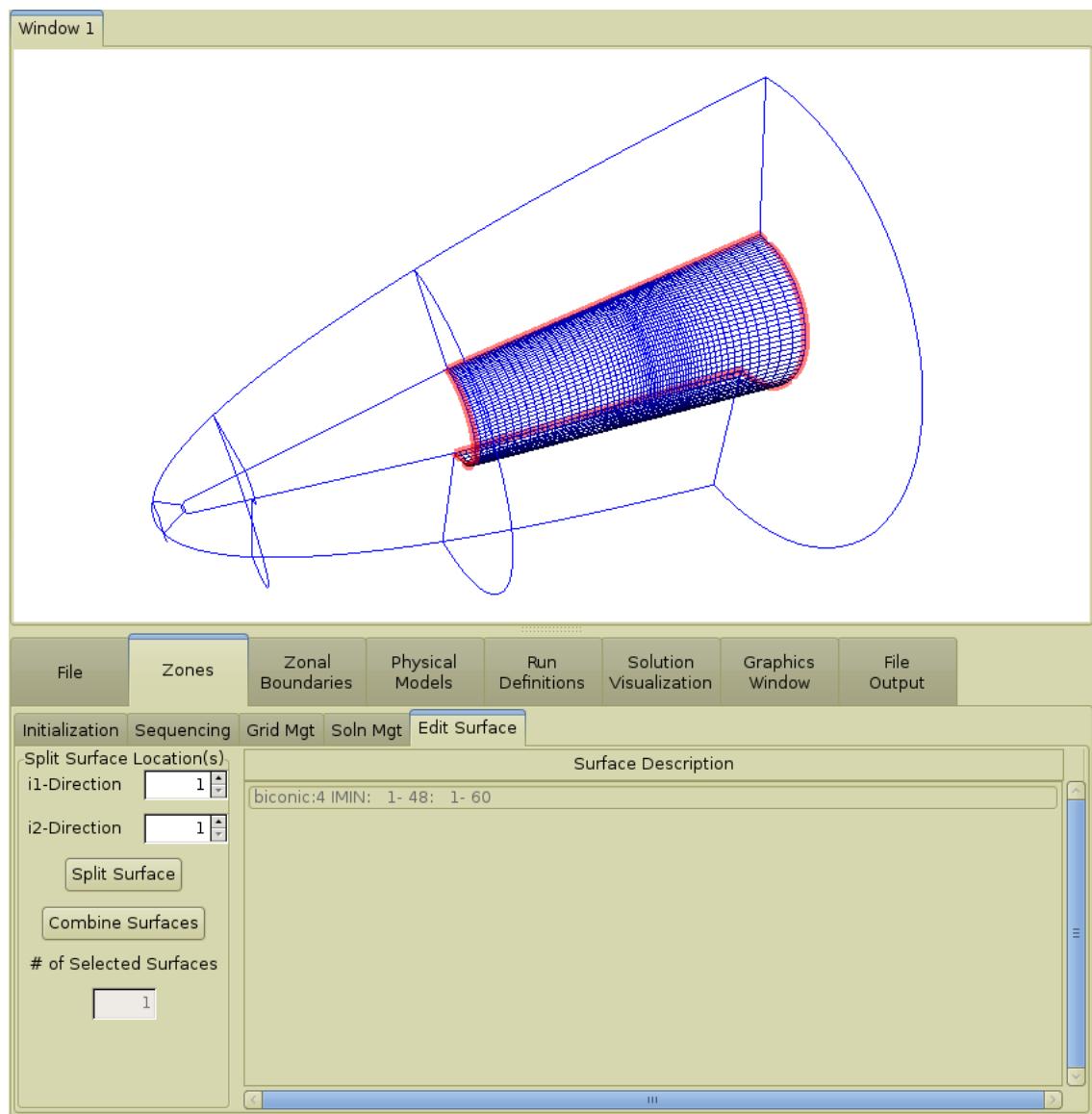


Figure 8.8: A multi-zone Biconic problem.

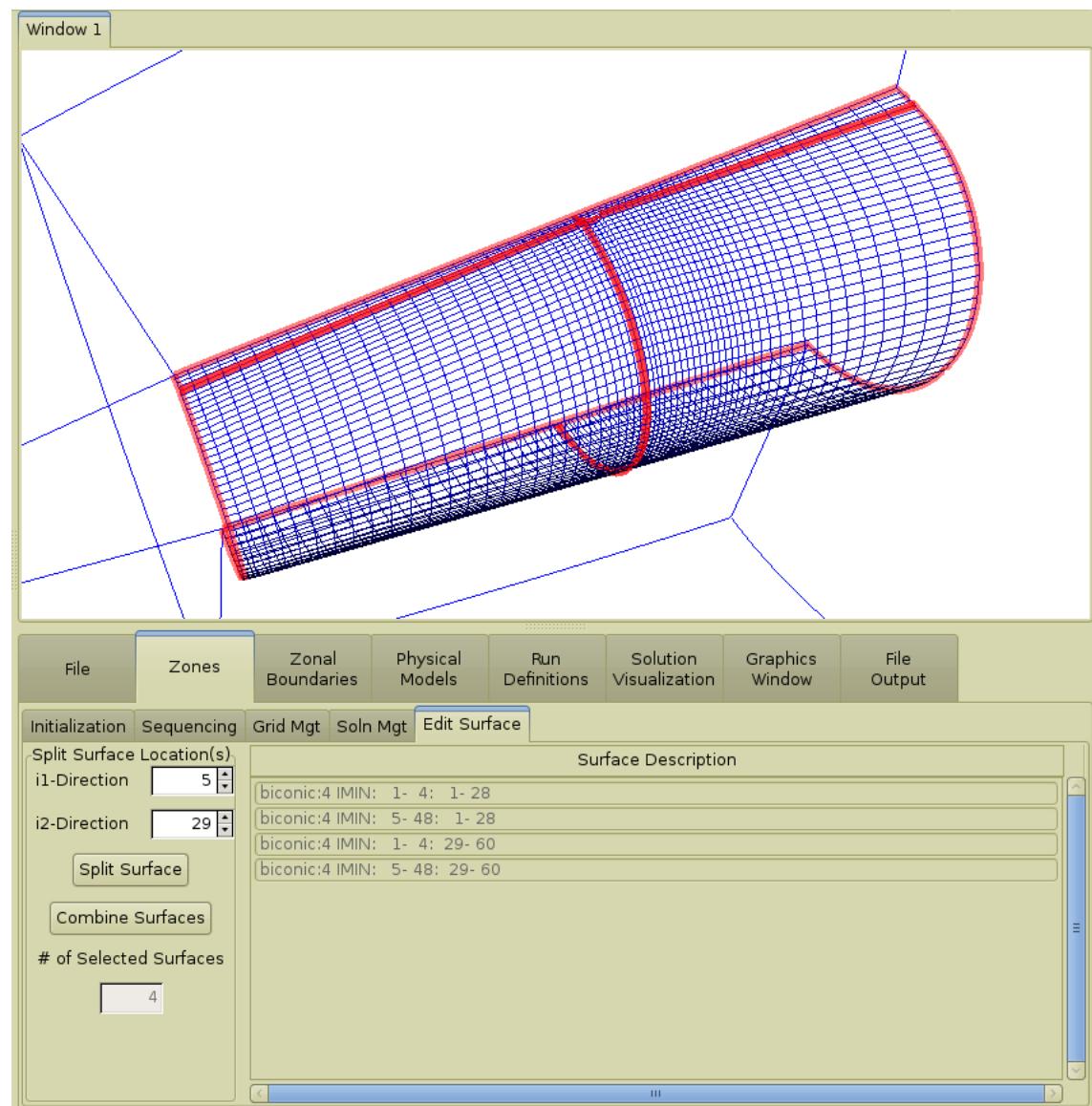


Figure 8.9: Original Biconic surface split into 4 surfaces.

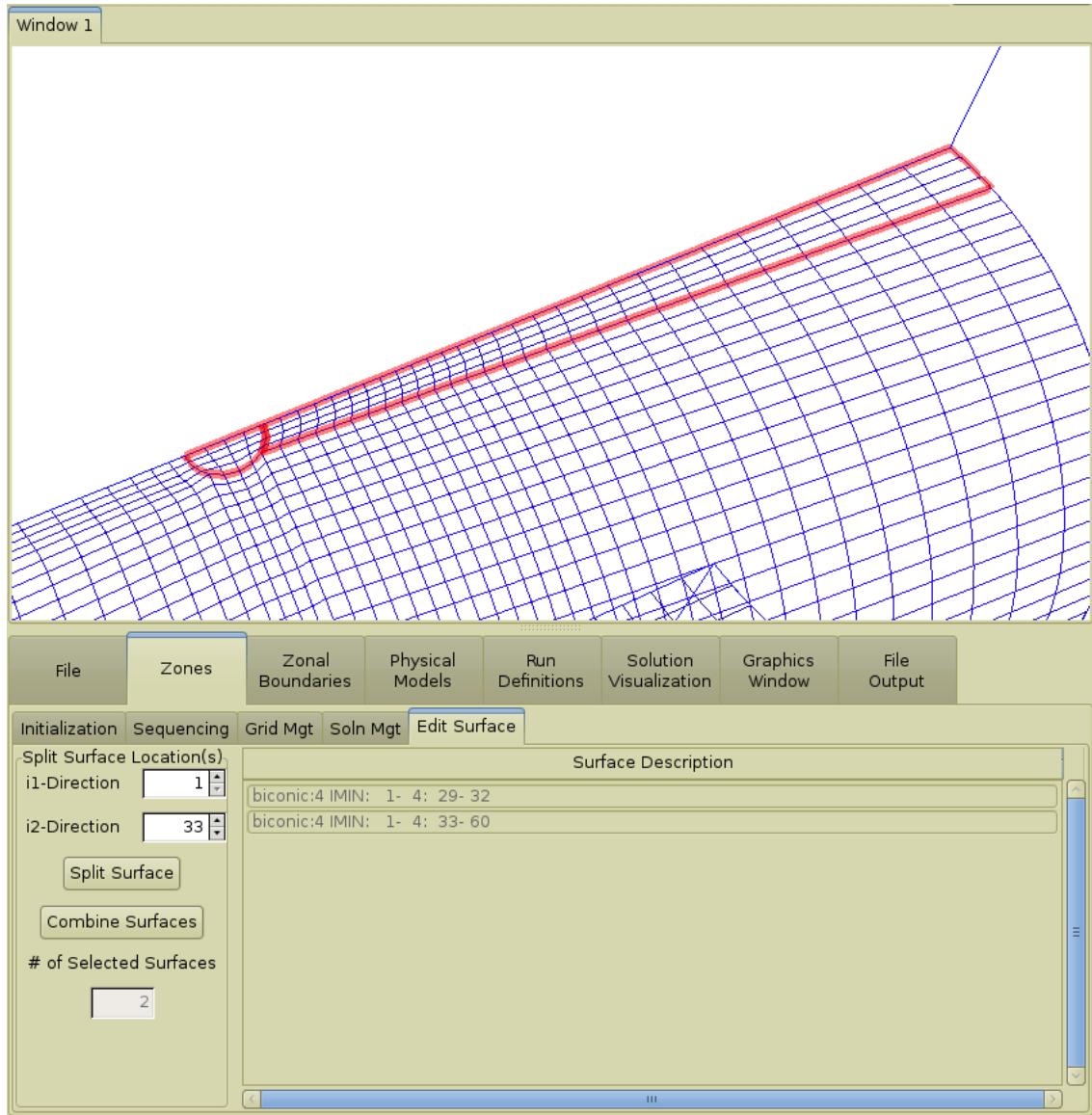


Figure 8.10: Biconic surface split to create injection hole.

a solid boundary folder where a single boundary condition can be applied.

The Biconic example used surface splitting for the purpose of applying separate boundary conditions. A similar application of the surface splitting is to create separate cutting surfaces. When performing automated hole cutting within the Chimera algorithm, surfaces must be properly grouped in order to ensure that hole cutting is done correctly. For collar grids (grids that are used to give a more accurate discretization of the intersection region between two solid surfaces - *e.g.*, wing and body) that have a single surface which spans the intersection between two separate bodies, the user will typically need to split the surface along the body intersection grid line. Then the two new surfaces can be placed into the appropriate cutting

folder (*e.g.*, one in the wing, and the other in the body).

8.7 Interp Coefs

Several solution interpolation options were discussed in the **Soln Mgt** section. These were all non-conservative algorithms that yield a first order interpolation. If the user needs a higher-order, conservative interpolation between two overlapping zones, then the interpolation algorithm described in Sec. 8.8.3 on pg. 115 should be used.

The higher-order conservative algorithm requires the volume of overlap to be computed. The overlap calculation is currently performed from within the GUI in the Zones **Interp Coefs** section.

To compute the interpolation overlap coefficients, press the **Compute Interps** button. This will bring up a volume of overlap creation wizard to assist the user in performing the following three steps.

1. Selection of the first set of zones
2. Selection of the second set of zones
3. Perform the volume of overlap calculation

The order of the zone sets is not important since the interpolation can be applied in either direction once the coefficients are computed. In other words, once the coefficients are computed, solution interpolation can be performed from the first set of zones to the second or vice versa.

The overlap coefficients are stored in memory until the input deck is saved. Upon saving the input deck, the interpolation data is written to the grid file for use by the solver. To remove the interpolation data from the grid file, press the **Delete Interps** button.

The overlap data can be verified by pressing the **Verify Interps** button. This will refresh the table information as well as show a dialog with the actual total volume of overlap along with the percent of overlap. If there are multiple overlap zones for a zone, the total volume of overlap will be cumulative.

One final comment should be made about computing overlap coefficients. The operation involves search algorithms and geometric calculations to determine the volume overlaps. For large grids this may take a considerable amount of time. In order to speed up the operation, zones should be decomped before computing the coefficients.

8.8 Zone Algorithms

The zone algorithm section performs a number of zone operations. These operations can be performed in the GUI or at the end of the flow solver. A brief description of the algorithms are given next, followed by more information in the sections that follow.

Grid Adaptation for Unstructured Soln This allows the user to adapt the grid for a zone or set of zones by adding cells based on solution parameters. This is only applicable to unstructured grids and must be performed from the GUI.

Shock Fitting This is an interface to the CFD Utilities package, OUTBOUND. This package, developed at NASA Ames, is intended to redistribute grid points in an optimal fashion for supersonic and hypersonic external flows. The utility will tailor the outer boundary to a shock wave and redistribute points extending from a surface in order to cluster grid points in a manner appropriate for resolving the viscous boundary layer. This is only applicable to structured grids.

Soln Interpolation Algorithm This performs a first or second order conservative solution interpolation from one set of zones to another.

Each algorithm operates on a zone family (folder) selected from the tree view. The default zone family is the top level Zones folder. A new zone family can be set by selecting a new zone folder and pressing the **Change To Selected** button.

If the zone algorithm is to be performed from the GUI, then the user can press the **Execute In Place** button. This will execute the selected algorithm. The algorithm may also be executed at the end of the solver (see Sec. 15.6 on pg. 260 for more information).

8.8.1 Grid Adaptation for Unstructured Soln

The grid adaptation allows the user to adapt the grid for an unstructured zone or set of zones by adding cells based on specified solution parameters. This option is not applicable for structured grids.

Duplicating Current Sequence

Grid adaptation can be performed on an existing sequence or on a newly created sequence. If the latter is desired, the first step when performing solution adaption is to use the **Dup** function in the **Sequences** list, as shown in Fig. 8.11 on pg. 108. This will create a copy of the sequence including the solution, and will place the new duplicate sequence above (e.g. a "finer" sequence) the current sequence in the list and will reset the current sequence to the duplicated one. Note that the duplicating algorithm only works on a non-decomped grid, so you will have to recomp the grid before performing this operation.

If the current sequence is decomped, the **Dup** sequence button will not work. A pop-up dialog will appear to inform the user that the grid must be recomped before duplicating a sequence. To summarize, solution adaptation can only be done on a non-decomped grid. After the entire adaptation process is completed, the newly adapted sequence can then be decomped.

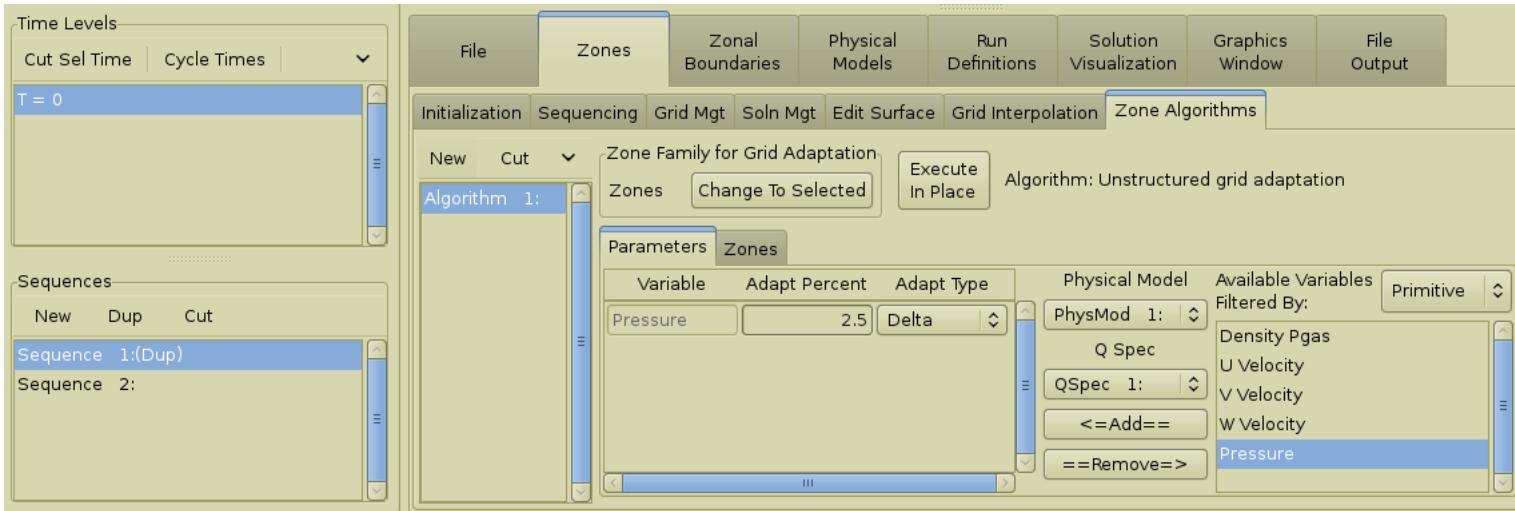


Figure 8.11: Duplicating the current sequence for grid adaption.

It is important to note that although the adapted grids are formed as sequences, unlike regular mesh sequencing, the adapted grids act independently from each other. Other than the initial copying of the solution during the duplicating phase, there currently is no communication allowed between the two "sequences" (i.e. no future interpolating up or averaging down of the solution). You can, however, view, output, solve, or perform any other typical action on each of the solutions independent of one another.

Creating a Grid Adaptation Algorithm

Using the **New** option under the zone algorithms section, select the **Grid Adaptation for Unstructured Soln** option. A grid adaption will consist of parameters that tell the algorithm which solution variables and the method of solution variable comparison to use to determine which edges in the grid to split (refine), and which set of zones and what type of cells in those zones to be refined.

The adaptation parameters are saved and can be used over and over to continually refine the grid. Consider a case where you are trying to capture pressure shocks and so you run the solver, adapt using the GUI, run the solver, adapt, etc. to adapt to pressure differences in the solution to refine the shocks. The adaption parameters can also be edited from one refinement to the next. For example, if you want to change which solution variables are used to perform the next refinement to capture a different part of the solution better, or if you start out refining all of the zones and then decide to further refine only a certain subset of the zones.

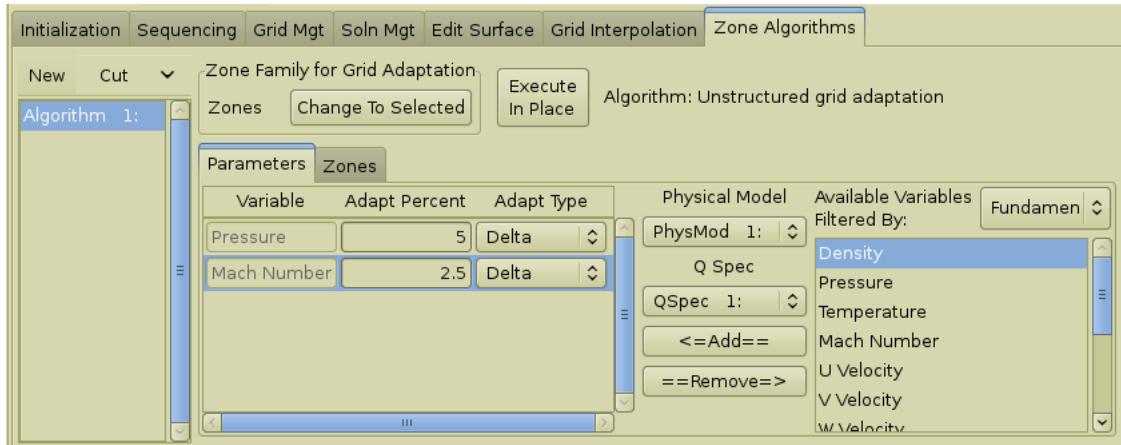


Figure 8.12: Grid adaptation Parameters tab

Grid Adaptation Select List

At least one grid adaption definition is required, but additional ones can be created or deleted using the zone algorithms select list. The select list is inside the **Zone Algorithms** tab and contains the following buttons: **New**, **Cut**, **Copy**, and **Paste**. Below these buttons is a list which contains all the zone algorithms.

Each grid adaptation algorithm can be viewed in the GUI by selecting the entry in the list. Typically a single adaptation definition is used to refine the grid, but you can create more than one to do things like adapt different zones using different solution parameters, etc. The grid adaptation algorithm is relatively complex, so it is suggested that you stick to a single, simple grid adaptation algorithm starting out.

To create a new adaptation algorithm, use the **New** option. Both the **Copy** and **Cut** buttons create a duplicate of the data in the clip-board. The **Cut** button will also remove the entry from the select list. The last entry in the clip-board can be recovered using the **Paste** button.

Parameters Tab

The **Parameters** tab is shown in Fig. 8.12 on pg. 109. After creating a new adaptation algorithm in the list, the next step is to add the solution variables that you want to use in the adaptation algorithm. The widget for adding and removing solution variables is very similar to the solution property widgets used in the visualization and output. Select one or more variables from the **Available Variables** list on the right, and then press the **Add** button to add them to the adaptation variable list on the left. You can remove them by selecting them in the adaptation variable list and pressing the **Remove** button. When you add a new variable, it is initialized with default values for **Adapt Percentage** and **Adapt Type**.

The **Adapt Percent** is the threshold percentage used for determining which cell edges to cut. The **Adapt Type** specifies which form of the solution variable to compare, either the

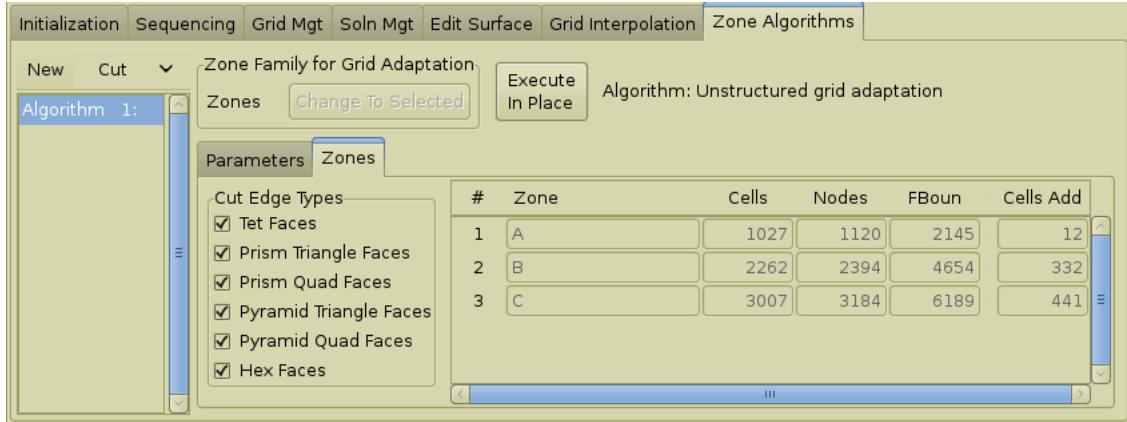


Figure 8.13: Grid adaptation Zones tab

Delta or the Gradient. The gradient is the delta of the variable divided by the edge length. When the grid adaptation algorithm is run, the delta or gradient of the solution variable is calculated for each edge in the grid, and then the edges that have values in the top **Adapt Percent** are selected to be divided. So the larger the value for **Adapt Percentage**, the more new cells will be created. In addition, the more variables that are chosen, the more new cells will be created because the edges are marked to be cut separately for each variable with its **Adapt Percentage** value. If an edge is marked for more than one variable, it will only be split one time.

Zones Tab

After adding one or more solution variables for adaptation, the next step is to select the edge types in the **Zones** tab. This tab is shown in Fig. 8.13 on pg. 110.

Select which type of cell faces in the grid that are allowed to have their edges divided. The default is to allow all types of cells to have their edges cut. However, the current algorithm only allows quad edges to be cut in matching pairs, and the cutting therefore propagates through an entire line of cells (i.e. what would be an equivalent to adding a grid line in a structured mesh in one direction). Therefore, in certain cases it may be wise to consider turning off the cutting of some of the quad faces.

When the algorithm is executed, the number of cells added for each zone can be seen in the zone list in the far right column. The current solution is interpolated onto the new grid where all new cells simply assume the values of the parent cell from which they came. It is possible to execute the algorithm a second time which would result in the adaptation running again and further adding cells to the grid, but it would be doing so on a grid with a solution that is no more refined than the original one, and therefore it is not a good idea.

Much like mesh sequencing, after adapting, the solution tends to converge rapidly at first on the adapted grid and then slows down to a regular pace when it reaches the level of convergence on the previous mesh. It is therefore recommended in general to converge your

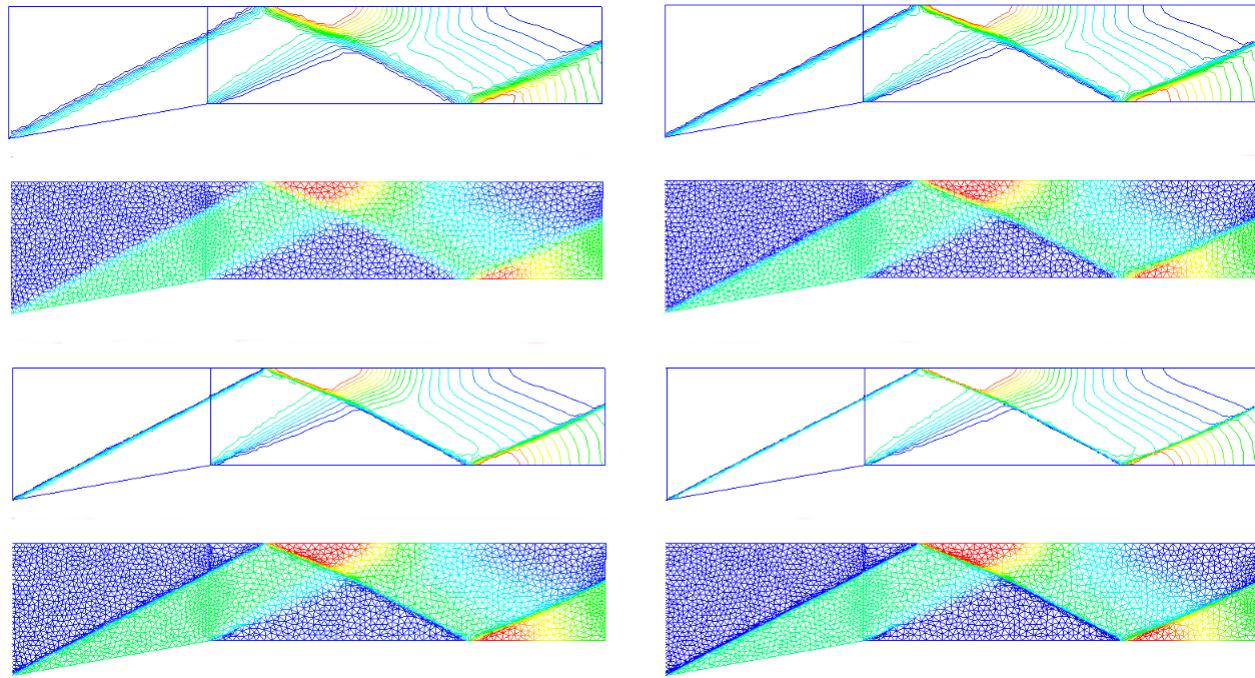


Figure 8.14: Solution adaptation on a simple 2-D ramp case.

original mesh well before beginning to adapt the grid to the solution.

Example of a Solution Adaptation

An example case of a simple 2-D ramp with a reflected shock using the solution adaptation algorithm is shown in Fig. 8.14 on pg. 111. The grid was adapted three times using both the pressure and mach number as solution adaptation variables each with a **Adapt Percentage** value of 5 and an **Adapt Type** of **Delta**. In the figure, a set is given for each run in which the pressure contours and mesh are displayed. The top left set shows the original grid and solution. The first grid adaptation is shown in the upper right, followed by the lower left and finally the lower right.

8.8.2 Shock Fit Algorithm

This is an interface to the CFD Utilities package, OUTBOUND. This package, developed at NASA Ames, is intended to redistribute grid points in an optimal fashion for supersonic and hypersonic external flows. Note that this algorithm is only appropriate for application with the Navier Stokes solver. The utility will tailor the outer boundary to a shock wave and redistribute points extending from a surface in order to cluster grid points in a manner appropriate for resolving the viscous boundary layer. Because routines being used for mesh improvement are external to AeroSoft, their use is subject to restrictions of third party

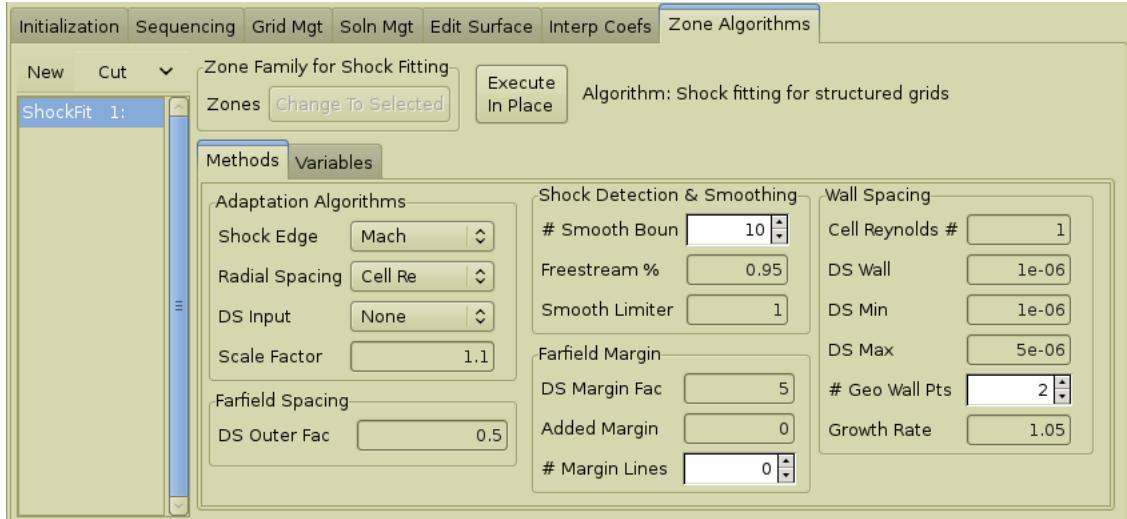


Figure 8.15: The **Shock Fitting** algorithm

software. In particular, this feature is intended for use with multi-zone, body fitted structured grids. The topology of the grids should be such that each zone in the domain should intersect the body surface with a single logical plane. Topologies which could be generated with a hyperbolic marching grid generation algorithm are perfect candidates.

Note the that OUTBOUND algorithm operates on grid lines emanating from a viscous surface. Shock detection and grid clustering is performed independently on each normal grid line. Smoothing is performed on the outer boundary as well as on the cell spacing at the wall in order to achieve smooth results.

Creating a Shock Fit Algorithm

Using the **New** option under the zone algorithms section, select the **Shock Fit Algorithm** option. Parameters for the Shock Fitting algorithm are stored as a Shock-Fit instances, where instances are displayed in a select list inside the Zone Algorithms tab. A Shock-Fit instance will consist of parameters that identify the algorithms, tolerances and parameters associated with a given execution of the routine. The Shock-Fit instance can then be executed either from the GUI, or during solver execution.

A new shock fit algorithm will appear as shown in Fig. 8.15 on pg. 112. The zones to apply the shock fit operation to should be specified in the **Zone Family for Shock Fitting** frame. To do this, organize the shock fit zones into a zone folder in the tree view. Select the folder and then press the **Change to Selected** button.

Shock Fit Select List

At least one shock fitting algorithm is required for shock adaptation, but additional ones can be created or deleted using the Zone Algorithms select list. The select list (located inside the

Zones Algorithms tab) contains the following buttons: **New**, **Cut**, **Copy**, and **Paste**. Below these buttons is a list which contains all the algorithms created by the user.

To create a new shock fitting definition, press the **New** button and select the **Shock Fit Algorithm** option. Both the **Copy** and **Cut** buttons create a duplicate of the data in the clip-board. The **Cut** button will also remove the entry from the select list. The last entry either cut or copied into the clip-board can be recovered by pressing the **Paste** button.

Methods tab: Adaptation Algorithms

Within the **Methods** tab are located all of the flags and parameters associated with the OUTBOUND utility. Let's start with the parameters inside the **Adaptation Algorithms** frame.

The first step in the mesh improvement process is to identify the outer location of the bow shock. There are a number of algorithms which may be used for this purpose, and are controlled by the **Shock Edge** enumeration whose values are listed as follows:

Scale Grid Simply adjust the arc length by the factor **Scale Factor**.

None Do not alter the outer boundary.

Mach (default) Move the edge of the outer boundary to the shock location identified as the point along a normal grid line where the Mach number first drops below the freestream value.

Press/Dens/T Mimic the shock detection algorithm found in the LAURA code which uses strong increases in pressure, density or temperature.

Gradient Use the magnitude of the gradient, in an algorithm similar to the SAGE code.

Once the outer boundary has been identified, the grid points along each normal arc are redistributed according to the algorithm identified by **Radial Spacing**. Note that this option selects the method for calculating the grid spacing for the first point off of the viscous wall. The options are:

Cell Re (default) Set the arc length to the first point off the wall to a value that will match the local cell Reynolds number to the parameter **Cell Reynolds #**.

Wall DS Set the arc length of the first point off the wall to a single value identified by the parameter **DS Wall**.

Outer Only Keep the value of the spacing to the first point off the wall unchanged.

The **DS Input** enumeration is only relevant when the **Radial Spacing** enumeration is set to **Wall DS**. When this condition is met, the enumeration takes on the following values:

Wall DS Set the arc length of the first point off the wall to a single value identified by the parameter **DS Wall**.

Arc Length Apply the value of **DS Wall** as a percentage of arc length.

Uniform DS Apply a uniform spacing across the entire arc length

Scale Factor is a multiple used to uniformly alter the arc length when the **Shock Edge** flag is set to **Scale Grid**.

Methods tab: Shock Detection & Smoothing

For the **Shock Detection & Smoothing** frame, there three inputs. **# Smooth Boun** is the number of iterations to be performed globally in order to smooth the outer boundary once shock detection has been performed. The **Freestream %** input is used in different shock detection algorithms. Finally, the **Smooth Limiter** is a multiple of the local spacing used to limit the amount of smoothing performed on the outer boundary.

Methods tab: Farfield Margin

Once the bow shock is located, a layer of grid points upstream of the shock can be added. The parameters related to the farfield margin are now discussed.

DS Margin Fac represents the size of the outer margin as a multiple of the current grid spacing at the shock location.

Added Margin represents an addition to the boundary margin as a multiple of the grid spacing at the current outer boundary.

Note that **DS Margin Fac** and **Added Margin** are additive, and somewhat redundant.

Smooth Limiter is a multiple of the local spacing used to limit the amount of smoothing performed on the outer boundary.

Once a new farfield boundary is computed, the adaptation algorithm will then attempt to redistribute points along the grid lines emanating from the surface to the the farfield. Unless uniform spacing is requested, a blended combination of geometric growth and the hyperbolic tangent function are used to smoothly adjust the grid spacing from the requested wall and farfield values.

Methods tab: Wall Spacing

When the **Radial Spacing** method is **Cell Re**, wall spacing is computed based upon **Cell Re** according to $Re_{cell} = \rho a \Delta s / \mu$. Wall spacing values are limited according to **DS Min** and **DS Max** and then globally smoothed.

If the **Radial Spacing** method is **Wall DS**, then **DS Wall** is the value for directly specifying constant wall spacing. If the **Radial Spacing** method is **Arc Length**, then **DS Wall** is the value of constant wall spacing expressed as a percentage of total arc length.

The values **DS Min** and **DS Max** are used to limit the wall spacing values computed by any method other than direct specification. Positive values correspond to actual limits on the spacing, while negative values are used to place limits based upon a percentage of the total arc length.

Geo Wall Points represents the number of points away from the wall which should be distributed according to a geometric growth function. If this value is less than 3, then no geometric growth function is used. The geometric growth function will typically yield a larger clustering of points near the wall, and can be used to ensure adequate points in a boundary layer.

The **Growth Rate** represents the growth rate of the normal spacing away from the wall when **# Geo Wall Points** is greater than 2.

Methods tab: Farfield Spacing

The magnitude of the grid spacing toward the wall at the farfield boundary is controlled by **DS Outer Fac**. This value represents the outer spacing as a fraction of the spacing computed using a one-sided hyperbolic tangent with the wall spacing specified according to the current redistribution parameters. For example, if **DS Outer Fac** is set to 0.5, then the outer spacing will be set to half the value calculated using a one-sided hyperbolic tangent function. Once the outer spacing is computed, all points along the arc will be computed based upon a blending of the geometric growth away from the wall followed by a two-sided hyperbolic tangent function. If **DS Outer Fac** is set to 1, then a one-sided hyperbolic tangent function is used.

The Variables tab

This tab is used to select the value used for the shock location, when such a variable is required. Note that a suitable variable such as Pressure, Temperature or Density must be inserted to the active variables list in a fashion similar to that of the VTK **Variables** property when **Shock Edge** method is either **Press/Dens/T** or **Gradient**. Other variables will be added automatically.

8.8.3 Solution Interpolation

The conservative solution interpolation algorithm is shown in Fig. 8.16 on pg. 116. This algorithm is used to interpolate a solution from one set of zones to another in a conservative manner.

Setting Up Zone Families

The zones that will the solution interpolated to (*i.e.*, receiver zones) are specified in the **Zone Family: Soln Receiver** frame. The zones must be grouped into a separate folder in the tree view. Once this is done, select the folder and then press the **Change to Selected** button.

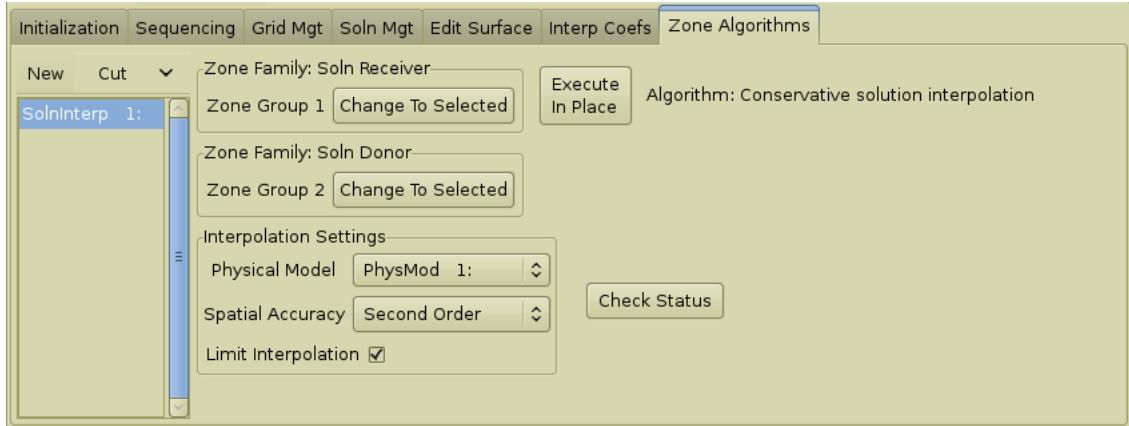


Figure 8.16: The **Soln Interpolation** algorithm

In a similar manner as the receiver zones, the donor zones must also be specified in the **Zone Family: Soln Donor**.

Interpolation Coefficients

Before the interpolation can be performed, the overlap coefficients between the donor and receiver zones must be computed. This operation is performed in the **Interp Coefs** section (located next to the Zone Algorithms tab). See Sec. 8.7 on pg. 106 for more details.

Interpolation Settings

There are only a few interpolation settings that the user needs to set. The first is the physical model associated with the interpolation. Recall that the physical model dictates the solution associated with a zone. In this case, the **Physical Model** selection specifies the solution parameters that will be interpolated from one set of zones to another.

The interpolation uses a conservative algorithm that can have first or second order spatial accuracy. The **Spatial Accuracy** input controls whether first or second order is used. If the interpolation is second order, there is the possibility of numerical “overshoots” where the interpolated value is outside the local range of values. To prevent overshoots, the user can select the **Limit Interpolation** option. This applies a limiter that will adjust the gradient used in the reconstruction to prevent new local minimum or maximum values. The limit interpolation option is still conservative.

A **Check Status** option allows the user to verify the data. If overlap coefficient data is missing or if the physical model is not compatible with the zone families, the check status option will alert the user.

Chapter 9

The Zonal Boundaries Frame

9.1 Introduction

The Zonal Boundaries frame is comprised of 5 tabs which allow the user to manipulate and set parameters for the zonal boundaries and overlapping grids. The data in the tabs is tied directly to the current time level and sequence level selected in the **Time Level** and **Sequences** lists, Fig. 8.1 on pg. 87. Several tabs are also dependent upon which zone nodes are selected in the tree view.

9.2 Zonal Boundaries Tabs

Most of the zonal boundary treatment in *GASpex* is highly automated and requires very little user input. The majority of that input takes place from within the **Zonal Boundaries** tab. In addition to the now-automated point-to-point zonal boundary mapping, *GASpex* can also detect singular lines and/or points, and perform automated hole cutting and computation of chimera zonal boundary interpolation coefficients for overlapping grids.

Here is a brief description of each of the five Zonal Boundaries tabs:

Pt-2-Pt ZB Perform manual or automated zonal boundary operations on all zones. Both singular line and point-to-point zonal boundaries can be automatically detected and set up from this tab. Manual creation of point-to-point zonal boundaries is also available if needed.

Cut Holes Used to perform Chimera overlapping-grid hole cutting. This assigns the “cut” cell type to distinguish them from “discrete” cells.

Chimera Coefs Used to compute Chimera overlapping-grid coefficients. Coefficients are necessary to transfer the solution data from one overlap grid to another (updates the “interp” cells).

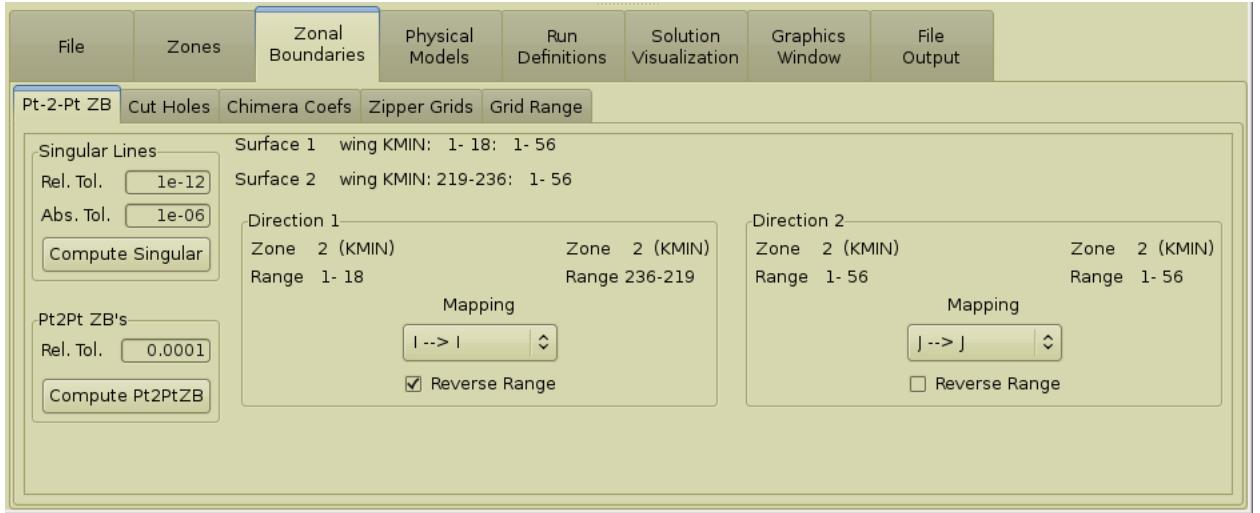


Figure 9.1: The Pt-2-Pt ZB tab used to create and edit zonal boundary information.

Zipper Grids Used to create and set up zipper grids, an aid in visualization and integration of overlapping physical surfaces.

Grid Range Settings for grid ranges used with hole cutting and chimera overrides.

The following sections will describe in detail the function of each of these tabs.

9.3 Point-to-Point Zonal Boundaries

The Pt-2-Pt ZB tab inside the Zonal Boundaries frame is used for creating, displaying and setting point-to-point zonal boundary information. A user may create a point-to-point zonal boundary either manually or by using the *GASPer* auto detect feature. In most cases, the user can use the auto detect feature for both point-to-point zonal boundaries and singular lines. This helps reduce the amount of work and time that is required by the user in setting up a problem. An example of information being displayed in the Pt-2-Pt ZB tab is shown in Fig. 9.1 on pg. 118.

9.3.1 Computing Singular Lines

Singular surfaces are those which collapse to a singular line or point. The user can choose to manually group singular surfaces (a surface which describe a singular line or point) into a surface folder, but there is a simpler way. To have *GASPer* automatically move all the singular surfaces currently located in the “Undefined” folder into a “Singular” folder, the user can press the **Compute Singular** button, Fig. 9.1 on pg. 118. Note that the “Singular” surface folder is created when singular surfaces are detected.

The relative and absolute tolerances are used to determine what is considered a collapsed point for the singular algorithm. The absolute tolerance is the physical distance in dimensional space between two points being checked. The relative tolerance is the absolute tolerance normalized by the physical size of the zone that contains the two points being compared.

In the case of automated singular line detection, the value of the absolute tolerance is typically the easier criteria for the user to manipulate and use appropriately. So if the default values don't work, then first try to change the absolute value before attempting to alter the relative value. There are typically two types of "failures" in the algorithm. The first kind is uneventful, and that is simply that a singular surface goes undetected all together. The second kind can get a bit messy. Sometimes the default tolerances result in either non-singular surfaces being considered as singular, or only parts of a singular surface are detected instead of the entire surface. In either case, the end result is usually an excessive number of undesirable surfaces since the algorithm will split surfaces in order to create the computed singular surfaces. Unfortunately, there is not currently an undo feature for the singular surface detection algorithm, so it is very wise for the user to save the input deck before pressing the **Compute Singular** button.

Note: If more than one sequence level exists, the **Compute Singular** button will be disabled, thus preventing the use of this feature. All of the singular surfaces must be created before any additional sequences are created. Also, parallel domain decomposition must not have been performed. If zones are decomposed, the user must recomp the problem before attempting to detect singular lines or points.

9.3.2 Introduction to Zonal Boundaries

A zonal boundary is a physical boundary between two neighboring zones across which the solution variables pass. A point-to-point (pt-2-pt) zonal boundary is a zonal boundary where the grid points match one another. Or in other words, the faces of each zonal boundary have a one-to-one mapping with each other. If this were not the case, the zonal boundary would be called non-point-to-point.

Information about a point-to-point zonal boundary is displayed in this tab only when a single zonal boundary folder containing two structured grid surfaces is selected in the tree view. Zonal boundary folders reside in the "Zonal Boundaries" folder under the "Pt2Pt ZB" folder. The user will need to select an individual folder inside the "Pt2Pt ZB" folder in order to see the information for that zonal boundary displayed in the frame. Note that a folder is selected by clicking on the folder name.

Point-to-point zonal boundaries are found in almost every multi-zone problem. Inside of *GASpex*, they are created in one of three ways: automatically via the **Compute Pt2PtZB** button inside the **Zonal Boundaries** tab, by manual creation inside the **Pt-2Pt ZB** tab, or through the grid import mechanism when importing a supported file type. If zonal boundaries are created using the **Compute Pt2PtZB** button operation, or through direct

import, the information displayed in the **Pt-2Pt ZB** tab will be correct. The user will not need to change any of the settings. On the other hand, if the user creates a zonal boundary manually, then the settings inside the **Pt-2Pt ZB** tab will need to be verified and set correctly.

In most cases, the user will create zonal boundaries either using the auto compute feature of *GASPE*, or via direct import. The auto compute feature will find zones that have matching boundaries with other zones, and if needed, will split the boundary surface in order to create the point-to-point zonal boundary. There are some cases that the user must create and set up the zonal boundary manually. An example of this case is periodic boundary conditions when the zonal boundary is not physically connected. Creating zonal boundaries manually will be described later in this section.

For consistent high-order accuracy at zonal boundaries, each zone is surrounded by multiple layers of “ghost” cells. At zonal boundaries, interior cells from one zone are mapped into the ghost cells of the adjacent zone. In the case where each zone has a different number of equations being solved (thus a different number of flow variables), *GASPE* will match up the names of the flow variables and pass them correctly. Flow variables that do not have a match are not passed.

9.3.3 Automated Point-to-Point Zonal Boundaries

Similar to computing the singular surfaces, we can compute all the point-to-point zonal boundaries. *GASPE* has an internal algorithm that will search the grid for zonal boundaries. The algorithm searches for common nodes to within some tolerance. This relative tolerance, Fig. 9.1 on pg. 118, is specified by the user and may need to be adjusted in order to get the algorithm to work correctly for a given set of grids. For most problems, the default value for the relative tolerance is acceptable.

By pressing the **Compute Pt2PtZB** button in the **Zonal Boun** tab, Fig. 9.1 on pg. 118, the point-to-point zonal boundaries that exist between surfaces located in the “**Undefined**” folder will be computed and moved to the “**Pt2Pt ZB**” folder. The “**Pt2Pt ZB**” folder will contain a sub-folder for each matching point-to-point zonal boundary surface pair found. The name of each sub-folder will be the zone name and surface type for the two surfaces contained in the sub-folder separated by a colon. For example, a point-to-point zonal boundary found between the IMAX surface of zone 1, and the IMIN surface of zone 2 might look like: “grid:1 IMAX: grid:2 IMIN”. Surfaces will be split when necessary to create a point-to-point zonal boundary surface pair. The “left-over” surfaces will remain in the “**Undefined**” folder.

In order to determine if two surface faces form a point-to-point mapping, the grid points of each face are compared to see if they lie on top of each other. The relative tolerance is the physical distance between two points being checked and normalized by the cell size (longest edge length). Occasionally this is problematic with a highly stretched grid where the (aspect ratio)*(relative tolerance) > 1 .

There are typically two types of “failures” in the point-to-point zonal boundary algorithm. The first kind is reasonably uneventful, and that is simply that a point-to-point zonal boundary goes undetected altogether. The second kind is more problematic. Sometimes the default

tolerance results in either non-matching surfaces being considered as point-to-point, or only scattered parts of matching surfaces being detected instead of the entire matching region of the surfaces. In either case, the end result is usually an excessive number of undesirable surfaces since the algorithm will split surfaces in order to create the computed zonal boundary surfaces. Unfortunately, there is not currently an undo feature for the point-to-point zonal boundary detection algorithm, so it is very wise for the user to save their input deck before pressing the **Compute Pt2PtZB** button.

Note: If more than one sequence level exists, the **Compute Pt2PtZB** button will be disabled, thus preventing the use of this feature. All of the point-to-point zonal boundaries must be created before any additional sequences are created. Also, the surfaces must not be decomped. If zone's are decomped, the user must recomp the problem before attempting to detect point-to-point zonal boundaries.

9.3.4 Creating Zonal Boundaries Manually

We will now discuss manual creation of a point-to-point zonal boundary. The first step is to identify the two surfaces that will make up the zonal boundary. A zonal boundary is made up of only two surfaces. If the zonal boundary has more than two surfaces, the user will need to create more than one zonal boundary. If the zonal boundary to be created has a surface that defines more than just the zonal boundary, the user will need to split the surface first. The surfaces to be used for the zonal boundary must correspond to each other in a one-to-one mapping. The surfaces should not have extra faces that belong to other zonal boundaries or a boundary condition.

Once you have the two surfaces for the zonal boundary identified, the next step is to create the zonal boundary folder. To do this, go to the tree view and open up the **Pt2Pt ZB** folder. Create a new folder inside the **Pt2Pt ZB** by right mouse clicking on the “**Pt2Pt ZB**” text and selecting the **Edit→New Pt2Pt** selection, Fig. 9.2 on pg. 122. An empty **Pt2PtNode** folder should appear with the name “**Untitled**”.

With the new folder empty, move the two surfaces associated with the zonal boundary into the new folder. You will only be able to move two surfaces into the folder. Once there are two surfaces inside the folder, you will not be allowed to move any more surfaces into the folder. The surfaces must be moved one at a time into the folder. *GASPer* will not allow you to move both surfaces into the folder at the same time. Once the first surface is moved into the folder, the folder starts to take on properties. The number of faces in each direction on the boundary is recorded as soon as the first surface is placed into the folder. With this information, *GASPer* requires that the second surface be consistent with the first surface. The number of faces in each direction must be consistent. The orientation can be different, but the total number of faces must be the same for each surface.

If you have moved one surface into the **Pt2PtNode** folder and noticed that *GASPer* will not allow a second surface to be placed into the folder, then make sure that only one surface resides in the folder and that both surfaces have the same number of faces. Another possi-

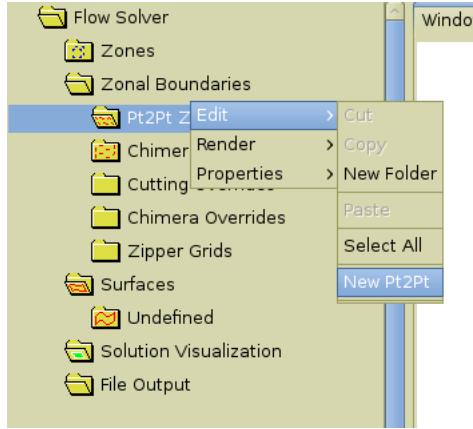


Figure 9.2: Manually creating a new zonal boundary folder via the “New Pt2Pt” menu item.

bility is that the folder was previously setup with two surfaces and then one was removed. If this is the case, then only the original two surfaces are allowed back into the folder.

If you wish to remove the zonal boundary, then select the **Edit→Cut** menu item. This will move both surfaces to the “**Undefined**” surface folder, and erase all zonal boundary information associated with them.

Once you have created a zonal boundary folder and moved surfaces into it, the remaining task is to set the zonal boundary information located in the **Pt-2-Pt ZB** tab. This will be discussed in the next section.

Note: Zonal boundaries should only be created and deleted (using the Cut option on the pop-up menu) under the following conditions: there is only one sequence level and the sequence is not decomped. When multiple sequences exist or zones have been decomped, zonal boundaries should not be created or removed inside the tree view.

9.3.5 Setting Zonal Boundary Parameters

The **Pt-2-Pt ZB** tab allows you to set several parameters associated with a point-to-point zonal boundary. Changing parameters in this frame is normally done when you have created a zonal boundary manually. Zonal boundaries created with the **Compute Pt2PtZB** button will create zonal boundaries automatically and will have the correct settings in this frame.

When creating zonal boundaries manually, the user needs to verify two things: the mapping coordinates and the range direction. The mapping coordinates are set using the mapping option menu. Every surface of a zonal boundary will have two coordinate directions. The coordinates will come in pairs of either (I,J), (J,K), or (K,I). The coordinates of one surface must map to the coordinates of the other surface. Note that each surface in the zonal boundary pair may have a coordinate pair different from the other.

Since there are two coordinate directions, the frame displays a direction 1 and a direction 2. A mapping option menu is available in each of the direction boxes, and lists the two

choices available for the mapping. The mapping option menus are tied together such that changing one will change the other. So the user only needs to set the mapping for one direction, which will in turn take care of the mapping for the other direction.

Besides the mapping direction, the user must specify whether the coordinate direction maps in reverse. In other words, does the face direction need to be reversed in order to make the mapping be correct. To determine this, look at the location of the first face on each surface (the lowest value in the range). Do the faces match or does the lowest range value of one coordinate match the largest range value of the other coordinate? If the latter is true, then the **Reverse Range** toggle needs to be selected.

The sample shown in Fig. 9.1 on pg. 118 may help clarify some of the parameters discussed. The first surface in the zonal boundary belongs to zone 1 and is part of the $k = 1$ surface. This surface therefore has an (I,J) pair that needs to be mapped to the second surface. The second surface comes from zone 4 and is part of the maximum I surface. It has (J,K) as the coordinates that define the surface. From the figure, the I direction of surface 1 maps to the K coordinate of surface 2. Likewise, the J coordinate of surface 1 maps to the J coordinate of surface 2. Because the faces in each direction are different, the option menu has only one valid choice (the other option is non-selectable).

If we continue looking at the figure, we see that direction 1 has the range direction reversed. This means that, for direction 1, the first index of surface 1 ($I = 1$) maps to the last index of surface 2 ($K = 20$). The mapping will continue in this fashion and simply means that the coordinates run in the opposite direction. This can often happen with meshes containing a large number of zones.

9.3.6 Checking Zonal Boundaries

After a zonal boundary has been created, the user may check the maximum tolerance of the point-to-point face values. This is done using the **Check Pt2PtZB** option. The difference between each point-to-point face center is computed and the maximum value is reported. This enables the user to double check the settings for the zonal boundary and confirm tolerances.

9.4 The Chimera Algorithm

The use of Chimera allows the user to create grids or zones that overlap one another. The flow solver computes a solution on both grids simultaneously. Information from each overlapping grid is passed back and forth so that the governing equations are solved on each mesh.

An example of where this technology would be used is in wing/pylon problems. An existing grid may be available for the wing. After computing the solution on the wing, the user may wish to add a pylon under the wing. With traditional CFD solvers, the user would have to make a new grid that modeled the wing and pylon geometry together. To do this would be tedious, as well as time consuming. With Chimera, the user would create a new grid just for the pylon. The new grid would not have to extend out very far from the pylon

and would have a simple grid topology. The flow solver, with the use of Chimera technology, would then solve for the wing/pylon solution using both grids. Information from the original grid would be passed to the pylon grid and vice versa. The practical use of Chimera is endless, and *GAS*Pex provides a straight forward and easy way to use this technology.

9.4.1 The Importance of Surface Groupings

Before performing any of the Chimera-specific tasks (*i.e.*, hole cutting, computing interpolation coefficients) in the **Zonal Boundaries** tab, we want to place all of the surfaces into the appropriate surface group folders in the tree view. Placement of the surfaces will affect both the hole cutting, and the computing of interpolation coefficients. Specifics of how the surfaces should be placed will be discussed in the following sections.

9.4.2 Hole Cutting

The first step in preparing *GAS*Pex to run properly on a Chimera grid system is to perform the hole cutting. Hole cutting does not need to be performed on non-overlapping grid systems. However, for problems where some zones overlap and some do not, the hole cutting will be performed over all the grids, and those which are unaffected will not be cut. The purpose of the hole cutting algorithm is to determine which cells lie inside the user-defined solid surfaces, and then to blank out these cells (*i.e.*, cut a hole in the grid) so that they will not be part of the solution. A two-layer set of “fringe” (interpolation) cells is then created around each hole to allow the passing of solution information to the region around the hole (*i.e.*, they act as a local zonal boundary condition around the hole). The interpolation of cells surrounding a hole is discussed more in the zonal boundary coefficient section.

The best way to explain much of the hole cutting is through example, so we will be referring to a simple example of a wing/body problem to help describe the details of the hole cutting. By wing/body problem, we mean a problem with a body grid, a wing grid, and a collar grid. When generating such a grid system using the Chimera philosophy, the body grid is typically completely solid (*i.e.*, no area cut out for the wing intersection region), the wing grid extends inside the body surface, and the collar grid is used to accurately describe the intersection between the body and wing. So the end result of the hole cutting will be for the wing to cut a hole for the wing in the body grid, and for the body grid to cut out the section of wing inside the body. No holes will be cut in the collar grid since it helps to describe the intersection of both the body and the wing and does not have any cells inside either solid surface. When solution fidelity near surface intersections is important, collar grids which describe that intersection are a necessity. They help prevent globally blanked out regions near the intersection where the two intersecting solid surface grids (in this case, the body and wing grids) will have hole cells.

The most important step in setting up the hole cutting is to specify which surfaces are used to define the physical (solid-surface) boundaries. Considering our wing/body example, we need to cut any cells that lie inside the wing surface or the body surface. Specifying

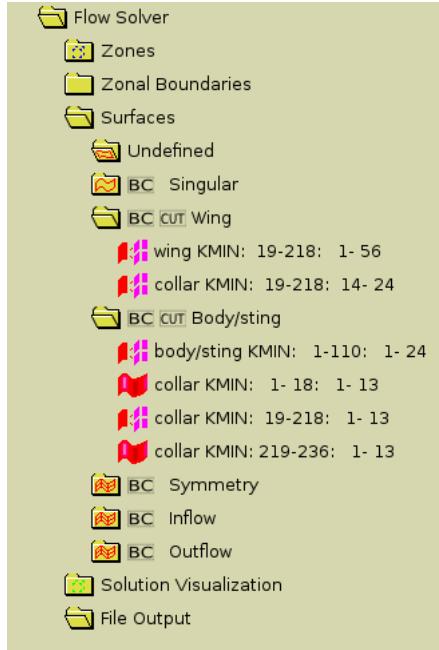


Figure 9.3: The tree view with the **Cut** flag enabled for the wing and body surfaces.

the wing and body surfaces as the physical boundaries for hole cutting is done by enabling the **Cut** flag that is located to the right of the surface group folder icon in the tree view. To enable the **Cut** flag, select the targeted surface node, and press the right mouse button to display a pop-up menu. From the pop-up menu select **Toggle Cutting Surface On/Off**. The tree view for this example should appear as in Fig. 9.3 on pg. 125. For a non-Chimera problem, the surfaces which describe the body and wing might typically be in a single surface group folder called “**Walls**”. However, for a Chimera case, we have to be much more careful about where we place the solid surfaces.

An important point about how to distribute your cutting surfaces into folders with the **Cut** flag enabled needs to be made. Notice for this case that we placed the surfaces for the body and the surfaces for the wing in separate cutting folders. This tells the cutting algorithm that the body grid is allowed to be cut by the wing surface, and vice-versa. If both sets of cutting surfaces had been placed in the same cutting folder, then the body grid would not be cut by the wing surfaces, nor would the wing grid be cut by the body surfaces. Basically, the rule is that any zone associated with any one of the surfaces in a given cutting folder will not be cut by any of the surfaces in that folder. Generally this translates into putting the surfaces for each solid body into separate folders. This methodology prevents overlapping surface grids from cutting into each other. Notice for this case that the collar grid has surfaces in both cutting folders, so it will not be cut into by either surface. For collar grids that have a single surface that spans the intersection between two separate bodies, the user will typically need to use the split surface feature (described in the split surface section) to split the surface into two surfaces along the body intersection grid line, and put each

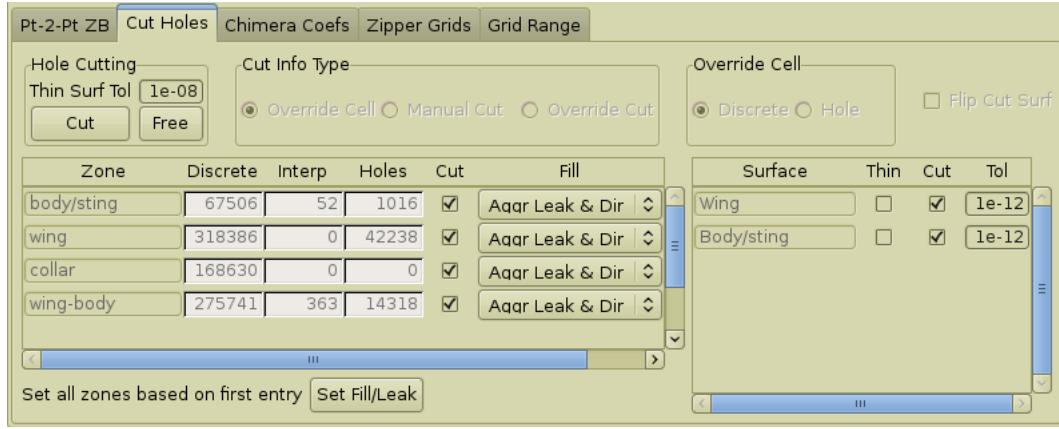


Figure 9.4: Cut Holes Tab

surface into the appropriate cutting folder.

After getting the cutting surfaces defined properly, we are ready to take a look at the **Cut Holes** tab in Fig. 9.4 on pg. 126. At first glance, the **Cut Holes** tab looks fairly complicated and involved. However, for most hole cutting cases, the user will only need to use a single button in the tab, the **Cut Holes** button, in order to cut holes in the grid. Most of the remaining portion of the tab is designed to give the user the flexibility to override the default automated hole cutting for special cases which may arise. These will be discussed in detail below.

After pressing the **Cut Holes** button, *GASPEX* may take several seconds or in the case of large problems, several minutes to perform the operation. A very general rule of thumb is on the order of the time it will take to run one iteration for the given problem.

Thin Surfaces

The thin surface tolerance input, Fig. 9.4 on pg. 126, is used to determine the precision of the intersection calculation for the special case of thin surfaces. In some CFD cases, a solid surface will have no or very little volume inside of it (*e.g.*, a splitter plate). The grids on either side of the surface share the same or nearly the same (x,y,z) locations. Even though these surfaces have no interior volume, they still need to cut holes in the other grids. In order to do so properly, the thin surface tolerance was added. From the user's perspective, the default value of 1.e-8 is a safe starting point, and 1.e-4 has worked well in some cases where 1.e-8 did not. If a thin surface isn't cutting properly, then adjust this value or use the individual zone thin surface override discussed below. The value should never be outside the range 0.1 to 1e-12. Thin surfaces that are likely to cause more problems are those which are curved, and have different grid distributions on either side of the surface (such that the faceting of the grid causes the surfaces to cross over).

As mentioned above, thin surfaces can also be explicitly defined by the user through the use of the **Thin** buttons in the surface table, Fig. 9.4 on pg. 126. If a user knows that the

entire surface defined in a given surface folder is “thin”, then it is advisable to use the **Thin** button to explicitly set the surface so that there is no chance of an error occurring due to precision issues. However, there may be cases where part of a surface is thin, and part is not, in which case the user must rely on the thin surface tolerance.

Intersection Tolerances

The intersection tolerance for thin surfaces is a single value input located next to the **Cut Holes** button. There is also a separate intersection tolerance for each surface located in the surface table, Fig. 9.4 on pg. 126. These tolerances are used to determine the precision of the intersection calculation between a cutting surface and a grid line. The default value is 1.0e-12, which is typically the precision of a 64-bit calculation. Of all the tolerances in the **Zonal Boundaries** tab, this one is least likely to need changing.

Cell Types

Inside the **Cut Holes** tab, you will notice a table listing each zone and the number of discrete, interp, and hole cells associated with it (Fig. 9.4 on pg. 126). A discrete cell is what you would think of as a normal cell. If we were not performing Chimera, then every cell in the problem would be a discrete cell. An interp cell is one that must have the flow variables interpolated to it. If a zone has cells that are cut, then the outer most faces become the boundary and again the first two layer of cells become interp cells. Hole cells, as you might expect, are cells that are cut from the zone (or blanked out). These cells are no longer used in the CFD computation.

When the hole cutting procedure is complete for the wing-body case, you will notice mostly non-zero numbers in the “holes” column for the zone list table. The collar zone is the only one that does not contain cells that overlapped the wing or body surfaces, and therefore has no holes cut from it. Both the wing and the body zones will have cells cut out of the computation.

9.4.3 Hole Filling Algorithms

The hole cutting algorithm in *GASPE*x is based on computing intersections between cutting surfaces and the overlapping volume grids: a cookie- cutter-like approach. In the ideal case, the intersection (hole) boundaries created should form closed surfaces. The cells on the “inside” of these closed surfaces can then be easily converted into hole cells (*i.e.*, “filling” the inside of the hole boundaries with hole cells). The problem is that in many cases (due to precision or overlapping or intersecting surfaces), the hole boundary surfaces do not form simple, closed surfaces and either have small “leaks” (or leaky holes) or in the case of multiple surfaces intersecting and overlapping, multiple boundary surfaces might overlap in a complex fashion. If a simple hole filling algorithm was used on a grid with a leaky hole, the entire grid might end up being converted into hole cells. In the case of multiple overlapping surfaces, regions inside of one surface, but outside of another might be left as discrete cells instead of

being converted into hole cells. Therefore, it is necessary to employ some more sophisticated techniques that attempt to fill the cut-out regions properly for complex cases.

There are currently three methods available to the user for filling the holes: the **leak** method, the directional (**dir**) method, and a combination of the two methods. Each of those methods has an aggressive mode and a conservative mode - aggressive meaning error on the side of making more holes, and conservative meaning error on the side of less holes. The **leak** method employs a leaky-hole corrector algorithm in an attempt to fix any leaky-hole surface before filling in the holes by essentially expanding holes in all directions until a hole boundary surface is reached. The directional method fills in the holes inside the boundary surfaces separately for each of the three logical directions: i,j,k. It then compares the results of the three sets of filled-in holes to determine which holes are legitimate and which are improper. The **leak** and **dir** combination method (which is the default) combines the results from both methods.

In the *GASPer* GUI, the **Fill** menu in the zone list, Fig. 9.4 on pg. 126, is used for setting the method in which the hole regions are 'filled in' for each zone during the hole cutting process. For many cases, all six hole filling algorithms will give essentially the same results. However, there can be instances where problems may arise such as leaky holes (the cutting surfaces do not cut out a completely closed region), or when a zone is completely inside of a surface (no intersections), to name a few. In these cases, the choice of hole filling algorithm gives the user some flexibility in controlling the resulting holes.

For most cases, if a zone is having too many holes cut, then either choose a conservative choice instead of an aggressive one or choose the **dir** (directional) method instead of the **leak** (leaky-hole corrector) method, and vice versa for the case of not enough holes being cut. There are, however, certain cases (typically with intersecting cutting surfaces) in which the **dir** method may result in more holes than the **leak** method. This is due to the fact that the leaky-hole corrector in the **leak** hole filling algorithm can have difficulty determining the difference between a leaky hole and a region that has been partially cut by a surface which lies beneath another cutting surface. The **leak** algorithms can produce better results for these zones by reducing the **Leak** number.

The **Leak** input for each zone is used for setting the maximum number of cells that will be corrected for a given detected leaky hole. Most leaky holes are small, so the default value is 10 cells. If a zone is becoming all holes after cutting, try increasing this number (say to 100). If you are not getting enough holes (*i.e.*, the surface cuts, but the interior cells are not turned into holes - particularly in regions where multiple surfaces overlap), try reducing this number. Note that the **Leak** parameter only applies to the **leak** hole filling algorithms.

Hole Cutting Overrides

The hole cutting in *GASPer* is very automated and generally requires very little user input. However, occasionally the user may want to override the default behavior in order to achieve a desired result. The first step in setting up inputs for overriding the default cutting is to select a zone and grid range over which to apply the override. This is done much the same

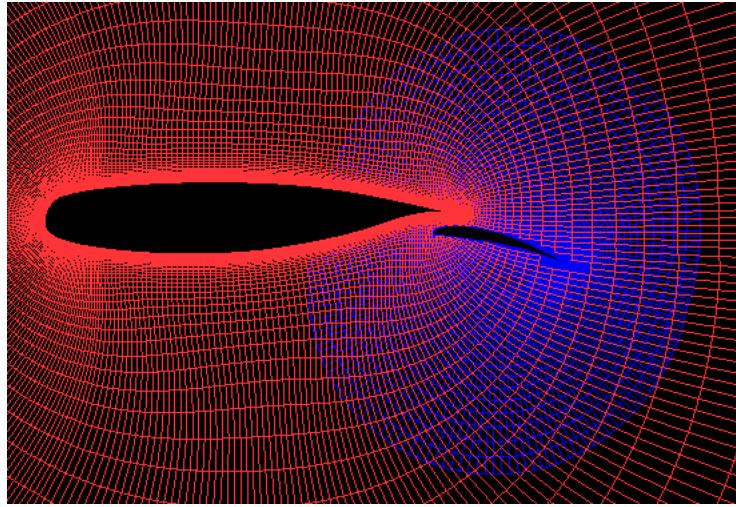


Figure 9.5: Multi-element airfoil with no cutting overrides

way as an output node is created. The user selects the desired zone or surface from the tree list, and drags and drops that node into the “Cutting Overrides” folder in the tree view. A new cutting override node is then created for that zone. If a zone node was originally selected, then the initial grid range for the cutting override is set to the entire zone (*i.e.*, a volume range type with the individual ranges set to the mins and maxes). If a surface node was dragged and dropped, then the initial range is set to the range of the surface node (e.g. an I, J, or K surface with the mins and maxes set to the bounds of the surface). These initial ranges are just that - initial. They can be edited in the **Grid Range** tab. This behavior is very similar to that of the output nodes.

There are three basic types of cutting overrides: **Override Cell**, **Manual Cut**, and **Override Cut** (see Fig. 9.4 on pg. 126). The example we will use in this section to demonstrate the various overrides is a simple, 2D multi-element airfoil with two zones - the main airfoil, and the smaller flap-like airfoil in the rear. The basic hole cutting for this case without any overrides is given in Fig. 9.5 on pg. 129.

The **Override Cell** cutting override allows the user to directly specify the cell type for the given zone and grid range regardless of the results of the hole cutting algorithm. In other words, the hole cutting will be performed first and result in cell types for all of the cells in the problem. The code will then look at the **Override Cell** type cutting overrides, and set the cell type for cells in the given zone and grid range to the cell type specified in the neighboring **Override Cell** radio box. Currently the choices are either **Discrete** or **Hole**. The most obvious and common use of this override would be when the user wants to manually blank out a region in a zone that isn’t cut out by the automated routine.

An example of this for the multi-element airfoil case is shown in Fig. 9.6 on pg. 130. The red box graphically represents the grid range of the flap zone which has been created as an override cell type node of type holes. The blue grid is the resulting flap grid (discrete and interp cells only) after the hole cutting. In comparison to the no override case, the cells in

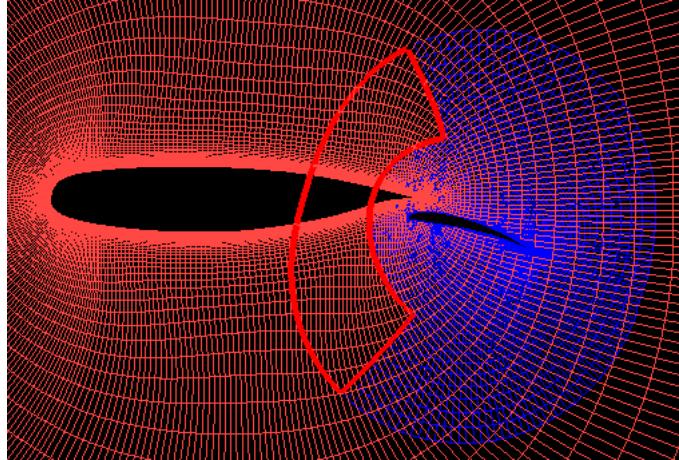


Figure 9.6: Multi-element airfoil with override cell type

the red box portion of the flap grid are now all set to type hole instead of a mix of discrete and hole (as originally determined by the surface cutting algorithm).

The **Manual Cut** cutting override allows the user to create additional surfaces for cutting holes from the given zone and grid range. If a volume grid range is selected, then six new cutting surfaces will be created from the six faces that define the limits of the volume. Otherwise, if a surface type range is selected then the corresponding single cutting surface is created.

The **Cut** button in the zone list shows which zones will be cut by the surface. The default **Cut** values (all zones not connected via pt2pt zonal boundaries to the cutting zone) can be overridden by setting the **Cut** button in the list. The default values are restored when the user selects a **Cut Info Type** other than **Manual Cut**.

The **Flip Cut Surf** button flips the direction of manual cutting surface(s). The default for a 'surface' type grid range is for the outward normal of the surface to point in the positive coordinate direction. The default for a 'volume' type grid range is for the surfaces to point outward from the center of the volume (*i.e.*, the holes are cut 'inside' the volume).

An example of manual cutting surfaces for the multi-element airfoil case is shown in Fig. 9.7 on pg. 131. The red box graphically represents the grid range of the airfoil zone which has been created as a manual cut type node (volume grid range with six cutting surfaces - really only 4 in this case since it's 2-D). The blue grid is the resulting flap grid (discrete and interp cells only) after the hole cutting. In comparison to the no override case, the cells in the red box portion of the airfoil grid now cut out a hole in the flap (blue) grid. Note that the default behavior of a volume type manual cutting surface is to cut the hole inside of the region defined. If we flipped the cutting surface, then all of the blue grid would be cut except for the region inside of the red box area of the airfoil grid as shown in Fig. 9.8 on pg. 131.

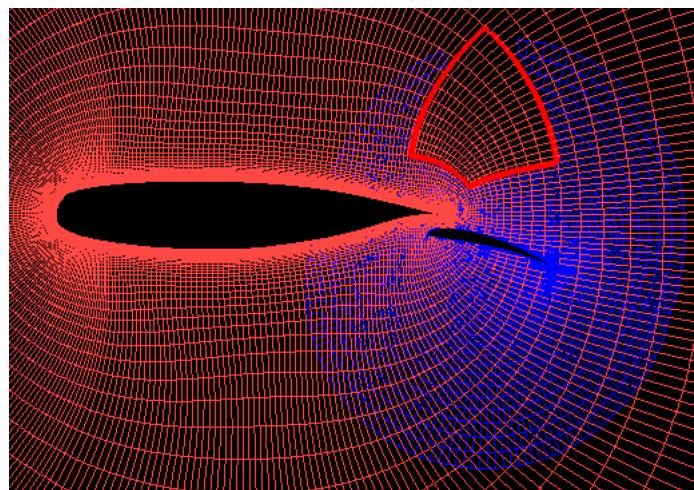


Figure 9.7: Multi-element airfoil with manual cutting surfaces

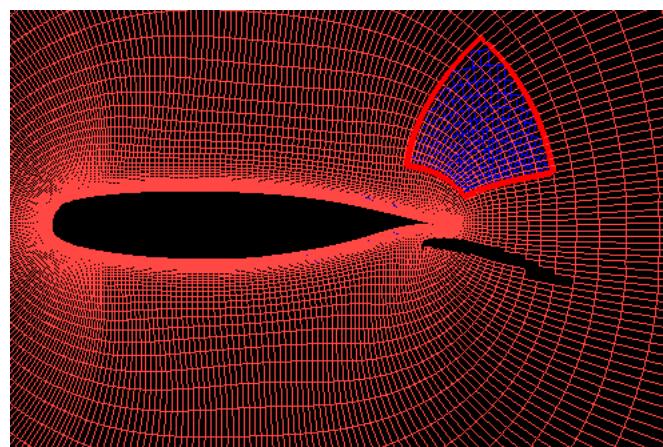


Figure 9.8: Multi-element airfoil with manual cutting surfaces flipped

Important: When creating manual cutting surfaces, it is important to keep in mind that like the automatic cutting surfaces (solid surfaces), the manual cutting surfaces need to form closed regions in order to prevent leaky holes and bad hole cutting. A volume type manual cutting surface by definition is a closed surface, so this is only a concern for single surface cutting surfaces. They can form a closed region in conjunction with other solid surfaces or the user can use multiple manual cutting surfaces to form a closed region.

So what is the difference between creating holes by manually overriding the cell type versus using manual cutting surfaces? There are two main differences. First, when overriding the cell type, the hole region is defined by a rectangular range of the zone in which the holes are being created. For the manual cutting surface case, the region for cutting out is defined by the cutting zone, not the zone where the holes will be created. The other major difference is that the override cell type method only creates holes in a single zone for each cutting override specified. In the case of manual cutting surface(s), more than one zone may have holes cut out depending on the values defined by the **Cut** buttons in the zone table.

The **Override Cut** cutting override allows the user to override the default “can be cut” behavior for a given zone and grid range. When the user groups the cutting surfaces in the tree view, default values for each zone of whether or not it can be cut by each of the surfaces is determined based on the zones in each cutting folder and the pt2pt zonal boundary connectivity of the problem. When a user creates an **Override Cut** cutting override node, the default values for whether or not the given zone can be cut by each of the surfaces is shown in the surface table (Fig. 9.4 on pg. 126) by the **Cut** buttons. The user can choose to alter the values for any of the **Cut** buttons to change whether or not the given zone and grid range are cut by each of the cutting surfaces in the table. The default values are restored when the user selects a **Cut Info Type** option other than **Override Cut**.

An example of overriding cutting for the multi-element airfoil case is shown in Fig. 9.9 on pg. 133. The red box grid graphically represents the grid range of the flap zone which has been created as an override cut type node with the **Cut** button for the airfoil surface manually overridden and therefore turned off. The blue grid is the resulting flap grid (discrete and interp cells only) after the hole cutting. In comparison to the no override case, the cells in the red box portion of the flap grid are no longer cut as before (there is a blue grid beneath them, though it is difficult to see).

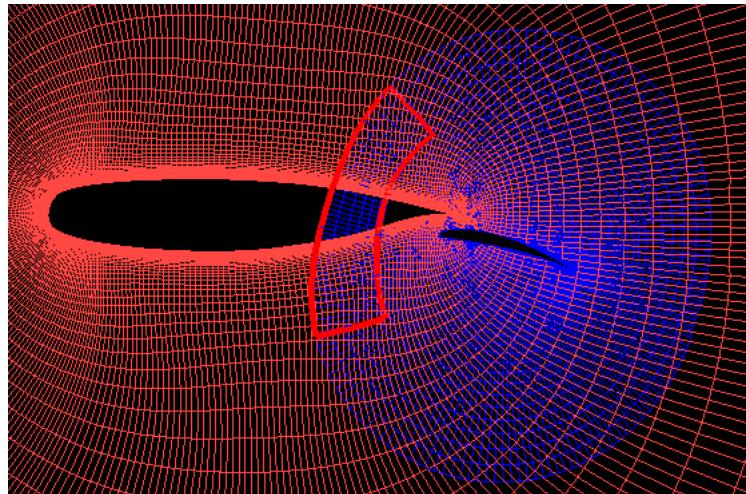


Figure 9.9: Multi-element airfoil with airfoil cutting surface overridden

Important: When creating overriding cutting surfaces, it is important to keep in mind that the leftover (those that are not overridden) cutting surfaces for the given zone that has the **Override Cut** need to form closed regions in order to prevent leaky holes and bad hole cutting. Even if the override is for the entire zone, if the override is for a surface that intersects a zone that is connected to the given zone via a pt2pt zonal boundary, problems can arise if care is not taken to ensure closed cutting regions. The example shown is actually not a closed region and only works because the conservative directional hole filling was used. If the leak hole filling was used, the results might be catastrophic - almost the entire flap grid could turn into hole cells.

A more correct way of implementing an override cutting for this case would be to set the range to the entire flap grid, so that the airfoil grid would not cut any of the zone, and therefore not result in cut regions that were not closed. Of course, overriding the cutting in this case doesn't really make sense at all. So when does it make sense? The best answer I can come up with is - not very often. Typically if you want certain cells in a zone to be discrete cells (*i.e.*, not be cut), then it's simpler and safer to use the **Override Cell** cutting override and set them explicitly to discrete cells. It accomplishes the goal of preventing the cells from being cut, and there is no concern about creating leaky holes that could have catastrophic results on the remaining cells.

9.4.4 Computing Chimera Coefficients

The next step in preparing *GASPE* to run properly on a Chimera grid system is to compute the zonal boundary (or Chimera) interpolation coefficients. Like hole cutting, chimera coefficients do not need to be performed on non-overlapping grid systems. Likewise, for problems where some zones overlap and some do not, the chimera coeffs will be computed over all the grids, and those which are unaffected will not have any coefficients found.

Identifying the Chimera Surfaces

Before computing the zonal boundary interpolation coefficients for a Chimera problem, we need to ensure that we have properly identified all of the “interp” cells. Interp cells are created in two ways. First, as previously discussed, they are created by the hole cutting algorithm as a two-layer set of “fringe” cells around each hole to allow the passing of solution information to the region around the hole (*i.e.*, they act as a local boundary condition around the hole). Second, they are created as a set of two-layer boundary condition cells for each surface that is placed inside the “Chimera ZB” surface folder. The “Chimera ZB” surface folder is a permanent folder which exists under “Zonal Boundaries”. It is designed to contain all of the overlapping zonal boundaries (otherwise known as Chimera zonal boundaries). It is the responsibility of the user to place the appropriate surfaces into the “Chimera ZB” folder prior to computing the zonal boundary interpolation coefficients. Surfaces that belong in the “Chimera ZB” folder are those that meet the following criteria:

1. The surface overlaps another zone (*i.e.*, it’s not a point-to-point zonal boundary).
2. The surface does not represent a physical boundary (e.g. solid surface, symmetry plane, etc).
3. The surface is in the interior of the domain (*i.e.*, it’s not an inflow/outflow boundary). Part of the surface may be inside of another solid surface (in which case that portion would be cut from the computation).

Generally speaking, the “Chimera ZB” surfaces are the “left-over” surfaces that don’t belong in any of the typical boundary condition folders.

Computing the ZB Coefficients

The majority of the setup for computing the interpolation coefficients is located in the **Chimera Coefs** tab in Fig. 9.10 on pg. 135. As with hole cutting, the **Chimera Coefs** tab looks more complicated than it is for most cases. Typically, the user will only need to use a single button in the tab, the **Compute ZB Coefs** button to compute the interpolation coefficients. This command not only computes the stencil for each interp cell, but assigns each cell in the stencil a coefficient value used in the interpolation process.

The process of determining which donor cells to use in the interpolation stencil for each interp cell is discussed in more detail in the next section on quality cut-off and quality

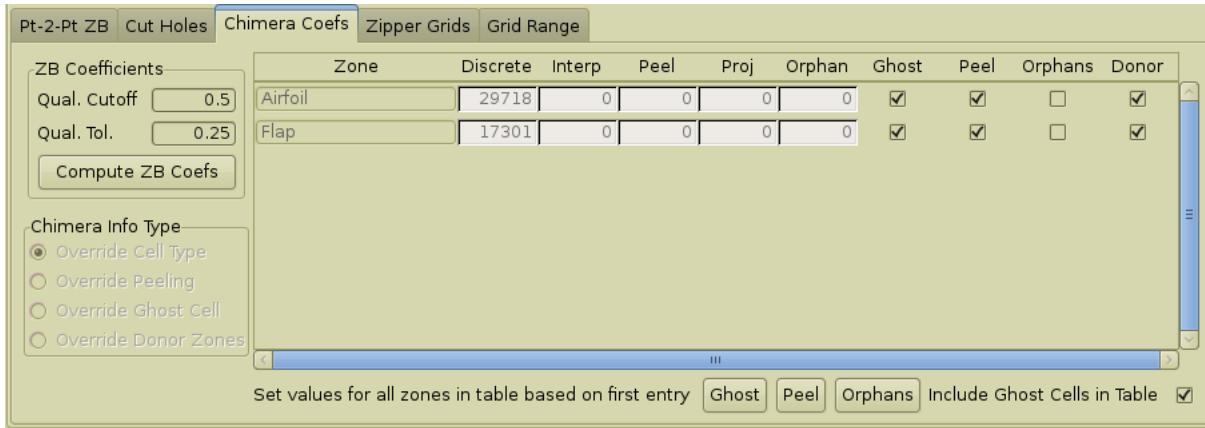


Figure 9.10: The chimera coefs tab

tolerance. The default values for quality cut-off and quality tolerance are 0.5 and 0.25 respectively. If these values are used with problems that have minimal overlap on coarser sequence levels, the result can be that a good percentage of the interp cells will end up as orphan cells. It often requires several trial and error runs to determine what values are best suited for a given problem. Fortunately, unlike the split surface and point-to-point algorithms, a “failure” (e.g. too many orphans) in the computing of zonal boundary coefficients does not have any catastrophic consequences and is easily redone. Simply change the values for the quality cut-off and quality tolerance, and press the **Compute ZB Coefs** button a second time.

Important: Zonal boundary coefficients must be recomputed after performing a decomposition. This has to do with the fact that they are currently stored in the decomped zones which are deleted when a new decomposition is done. So if you compute the coefs and then decompose the grid into more zones or change the decomposition, then you will need to compute the ZB coefficients again. Hole cutting is preserved during a decomp, and so the hole cutting should be performed before decomping because it is more efficient on a non-decomped grid.

Quality Cutoff and Quality Tolerance

The quality cut-off and quality tolerance, Fig. 9.10 on pg. 135. are user-input values that help determine the stencil selection for each interp cell. To choose the values for these parameters, you will need to understand how stencils are selected for each interp cell. Recall that each interp cell may have one or more stencils from which to select. In the wing/body example, when interpolating to the wing grid, in the region of the wing body intersection, there will be two candidate stencils. One candidate from the body grid, and one candidate from the collar grid. Each stencil has a “quality” value and a “cell difference” value that is used in the stencil selection process. What these terms mean and how they are used in the

stencil selection process will now be discussed.

An interpolation stencil, which each interp cell will have when performing a Chimera operation, is made up of other cells in the grid. Cells used in a stencil are either hole cells, interp cells, or discrete cells. Each of the cell types are given a numerical value for assessing the quality of the stencil. A hole cell has a value of 0, an interp cell has the value 0.1, and a discrete cell has a value of 1. For each stencil, a quality value is then computed based upon the cell type values for the stencil. Quality values will range from 0 to 1. A stencil with all hole cells would have a quality of 0, while a stencil with all discrete cells would have a value of 1. This is the first of two parameters that will be used to determine stencil selection.

In addition to the quality value, each stencil will have a cell difference value. The cell difference is a measure of how close in size the stencil cells are to the cell that is being interpolated to (the interp cell). If all the cells in the stencil are identical in size to the interp cell, the cell difference would be zero. As the difference in cell sizes increase, the value of the cell difference increases.

The relation between the quality cut-off and quality tolerance with the stencil quality and cell difference will now be discussed. The quality cut-off parameter is the smallest value for stencil quality that a user deems acceptable. A value of 1 for the cut-off would require every stencil to contain only discrete cells (the ideal case). A value of 0 states that every stencil is acceptable. This is usually not a good idea since hole cells do not contribute information to the stencil. The value for the quality cut-off must be between 0 and 1.

With a list of stencils computed for each interp cell, the quality cut-off parameter is used to throw out all stencils below the cut-off value. Once a stencil is thrown out, it is no longer used in the selection process. If there are no stencils which meet the cut-off criteria, the interp cell becomes an orphan. If only a single stencil remains after checking stencil values with the cut-off, then that stencil is used in the interpolation. If two or more stencils remain, and each one has the same quality value, then the stencil with the smallest cell difference value is selected. If two or more stencils remain and each has different quality values, and each meets the cut-off criteria, then the quality tolerance parameter is used to further reduce the list of acceptable stencils.

The quality tolerance is used to gauge the difference between stencil quality values. For example, if a given interp cell has two stencils to select from, and the quality value of one stencil is 0.70, and the second has a quality value of 0.52, then the quality difference between the two stencils is 0.18. If this value is less than the specified quality tolerance, the second stencil is kept in the list. If the quality difference value is greater than the quality tolerance, the stencil is thrown out. To determine the quality difference, the largest quality value will always be used as one of the terms in the difference. The quality tolerance parameter is used to further reduce the number of stencils to select from by throwing out stencils that have values greater than the user specified quality tolerance. Keep in mind that this only comes into play if there are two or more stencils that already meet the quality cut-off. After going through both the quality cut-off and quality tolerance tests, a stencil is then chosen based upon which has the smallest cell difference.

An example may help clarify the stencil selection process. Suppose a given interp cell

has four possible stencils which are listed below.

Stencil 1: quality = 0.70, cell difference = 0.5

Stencil 2: quality = 0.45, cell difference = 0.5

Stencil 3: quality = 0.52, cell difference = 0.4

Stencil 4: quality = 0.65, cell difference = 0.4

The user has specified a quality cut-off value of 0.5 and a quality tolerance value of 0.1. The first step in the stencil selection process is to check the cut-off value. Stencil 2 is below the quality cut-off and is thrown out. The remaining three stencils all meet the quality cut-off. Since the quality value is different for the three remaining stencils, the quality tolerance will be used to narrow our stencil selection. Stencil 1 has the best quality, so stencil 3 and 4 will be compared to stencil 1 to check the quality tolerance. Stencil 3 has a quality of 0.52 which is 0.18 less than stencil 1. The quality difference (0.18) is greater than the quality tolerance of 0.1, so stencil 3 is thrown out. Stencil 4 has a quality of 0.65 which is 0.05 less than stencil 1. The quality difference (0.05) is less than the quality tolerance, so we have to compare cell difference parameters to determine which stencil to keep. Stencil 4 has the lower cell difference, so it is selected.

Ghost Cells

One of the Chimera parameters that a user might be more likely to alter on a consistent basis is the “Ghost” setting for each zone in the table, Fig. 9.10 on pg. 135. This parameter allows the user to choose whether or not to use the two interior cells nearest a “Chimera ZB” surface boundary as the interp cells, or the two ghost cells at the boundary as interp cells. The default value is to use the ghost cells because this typically provides a larger area of overlap which can reduce orphans in cases when there is not otherwise enough overlap. However, the (x,y,z) location of the ghost cells is based on projecting the boundary of the Chimera ZB surface, and this sometimes can have a bad result. For example, the ghost cells end up being underneath a solid surface and therefore end up being orphans. The user can decide whether or not to use ghost cells on a zone-by-zone basis. It is also possible to override the ghost cell setting for a region within a zone (e.g. a zone can have one surface that uses ghost cells, and one that uses interior cells) using one of the Chimera Override features discussed.

Peeling

For most cases, the peeling parameter, Fig. 9.10 on pg. 135, has the biggest impact on the end result of the Chimera zonal boundaries. Peeling is the process in which cells are removed from the computation where there is an extensive overlapping of grids. When peeling is selected, *GASPex* will remove unnecessary cells from the computation in regions of the flow

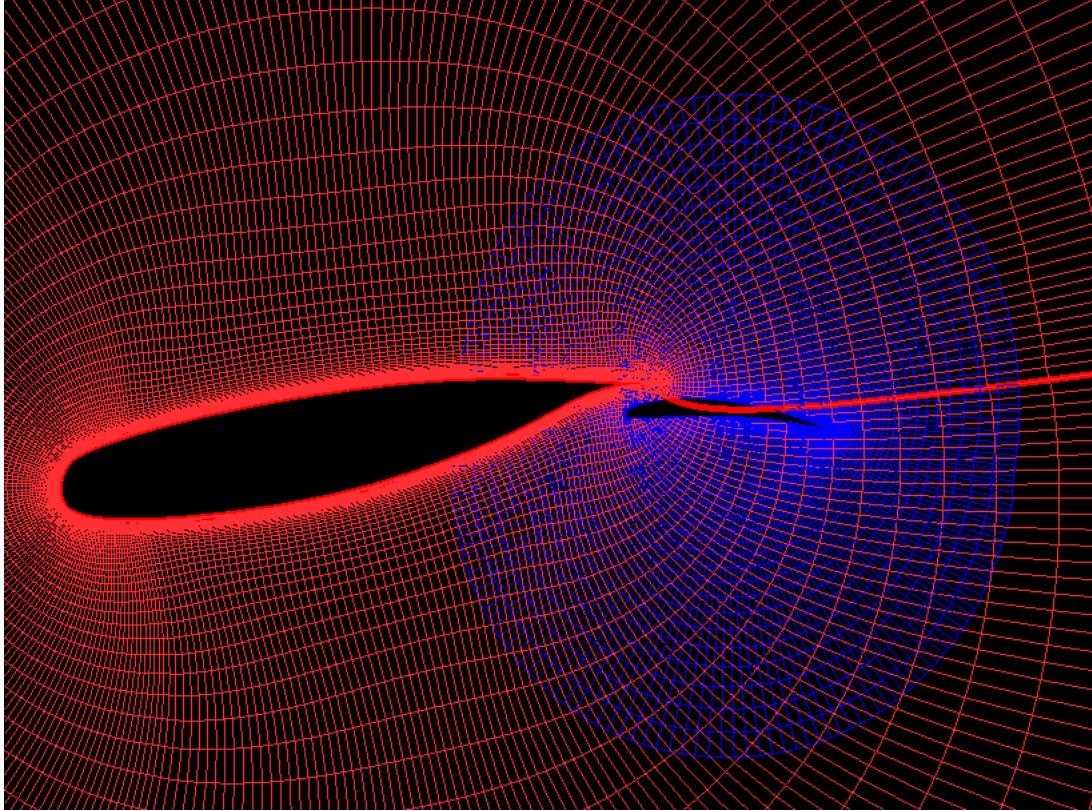


Figure 9.11: Multi-element airfoil without peeling

field where two or more grids are used to discretize the same area. This process is intended to improve the interpolation stencil quality. The peeling algorithm is designed so that coarser zones interpolate from finer zones, and the best overlap between two zones is defined as the region of overlap between the zones where the cell sizes are most similar. When cells are peeled from a zone, a new fringe of interp cells is created. The remaining peeled cells are marked as peel cells, and from a flow solver perspective, they are treated the same as hole cells. They only remain as peel cells (as opposed to hole cells) for the purpose of visualization in the GUI. In general, the use of peeling is highly recommended as it typically results in better quality solutions due to improved interpolation.

An example of the benefits of peeling can be seen in a simple 2-D multi-element airfoil case. Without peeling, the discrete and interp cells are shown in Fig. 9.11 on pg. 138 . With peeling turned on, the discrete and interp cells are given in Fig. 9.12 on pg. 139 . The overlap has been significantly reduced, and the interpolation is now happening much further from the solid surfaces (*i.e.*,in regions with smaller gradients) which improves solution accuracy.

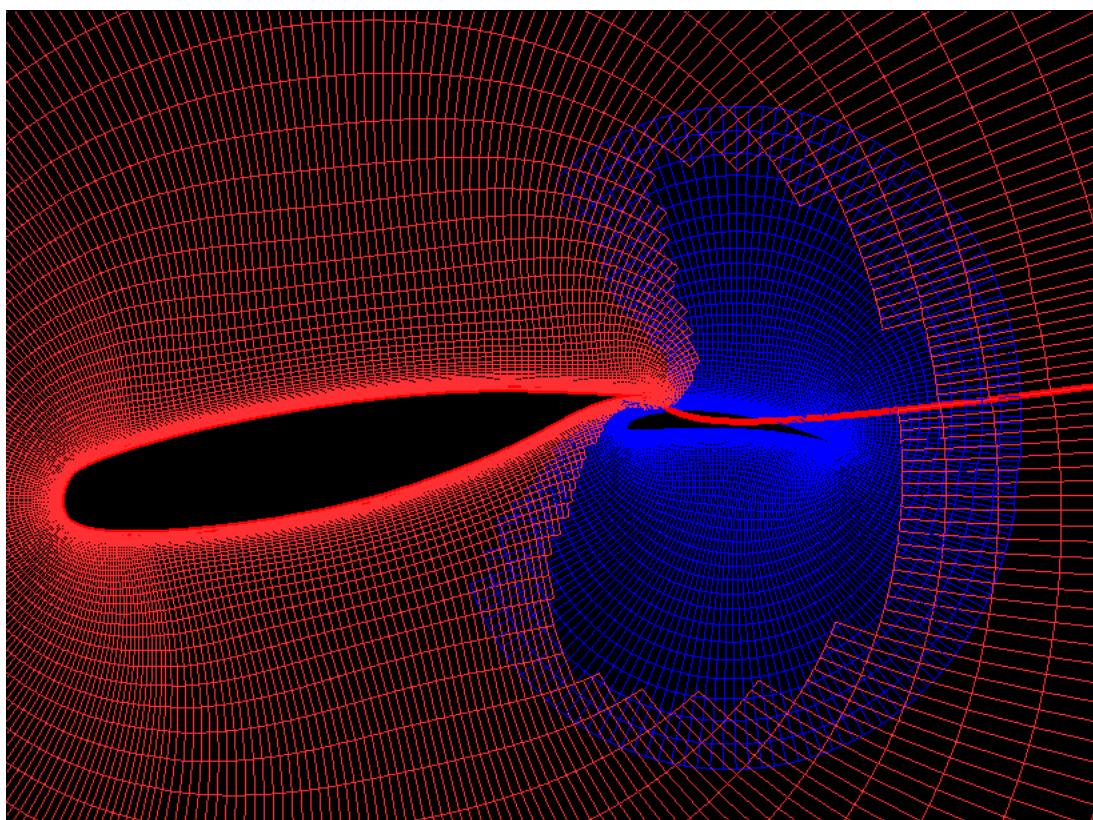


Figure 9.12: Multi-element airfoil with peeling

Donor Cells

The “Donor” option in the Chimera coefficients table (see Fig. 9.10 on pg. 135) is used to control whether a zone will be used in the stencil process. In other words, when overlap stencils are formed, this controls if the zone will “donate” cells to the stencil. This option is used in conjunction with the **Override Donor Zones** (see Sec. 9.4.4 on pg. 141).

Projection

For cases in which a single solid surface is defined by overlapping highly-stretched viscous grids, projection is applied for the interpolation cells near the solid boundary in the overlap region. The most typical use of projection is for viscous grids that use “collar” grids around surface/surface intersections. After the coefficients are calculated, the number of projection coefficients can be seen in the Chimera coefficients table under the “Proj” heading.

Important: Note that the actual grid coordinates of the collar grid have not been change during the projection. Only the local coordinate values stored inside the zonal boundary coefficient information are altered so that an appropriate stencil can be found. Therefore, the projection has no impact on other functions outside of the zonal boundary coefficient computation.

Cell Types

Inside the **Chimera Coefs** tab, you will notice a table listing each zone and the number of discrete, interp, peel, proj and orphan cells associated with it, Fig. 9.10 on pg. 135. The **Cut Holes** tab has a similar table with each zone and the number of discrete, interp, and hole cells, Fig. 9.4 on pg. 126. The number of discrete cells should be the same in both lists. A discrete cell is what you would think of as a normal cell. If we were not performing Chimera, then every cell in the problem would be a discrete cell. The number of interp cells for the same zone in each list will likely be different. In both cases, an interp cell can be thought of one that must have the flow variables interpolated to it. In the **Cut Holes** tab, an interp cell represents the interpolation fringe around a cut-out hole. In the **Chimera Coefs** tab, an interp cell represents a cell (or ghost cell) at the outer boundary interpolation fringe of a zone (*i.e.*, those cells represented by the surfaces in the “Chimera ZB” folder). Hole cells, as discussed in the hole cutting section, are cells that are cut from the zone (or blanked out). Peel cells are cells that have been turned into hole cells in the process of peeling back the grids to obtain an ideal overlap of the grids (see Sec. 9.4.4 on pg. 137). Proj (projection) cells are interpolation cells that have been projected above a surface in order to find a viable donor (see Sec. 9.4.4 on pg. 140). And finally, orphan cells are interpolation cells that could not find a viable interpolation stencil. During the solution process, these cells are updated with the average values of surrounding discrete and interpolation cells in order to minimize their impact on solution quality. Obviously, it is best to have zero orphan cells, but it is not a requirement.

The cell types also have an impact on output and visualization. In the **Solution Visualization** or **File Output** (see Sec. 17.5 on pg. 354), you have the option of selecting which cell types to display. By default, the discrete and interp cells are displayed since they are the ones that have valid solution values. However, we can also use this feature to visualize the hole cutting, peeling, projection or orphans. More detailed discussions of this feature are given in the post processing section and the Chimera tutorial.

Chimera Overrides

Computing interpolation (Chimera) coefficients in *GASpex* is very automated and generally requires very little user input. However, like hole cutting, occasionally the user may want to override the default behavior in order to achieve a desired result. The first step in setting up inputs for overriding the default Chimera coefficient calculation is to select a zone and grid range over which to apply the override. This is done much the same way an output node is created. The user selects the desired zone or surface from the tree list, and drags and drops that node into the “**Chimera Overrides**” folder in the tree view. A new chimera override node is then created for that zone. If a zone node was originally selected, then the initial grid range for the chimera override is set to the entire zone (*i.e.*, a volume range type with the individual ranges set to the mins and maxes). If a surface node was dragged and dropped, then the initial range is set to the range of the surface node (*e.g.* an I, J, or K surface with the mins and maxes set to the bounds of the surface). These initial ranges are just that - initial. They can be edited in the **Grid Range** tab. This behavior is very similar to that of the output nodes.

There are four types of chimera overrides: **Override Cell Type**, **Override Peeling**, **Override Ghost Cell**, and **Override Donor Zones**. The example we will use in this section to demonstrate the various overrides is the same one used for the cutting overrides: a simple, 2D multi-element airfoil with two zones - the main airfoil, and the smaller flap-like airfoil in the rear. The compute coefficient computation using the defaults values (*i.e.*,ghost cells and peeling turned on) with no overrides is given in Fig. 9.13 on pg. 142. The blue grid represents the peeled cells of the flap grid, the light blue are the interpolation cells of the flap grid, the red are the peeled cells of the airfoil grid, and the yellow are the interpolation cells of the airfoil grid. Notice that the interpolation cells do not appear to go all the way around the outer boundary of the flap grid. This is because ghost cells are turned on, and they do not appear graphically. However, they are included in the count of interp cells in the table and can be excluded from the table values by turning off the “**Include Ghost Cells in Table**” option.

The **Override Cell Type** chimera override allows the user to directly specify the cell type for the given zone and grid range regardless of the results of the hole cutting algorithm. When the chimera coefs are computed, the code will look at the **Override Cell Type** chimera overrides, and set the cell type for cells in the given zone and grid range to interpolation cells. The most obvious and common use of this override would be when the user wants to manually force a region in a zone that is discrete cells to instead interpolate the solution

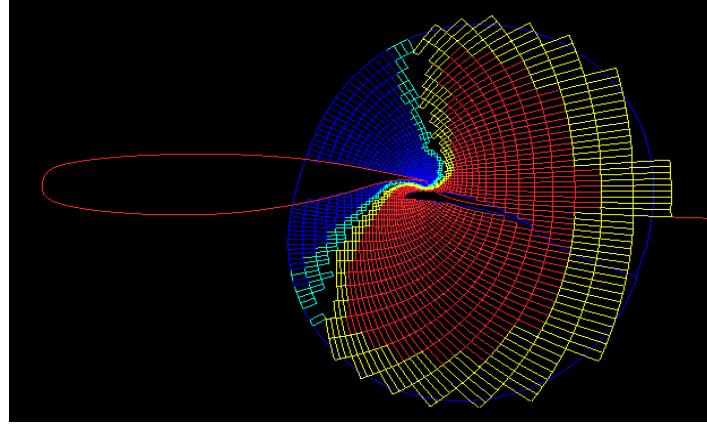


Figure 9.13: Multi-element airfoil with no chimera overrides

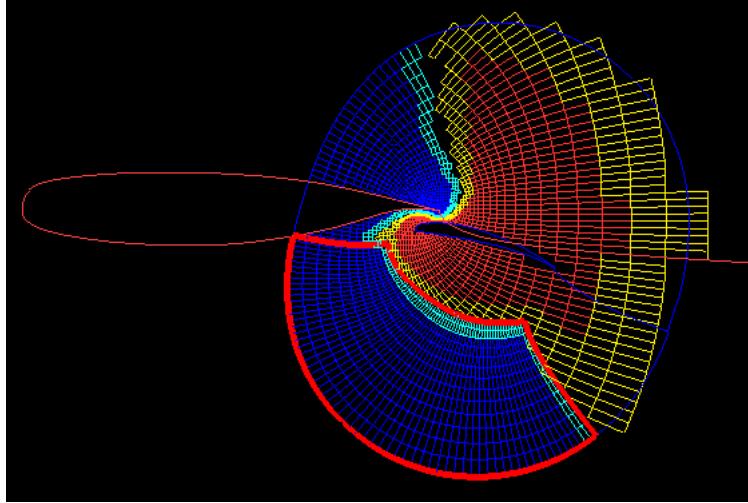


Figure 9.14: Multi-element airfoil with interp override cell type

from another zone. This can be useful when the user wants to locally alter the behavior of which zones are peeled in the peeling algorithm.

An example of this for the multi-element airfoil case is shown in Fig. 9.14 on pg. 142. The red box graphically represents the grid range of the flap zone which has been created as a chimera override cell type node. As in the no override case above, the blue grid represents the peeled cells of the flap grid, the light blue are the interpolation cells of the flap grid, the red are the peeled cells of the airfoil grid, and the yellow are the interpolation cells of the airfoil grid. In comparison to the no override case, the cells in the red box are now interp and peeled cells of the flap grid whereas before they were mostly interp and peel cells for the airfoil grid. We have forced the flap grid to peel in this area instead of the default behavior of the code which was to peel the airfoil grid in this region. This was the default behavior because the airfoil grid is coarser in the overlap region between the two zones. However,

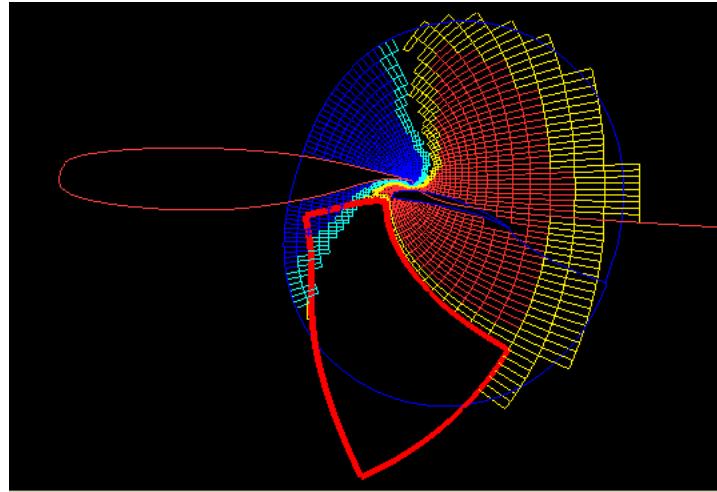


Figure 9.15: Multi-element airfoil with override peeling

some overlaps between two grids may contain local areas where the grid to be peeled (the overall coarser grid) is finer, and so the user can use this method to set those regions to interp cells and alter the local peeling results.

Another method for altering the local peeling is through the use of the **Override Peeling** button. This override will specify that the peeling for the cells in the selected zone and grid range will be opposite of the value for the entire selected zone. The value for a given zone is shown in the zone table list by the **Peel** button. If you want to change the peeling behavior for an entire zone (as opposed to part of a zone), then it is more efficient to just switch the value of the **Peel** button in the zone table list.

An example of overriding peeling for the multi-element airfoil case is shown in Fig. 9.15 on pg. 143. The red box graphically represents the grid range of the airfoil zone which has been created as a chimera override peeling node. As in the no override case above, the blue grid represents the peeled cells of the flap grid, the light blue are the interpolation cells of the flap grid, the red are the peeled cells of the airfoil grid, and the yellow are the interpolation cells of the airfoil grid. In comparison to the no override case, the cells in the red box are now discrete cells and for the airfoil grid whereas before they were mostly interp and peel cells for the airfoil grid. We have reduced the peeling region for the airfoil grid by overriding the default peeling in this local region. Typically the peeling gives the “ideal” overlap, but the **Override Peeling** gives users the ability to tailor the results of the peeling to meet their specific needs.

The **Override Ghost Cells** button specifies that the use of ghost cells vs. the two layers of interior boundary cells (for **Chimera ZB's** surface boundaries) for the boundary cells in the selected zone and grid range will be opposite of the value for the entire selected zone. Note that only **Chimera ZB's** boundary cells (ghost cells and two layers of interior cells at the boundary, not any other interior cells) in the selected grid range will be affected. The value for a given zone is shown in the zone table list by the **Ghost** button. If you want to

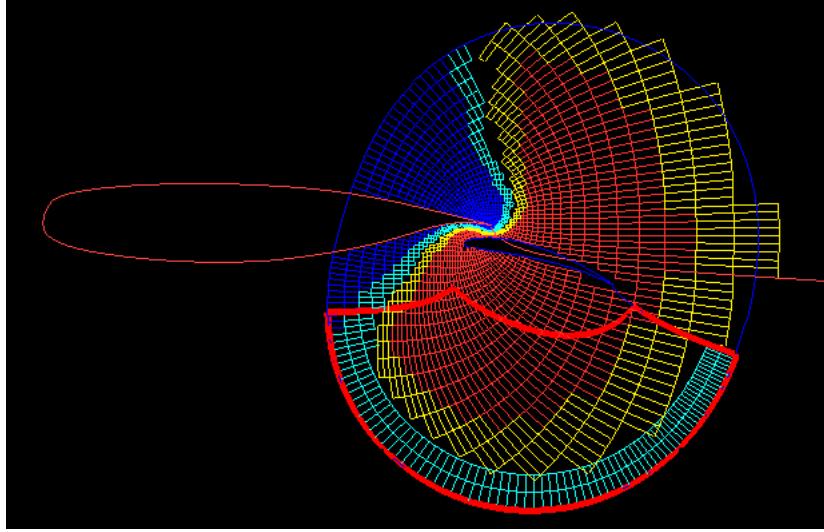


Figure 9.16: Multi-element airfoil with override ghost cells

change the ghost cell interpolation behavior for an entire zone (as opposed to part of a zone), then it is more efficient to just switch the value of the **Ghost** button in the zone table list.

An example of overriding ghost cells for the multi-element airfoil case is shown in Fig. 9.16 on pg. 144. The red box graphically represents the grid range of the flap zone which has been created as a chimera override ghost cells node. As in the no override case above, the blue grid represents the peeled cells of the flap grid, the light blue are the interpolation cells of the flap grid, the red are the peeled cells of the airfoil grid, and the yellow are the interpolation cells of the airfoil grid. In comparison to the no override case, the two-layers of interior boundary cells inside the red box for the flap grid are now interp cells. The algorithm is no longer using the ghost cells in this region as the two-layer “fringe” of cells around the exterior of the flap grid because we have overridden the behavior specified by the **Ghost** button for the flap grid. As a result, the peeling and interp region of the airfoil grid has shrunk somewhat in this region to account for the reduced number of discrete flap grid cells from which to interpolate. When peeling, the airfoil grid wants to only interpolate from discrete flap grid cells, so it can not peel as much if the number of discrete flap grid cells are reduced. Although it shows the results well, this is not a good example of when a user would be likely to override ghost cell interpolation. A more typical case would be when the physical location of some of the ghost cells (imagine adding on two more grid lines to the grid) are poor in comparison to the last two interior cell layers.

The **Override Donor Zones** button allows the user to specify which zones will serve as donor zones for the selected zone through the use of the **Donor** buttons in the zone table list. Currently the grid range for this override is ignored, and the override will be applied across the entire selected zone. The default **Donor** values for each zone (based on pt2pt zonal boundary info and simplified grid overlap calculations) are restored when the user selects an option other than **Override Donor Zones**.

A two zone case such as the airfoil is not a good example for this override since it is intended for problems in which a zone may have more than one donor zone (a donor zone is a zone in which a given interp cell is interpolating “from”). Consider a hypothetical case with three overlapping zones. Suppose for some reason the user wants the interp cells in Zone 1 to only interpolate from Zone 2, and not from Zone 3. This would be a case where the user could drag and drop Zone 1 into the **Chimera Overrides** folder, select the **Override Donor Zones**, and then set the value of the **Donor** button for Zone 3 to off. The user would then be ready to compute the coefficients, and Zone 1 would only find interpolation stencils from Zone 2. Be aware that by turning off potential donor zones, the risk of orphan cells increases.

Chapter 10

The Physical Models Frame

10.1 Introduction

In computational physics, a problem is simulated by solving mathematical equations using numerical methods. A physical (real-life) problem is therefore described using numerical parameters in order to properly model the physics of the problem. In *GASPE*, the physical model is used to describe the physics of a problem using numerical terms. For example, when solving the Navier-Stokes equations a chemistry model must be selected that defines how the air composition behaves inside the flowfield. The user must select the chemistry model that best defines the physical problem being simulated.

Any number of physical models can be created, which are then assigned to zones and to a run definition. Physical models are inserted or assigned to a zone when it is created or from the solution management (see Sec. 8.5 on pg. 96). In a similar fashion, a physical model is assigned to a run group (see Sec. 15.5 on pg. 257) inside the run definition.

For the zone initialization (see Sec. 8.2 on pg. 88), each zone should have at least one physical model associated with it. This serves two purposes. The first is to specify the initial solution for the zone when the solver begins. The second is to know what the solution unknowns (the primitive variables) are. Depending on how the user sets up the physical model, the number of equations being solved (number of solution parameters) will vary. The physical model assigned to a zone in the initialization section determines the solution unknowns for a given zone, which in turn corresponds to the variables stored in the solution file.

Since a problem may contain multiple zones and physical models, there must be some way of specifying what physical model describes what zone during the solver. This is done in the Run Definition section. When the user sets up a run definition (which describes what the solver will do when started), zones will be matched to physical models. See the Run Definition chapter (Chap. 15 beginning on pg. 239) for more information on this.

There are a number of physical model types supported by *GASPE*. This chapter will focus on understanding the physical model settings common to most all the physical models. The chapters that follow are devoted to each individual physical model type.

Most all the physical models contain the following information:

- A summary of the model, including the solution variables and equations being solved.
- Assignment of boundary conditions to surface families.
- Pointwise information for use with boundary conditions or source terms.
- Inputs to the boundary conditions and solution initialization (*e.g., Q Spec*).
- Coupling information describing the relationship between the current physical model, and other physical models in the current problem.

The physical model is accessed by selecting the **Physical Models** tab located next to the **Zonal Bouns** tab (located under the graphics window display). Because *GASPEX* has different types of solvers (fluid flow, solids, etc), there are different types of physical models corresponding to the different solvers. For example, when performing a Navier-Stokes fluid dynamic problem, the user will need to set up a “Navier-Stokes” physical model. This is also the default physical model that appears in the GUI. When creating a new physical model, the user will select which type of physical model to create.

Each physical model consists of sub-sections that are displayed in the GUI as tabs. Some of the tabs inside a physical model will be similar in each physical model (like boundary conditions), while some will be unique to a model.

10.2 Parent/Child Relationships

A physical model will be either a parent or a child. A parent model is independent of all other physical models. A child on the other hand has a connection to one and only one parent physical model. A child physical model has the following properties.

- The number of solution variables must be the same as the parent. If the user wishes to change this number, it must be done in the parent physical model. Changes in the parent model that impact the number of solution variables will also be made in the child physical model.
- If a parent model can be applied to a particular zone, then so can all child models descendant from that parent model.
- If the parent physical model is cut or deleted, the next child model in the select list will become the parent model.
- A parent physical model can have any number of child models.
- A physical model is deemed a parent or child during the creation process. A child physical model has red text in the select list to distinguish it from the parent. See Sec. 10.4 on pg. 149 for more information on creating a model.

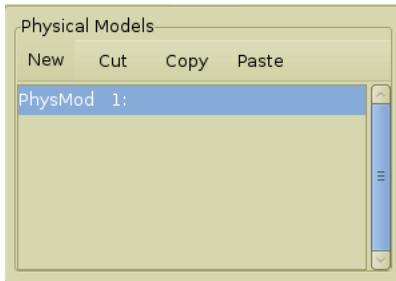


Figure 10.1: The physical model selection list.

The parent/child physical model relationship can be used to alter certain physical model parameters within a given solution process. For example, boundary conditions can be changed, or spatial accuracy can be changed from one run definition to another. However, it is not possible to change the independent variables being solved (*e.g.*, changing the number of species during a solution sequence).

10.3 Physical Models Select List

When the **Physical Models** button has been pressed, a physical model selection list will appear to the left of the physical model frame (lower left corner of the GUI). The physical model selection list is shown in Fig. 10.1 on pg. 149. The selection list allows the user to create, copy, or delete physical models. If more than one physical model exists, the user may switch the display between models by selecting the model name using the left mouse button. The physical model name can be changed or edited by double clicking on the model name, and entering new text using the keyboard. When finished, the new text is finalized by pressing **Return** (or **Enter**) from the keyboard.

The previous section explained the parent/child relationship that may exist between physical models. To help distinguish between parent and child models, the name of the child model will be red, while the parent model's text will be black. The select list will also be organized such that parent models will come before the child and child models will always follow immediately after a parent.

10.4 Creating a New Physical Model

A physical model can be created using either the **New** button or the **Paste** button. The **New** command will launch a dialog which is used to guide the user in creating a new model, while the **Paste** command will create a physical model by copying the clipboard contents.

If the clipboard is empty, a model cannot be created using **Paste**. If the paste command is used to create a new physical model, the created model will be a child of the existing parent model. If the parent model has been deleted, then the new physical model will be a parent.

When the **New** button is pressed, a create dialog will appear. The first step in creating a new model is to decide between the following choices.

Create a new Independent Physical Model This option allows the user to create a new physical model which is set to the default setting. The user will select from a list of physical models to create, as well as which zones to apply the physical model to. In terms of parent/child relationships, the new physical model will be a parent.

Create a new Physical Model Inheriting from Existing Instance This option allows the user to create a child physical model by making a copy of an existing model. The user will select from a list of existing physical models. The properties of the selected model will be applied to the new physical model. This option is very similar to the copy and paste sequence.

If a new independent model is chosen, the next step (using the **Forward** button) in the creation process is to select the model type. Once the model type is selected, the user must apply the model to a set of zones. To do this, the zones folder as it appears in the tree view is displayed for the user to browse through. The model may be applied to all the zones, a sub-set of zones, or to no zones at all. If no zones are applied, the user may apply the model to zones at a later time using options in the solution management section. (see Sec. 8.5 on pg. 96).

For the inheriting physical model option, the new model will be a child. Recall that every child has a parent assigned to it, which is selected from the next step after using the **Forward** button. Once a parent physical model is selected, the user must press the **Apply** button to finalize and create a new model. The new physical model will be a copy of the parent.

10.5 Copying or Cutting a Model

Physical models are placed in a clip-board using either the **Cut** or **Copy** buttons from the select list. The **Cut** command will delete the physical model from the list and place it in the clip-board (which is invisible to the user). The **Paste** command will allow you to recover the deleted physical model if this was the last item placed in the clip-board. The **Copy** command creates an identical copy of the selected physical model and places it in the clip-board. The **Copy** command is used for the sole purpose of pasting.

When cutting a physical model, *GASPEX* will first scan the GUI (which represents all the data in the input deck) to make sure that the physical model is not being used. For example, if the physical model is used to initialize a zone or used within a run definition, *GASPEX* will not allow the physical model to be cut. In order to cut a physical model that has “dependencies”, you must first remove the dependencies by changing each setting that uses the physical model. If dependencies exist, a pop-up dialog will appear when you try the **Cut** command, making you aware of the problem. For example, in Fig. 10.2 on pg. 151,

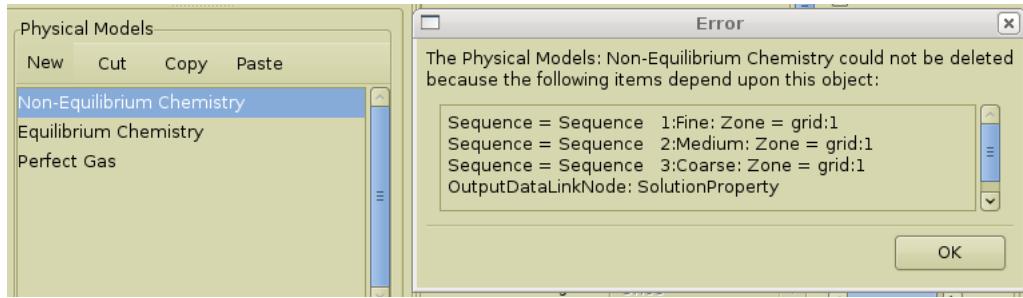


Figure 10.2: The pop-up dialog that appears when a physical model, which has dependencies, is cut.

the physical model named “Non-Equilibrium Chemistry” was attempted to be cut while dependencies existed. The pop-up dialog will list all dependencies. For this case, a number of dependencies were listed, including sequences and output nodes.

10.6 Summary Information

Every physical model will have a summary tab that provides the following information.

- The physical model type and a brief description of what it solves.
- If the physical model is a child, then the parent model is listed.
- The solution variables (also known as the primitive variables) are provided in a scrollable list.
- If a set of governing equations are solved, then the equations are provided in a list similar to the solution variables. Note that equations are presented for Lagrange based models.

10.7 Q Specification

One of the tabs found in most physical models is the **Q Spec**. A Q specification (**Q Spec**) is discreet set of values for the dependent variable vector, Q . Within the **Q Spec** window, the user will find not only values for the primitive solution variables, but also certain quantities that may be derived from the primitive variables. The user can create any number of **Q Spec** for a physical model. The **Q Spec** can then be used as references for boundary conditions, fortime integration, or output nodes. A **Q Spec** is also used to initialize the solution to a specific set of conditions.

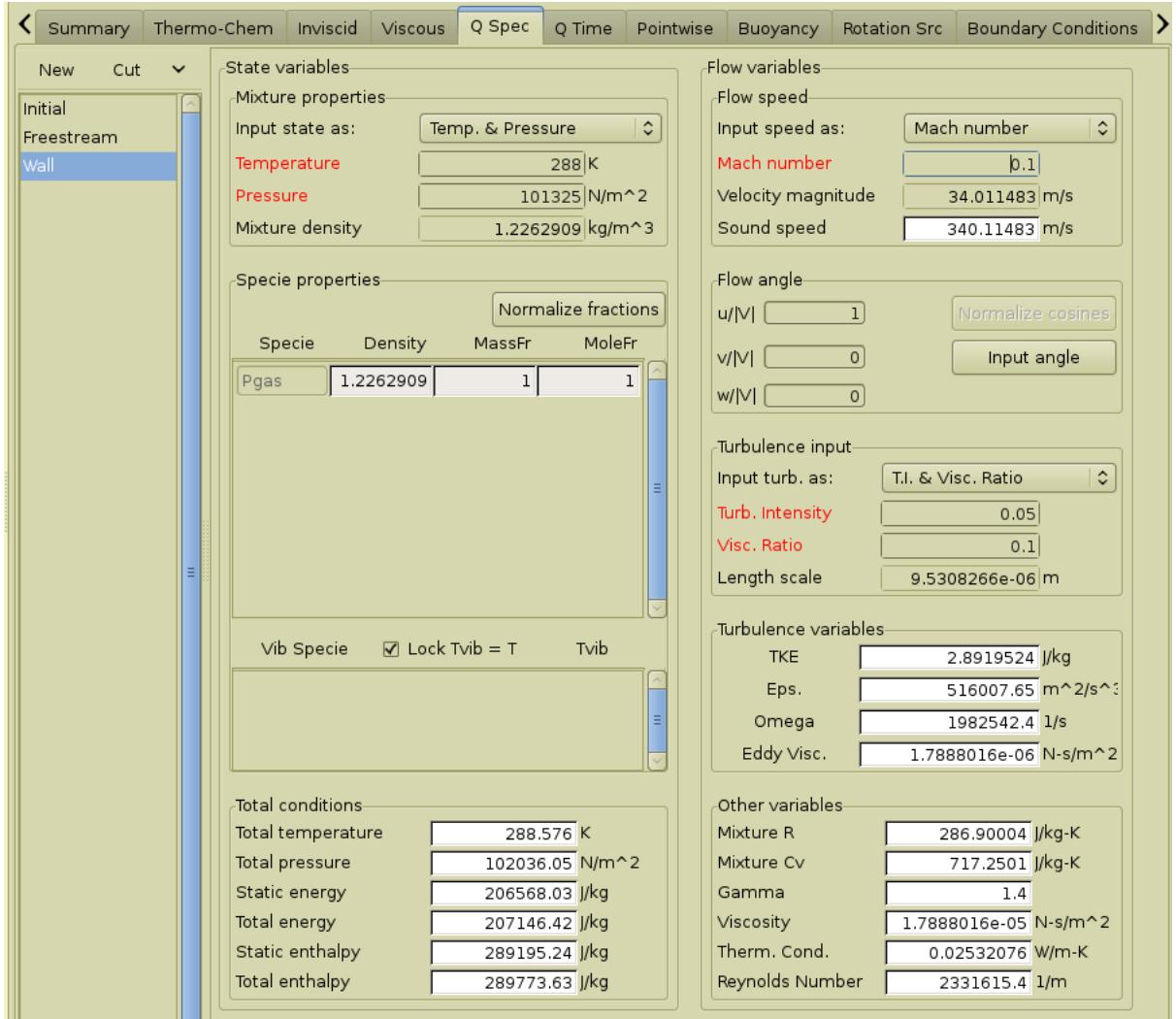


Figure 10.3: The Q Specification tab for fluid flow.

10.7.1 The **Q Spec** Select List

A **Q Spec** tab for the Navier-Stokes model is shown in Fig. 10.3 on pg. 152. To the left-hand side of the tab is located the **Q Spec** select list. Using the **New**, **Copy**, **Cut**, and **Paste** options, the user can manage **Q Spec** specifications. For example, a new **Q Spec** is added to the select list by selecting the **New** option. A **Q Spec** can be removed and placed on the clip board using the **Cut** option. **Paste** will create a new **Q Spec** identical to the one in the clip board, while **Copy** will make a copy of the **Q Spec** and place it in the clip board.

The details of each specific **Q Spec** is given in the corresponding physical model chapter. For example, details of the Navier-Stokes **Q Spec** can be found in Sec. 11.5 on pg. 195.

10.8 Pointwise Data

The **Pointwise** tab is used for importing and editing user specified surface data. This tab derives its name from the fact that the user must supply data for each face on a surface. This allows the user to define boundary values all along a surface. Below are examples of when the user may need to use pointwise data.

- To specify an inflow profile (ρ, u, v, w, p) at a boundary.
- To set a varying temperature profile along a solid wall surface.
- To model transition from laminar to turbulent flow along a surface where the user defines the transition location and/or length.

A sample pointwise data set is shown in Fig. 10.4 on pg. 154. Here, the pointwise data set has been set up to specify wall temperature along a surface (for a fluid flow simulation). The surface that the pointwise data is associated with is listed next to the **Surface Node** label, and the boundary condition folder (or surface family) that the pointwise surface belongs to is listed next to the **BC Folder** label.

Next to the surface description is the time information. Pointwise data may vary in time. The user may scroll through the time data by entering a new time in the **Display Time Step #** input box. The user may also use the up and down arrows to scroll the time list. The physical time associated with a data set is given by the **Physical Time** value.

If only one data set is specified in time, then that pointwise data set will be used for the entire solution, regardless of the physical time. When more than one time data set is specified, the data is linearly interpolated to the physical time of the computation. If the data is periodic in time, the user can enter one cycle of the data and then select the **Periodic Data** check box. This assumes that the last value in the data set is equal to the first value, such that the data repeats itself. The user may then use the pointwise data for any physical time in the computation.

For each pointwise data set, a table is displayed showing the data. The surface face indexes are listed to the left of the table, while the data itself is listed in row and column format. The top of the table has a header which identifies each piece of data.

Because pointwise data is associated with a surface, the data has a two-dimensional property to it. For example, a structured grid surface in the I direction (constant I value) will have faces with J and K indices. If data is read into the table in the wrong index order, the user may reverse the data using the **Reverse Data Order** button.

The actual data can be stored in the input deck file or in a user created file. The storage location is set when creating the pointwise data set (see the next section). If the data is stored in the input deck file, a text string will appear above the data table showing guilabelData stored in .gsp file. Otherwise, an input text box with the file name will appear.

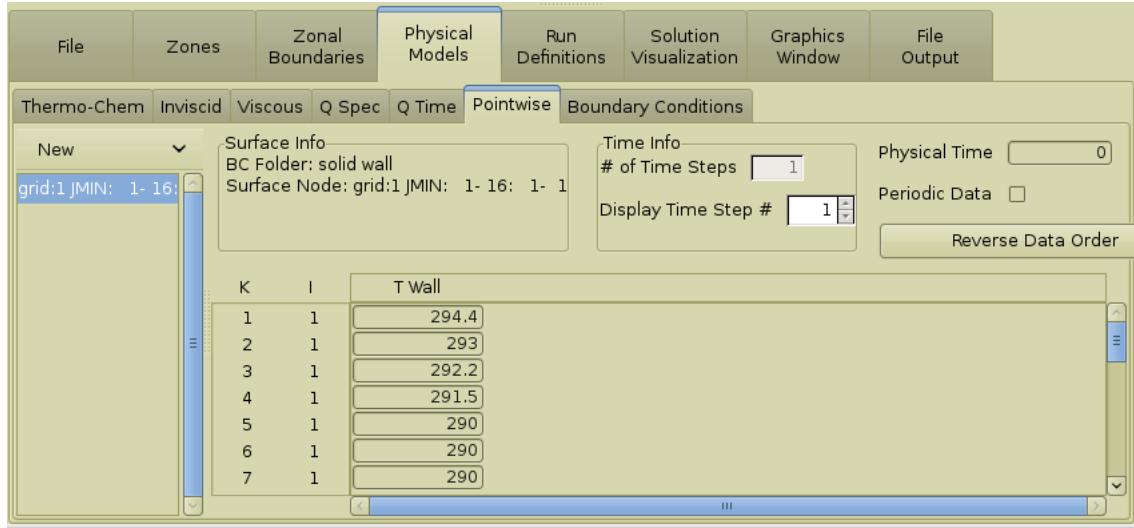


Figure 10.4: A sample pointwise data window for wall temperature.

10.8.1 Creating a Pointwise Data Set

Initially there will be no pointwise data sets. The user must create a data set for each surface that requires pointwise data. A surface can only have one corresponding pointwise data set in each physical model. To create pointwise data for a surface, follow the steps below.

1. Select a surface from the tree view. For our sample pointwise data set (Fig. 10.4 on pg. 154), the surface was selected as shown in Fig. 10.5 on pg. 155. Note that a surface must be selected before you are allowed to create a data set.
2. Press the **New** button. Pressing the **New** button will bring up a window similar to the one shown in Fig. 10.6 on pg. 156. If a pointwise data set for a surface already exists, you will not be able to create a new data set. In this case, the user must edit the existing pointwise data or delete (cut) it.
3. Follow the steps given in the dialog window to create a pointwise data set. The first step has you select the type of data you wish to use. Here, your choices (in the case of a Navier-Stokes physical model) are Primitive Flow Variables, Wall Temperature, Intermittency, Primitive Flow Variables & Intermittency, and Wall Temperature & Intermittency. Each of these data types are described in Sec. 11.7.1 on pg. 201.
4. For the second step, select the number of time steps that the data will have. For each time step there will need to be a complete set of data for the surface. For steady state calculations, you will only need to specify one time step (default value).
5. Next, decide whether the data is periodic or not. If the data is periodic in time, select that option.



Figure 10.5: You must select a surface from the tree view before creating a new pointwise data set.

6. Finally, decide how you want to initialize the data and where to store it. Your choices for initializing the data are: read data from a file or use a **Q Spec** to set all the values. If you read data from a file, first select the units type and then press either **Read Primitive/Temp Data** or **Read Intermittency Data** (in the case of Navier-Stokes data). See Sec. 10.8.3 on pg. 155 for more information on entering data using a file. If you want to initialize the data use a **Q Spec**, select the **Q Spec** and press **Initialize Using QSpec**.
7. After the data is read in from a file or initialized using a **Q Spec**, the window will disappear and the pointwise data will appear in the table.

While the new pointwise data set window is opened (see Fig. 10.6 on pg. 156), the user may export the X, Y, Z face center values from the surface using the **Export XYZ Data** button. This information can sometimes be helpful if the user needs to know the face center location of each face or the order in which data is imported.

10.8.2 Pointwise Data Types

There are a number of data types associated with a pointwise data set. The data types will be a function of which physical model is being setup. See the pointwise description for each physical model for specific details of the data types.

10.8.3 Entering Pointwise Data From a File

The pointwise data is created by initializing to a **Q Spec** or by reading the data from a file. With either method if data is stored in the input deck file, the user can change values by entering data directly into the data table using the keyboard. This may be convenient if

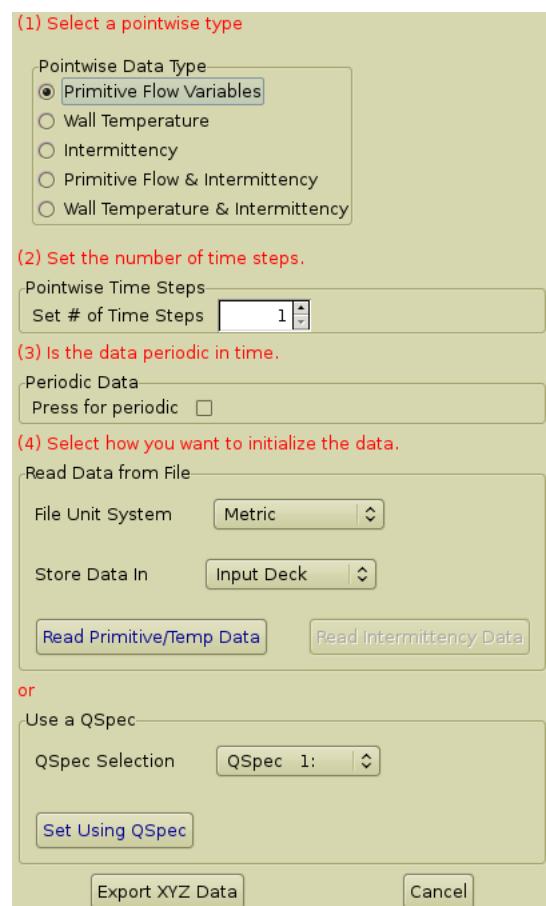


Figure 10.6: To create a new pointwise data set, press the **New** button and this window will appear. Follow the four steps to create the pointwise data.

only a few values are different from the **Q Spec**. But this may be time consuming if you have a surface with many faces. Most of the time data will be read in using a file.

If you wish to read Navier-Stokes data from a file, be sure to select either the **Read Primitive/Temp Data** button or the **Read Intermittency Data**. Reading data for other physical model types will be similar. When one of these buttons is selected, a file selection dialog box will appear allowing you to choose a file for opening. The pointwise data file must be in ASCII format, and contain data in column format. There should be no header lines in the file, only data. The first line of data should correspond to the physical time. After that, the number of lines of data should be the same as the number of faces on the surface associated with the pointwise data. This is then repeated for the number of time steps specified when creating a new data set. A sample pointwise data file for the primitive flow variable type (ρ, u, v, w, p) is given below.

```
0.0
1.226 100 0 0 101325
1.226 105 0 0 101325
1.226 110 0 0 101325
1.226 115 0 0 101325
1.226 120 0 0 101325
1.226 125 0 0 101325
1.226 130 0 0 101325
1.226 135 0 0 101325
1.226 140 0 0 101325
1.226 145 0 0 101325
1.226 150 0 0 101325
1.226 155 0 0 101325
1.226 160 0 0 101325
1.226 165 0 0 101325
```

GASPE will read up to 250 characters per data line. If you need more than this max, please contact AeroSoft. For an I constant surface, the j index is read first, followed by k. For a J constant surface, the I index is read first, followed by K. And for a K surface, the I index is read first, with J following. Once the data is read in, a table will appear in the GUI showing the data with the corresponding indices.

If the data to be read in from a file is not in the above order, the user may reverse the index order after reading in the data. This is done using the **Reverse Data Order** button. Pressing this button will assume that the file data was ordered incorrectly and will reverse the index order. Again, this should be done after the data has been read in from a file. Each time the button is pressed, the index ordering will be switched, such that pressing it twice will return the data back to the original setting. Note that this is only effective if data is stored in the input deck file and not in an external file.

For a fluid flow (Navier-Stokes) physical model, selecting the **Read Primitive/Temp Data** button will read either primitive or temperature data. What is read will depend on what

you have selected in the (1) **Select a pointwise type** selection (pointwise new window). If you need to read in both intermittency data and primitive/temperature data, both buttons will be selectable (otherwise only what you need to read in will be selectable). In this case, you will need to have pressed both buttons before the window will close (which is done automatically).

If you insert data from the keyboard, the units of the data should be consistent with the input deck units as set in the **File** frame (see Chap. 5 beginning on pg. 69). When reading in data from a file, the units type should be selected first before opening the file selection dialog. The values read in from a file will be converted, if necessary, to the input deck unit system.

10.8.4 Connection to Boundary Conditions

Now your next question may be “how do I apply a pointwise data set once I have created it?” Most of the time pointwise data is used by the boundary conditions, so this is where the pointwise data is selected for use. Inside the **Boundary Conditions** tab will be a table used to specify boundary conditions. For each surface, a boundary condition is set. If a boundary condition needs outside data, the **Q Source** option menu will be selectable. The third selection in this option menu is labeled **Pointwise**. If each surface in the surface family has pointwise data available, then setting the **Q Source** to this option will tell the boundary condition to use the pointwise data for any information that it needs.

Pointwise data can be used for any boundary condition that needs a **Q Source**. For those boundary conditions that do require a **Q Source**, not all the pointwise data may be needed. For example, the Navier-Stokes **Pback Subsonic Outflow** condition requires a **Q Source** for the pressure. This condition sets the back pressure and takes the pressure value from the **Q Source**. If the **Q Source** is set to **Pointwise**, only the pressure from the pointwise data is used. Note that using this combination allows the user to set a spatially, and, potentially, temporally varying back pressure.

If the pointwise data contains intermittency data, then the intermittency data will automatically be applied in the flow solver. The user does not need to select pointwise data for the boundary condition. But if primitive or temperature data is used for pointwise data, the data will not be used unless the boundary condition is set up to use it.

10.8.5 Pointwise Select List

The user may display different pointwise data sets using the pointwise select list. The select list for the pointwise data has similar functionality as other select lists in *GASPE*. For example, selecting an item in the list will change the display to reflect that item’s data. A data set can be removed using the **Cut** button.

The **Edit** button will bring up a window allowing the user to change the pointwise data values. For example, with edit the user may overwrite the pointwise data by either reading in data from a file or by setting the data using a **Q Spec**. The edit function does not allow

changes to the data type or the number of time steps. To change these parameters, the user will need to remove the pointwise data (using the **Cut** button) and create a new pointwise data set.

Remember, only one pointwise specification is allowed per surface node for each physical model. Or in other words, each surface can have only one pointwise data set.

10.9 Boundary Conditions

The **Boundary Conditions** frame allows the user to assign boundary conditions to surface families. The specific boundary conditions are unique to the type of physical model. A description of each boundary condition is given in the corresponding physical model chapters.

The boundary condition frame has two tabs: BC Table and Special BC Data. The **BC Table** is described next and is used to assign the boundary conditions to surface families. The **Special BC Data** contains boundary condition specific inputs. The Special BC Data tab may or may not exist depending on the physical model (only appears if boundary conditions need it).

10.9.1 Boundary Condition Selection

The **BC Table** tab presents information in table format (see Fig. 10.7 on pg. 160), where each row corresponds to a surface family. A surface family is placed in the table by selecting the surface folder in the tree view (these are folders under “Surfaces”) and selecting the **Edit→Toggle Folder BC On/Off** menu option (the edit menu is accessed using the right mouse button). There are special zonal boundary (ZB) folders that have this same property. In cases where a zonal boundary folder has the BC setting toggled on, the ZB surfaces are overridden by the boundary condition.

For each surface family in the table you will need to assign a boundary condition. This is done by pressing the option menu and selecting the appropriate condition for the surface family.

10.9.2 Split Versus Full Flux Setting

Boundary conditions can be applied as either a split flux or a full flux. The selection between a split flux and a full flux is done using the check box under the **Split** header. The difference between split and full flux stems from a design feature of any flux-splitting scheme. Numerical flux functions are functions of two states, typically called a left and right state (or a plus and minus state). If the two states are equal, then the split flux (*i.e.*, the numerical flux function) equals the full flux as it appears in the governing equations.

What does that mean in terms of the boundary conditions? If the boundary condition is set to a full flux, then the boundary state is calculated and the flux at the cell face is

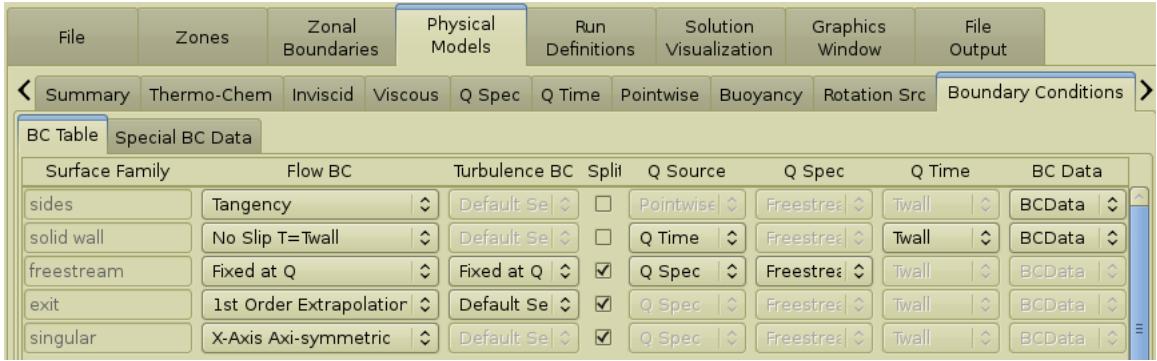


Figure 10.7: A sample boundary condition set up using **Q Spec** for the **Fixed at Q** BC and **Q Time** for the **No Slip T=Twall** condition.

computed based solely on the boundary value. This implies that the boundary face lies on a physical surface. The split flux condition designates that the flux function should be evaluated with different left and right states. One of these states is the boundary condition and the other comes from the interpolation (*e.g.*, for a first-order scheme the other state would be the first interior cell's state). Split flux boundary types imply that the boundary state is actually a ghost cell.

*GAS*Pex will automatically select the appropriate flux setting (full or split) with each boundary condition selection. The user may then override the default setting by changing the **Split** check box value. When the box is pressed, the boundary condition enforces a split flux.

10.9.3 Q Source Settings

Some of the boundary conditions that are available in *GAS*Pex may require user specified parameters in order to set the boundary values. For example, the **Fixed at Q** condition sets all the boundary values to a given flow state. The letter "Q" here represents a list of state parameters (*e.g.*, density, velocity, pressure, temperature). The conditions which are used by a boundary condition (if needed) can come from one of three sources: a **Q Spec**, a **Q Time**, or **Pointwise** data. A Q specification (**Q Spec**) defines the flow variables at a state (independent of time), while a Q time (**Q Time**) defines either the primitive variables, wall temperature, or pressure with respect to time. Pointwise data is user specified data values for a surface and can be a function of time as well. The **Q Spec**, **Q Time**, and **Pointwise** data are discussed in Sec. 10.7 on pg. 151, Sec. 11.6 on pg. 197, and Sec. 10.8 on pg. 153 respectively.

If a boundary condition is chosen that does not need a "Q Source", then the Q source selection will be disabled. For example, if the **1st Order Extrapolation** boundary condition is chosen, the **Q Source** option menu will not be selectable since this condition sets the boundary values based on the interior solution. If a boundary condition does require a Q source, then the option menu will list the possible choices. The **Q Time** and **Pointwise**

options will only be available if they exist in the physical model. In this way the user cannot select a Q source that does not exist.

10.9.4 Q Spec and Q Time Settings

The **Q Spec** option menu displays all of the Q specifications that have been created for the physical model. In a similar manner, the **Q Time** (used for Navier-Stokes) option menu allows the user to select any Q time selections that have been defined. If the **Q Source** option menu is set to **Q Spec**, only the **Q Spec** option menu will be enabled. The same is true for **Q Time**. If **Q Time** is selected for the Q source, then only the **Q Time** menu is enabled. If **Pointwise** is selected for the Q source, then neither the **Q Spec** or **Q Time** option menus will be active for the user.

For example, in Fig. 10.7 on pg. 160, there are a total of five surface families in the table with boundary condition assignments. Of the boundary conditions used here, only the **No Slip T=Twall** and **Fixed at Q** selections require flow conditions to properly set the boundary values. This is clearly seen from the Q Source option menus. With the way things are set up in the figure, the **Fixed at Q** boundary condition will use the “Freestream” **Q Spec** to set the boundary values, while the **No Slip T=Twall** boundary condition will use the time varying “*Twall*” **Q Time** flow conditions.

10.9.5 BC Data Settings

In many cases, boundary conditions will need input data set by the user (such as inflow conditions or back pressure). In the case where additional information is needed, this can take the form of solution variables, in which case the “**Q Source**” is used, or it may be a special input unique to the boundary condition. In the latter case, the **BC Data** is used to assign which special BC data entry to use with the corresponding boundary condition.

The special data (if needed), is located in the **Special BC Data** tab under the Boundary Conditions frame. If a physical model has boundary conditions that do not need any special data inputs, then the special BC data tab will not appear.

10.10 Changing Solution Variables

When performing calculations using *GASPer*, the primitive variables are stored at each cell center and used to compute all other flow variables. It is also the primitive variables that are stored in the binary solution file and used for solution restarts. In most cases, a primitive variable corresponds to an equation that must be solved in order to reach the final solution. Increasing the number of primitive variables in a computation means increasing the number of equations that must be solved, thus increasing the computational time.

The primitive variables for a Navier-Stokes solution may consist of the specie densities, velocity components, non-equilibrium vibrational energies, pressure, temperature, and various turbulence parameters like the turbulent kinetic energy. Not all of these variables will

be present all of the time, but which primitive variables exist will depend on which physical models are used to initialize the solution and how the physical model is set up. In vector form the primitive variables for a compressible Navier-Stokes simulation using the *k*-omega turbulence model can be written as:

$$\mathbf{q} = \begin{Bmatrix} \rho_1 \\ \rho_2 \\ \vdots \\ \rho_N \\ u \\ v \\ w \\ e_{n_1} \\ e_{n_2} \\ \vdots \\ e_{n_M} \\ p \\ k \\ \omega \end{Bmatrix} .$$

where N represents the number of species and M the number of non-equilibrium vibrational energies. For this particular example the turbulence variables come from the *k*-omega equation where k represents the turbulent kinetic energy and ω the turbulence frequency.

At some time while running *GASPer*, you may need to change one or more of the primitive variables while performing a calculation. For example, you may begin a problem assuming an inviscid flow, and then add a two-equation turbulence model to simulate the effects of turbulence. This operation requires adding two variables to the solution file. Other examples may be adding non-equilibrium vibrational energies or changing the chemistry model where the number of species increase or decrease. These operations must be performed using the *GASPer* GUI.

The following section will discuss how the solution can be changed after a problem has been started. While setting up the input deck for the first time, the solution does not exist so any changes can be made without worry. But after a solution has been written to the binary solution file, care must be taken in order to correctly modify the solution.

10.10.1 General Procedure for Changing the Solution

Since there can be multiple physical models and multiple zones for a problem, *GASPer* will perform several checks when a change is made to the physical model which requires the solution to change. First, *GASPer* will check to see if the physical model is used to initialize a zone. If so, then *GASPer* will check to see if a solution exists for that zone. If both of these things are true for the physical model, then *GASPer* will start the process of converting the solution.

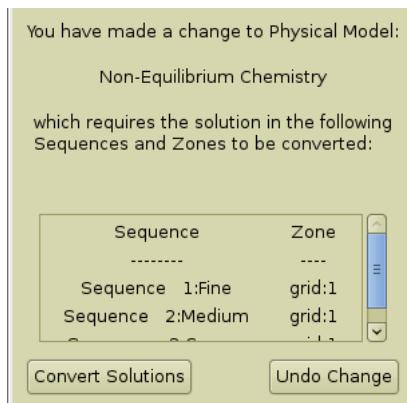


Figure 10.8: The solution conversion dialog that appears when the physical model has been changed in such a way that the solution must be converted.

GASPex has a two step process in converting the solution. The first step is to inform the user of the need to convert the solution, which is done by displaying a dialog window. An example dialog window is shown in Fig. 10.8 on pg. 163. In the figure the physical model, sequence, and zone that are involved in the solution conversion are listed. There may be multiple sequences and zones involved, but for this example there is one of each. The second step requires the user to reply to the dialog by pressing either the **Convert Solutions** button or the **Undo Change** button.

If you decide not to change the solution, then press the **Undo Change** button. This will return things back to the original setting, causing *GASPex* to undo the last change to the input deck.

By pressing **Convert Solutions** you give *GASPex* permission to go ahead with the solution conversion process. The dialog will disappear and the solution will be converted.

Chapter 11

Physical Models: Navier-Stokes

11.1 Introduction

The Navier-Stokes physical model is used to simulate fluid dynamic problems in which the Navier-Stokes equations are solved. The Navier-Stokes equations consist of the conservation equations of mass, momentum and energy. Additional equations may also be solved for including, but not limited to, turbulence and non-equilibrium vibrational energy. For more information on the Navier-Stokes equations, the user is referred to the *GASpex* technical manual.

The physical model contains a number of tabs for inserting data (see Fig. 11.1 on pg. 165 for a view of the Navier-Stokes tabs). Certain tabs are required in order to run a fluid dynamic problem (*e.g.*, Thermo-Chem, Inviscid, Q Spec and Boundary Conditions) while others are not. Each tab will be explained in more detail in the sections that follow.

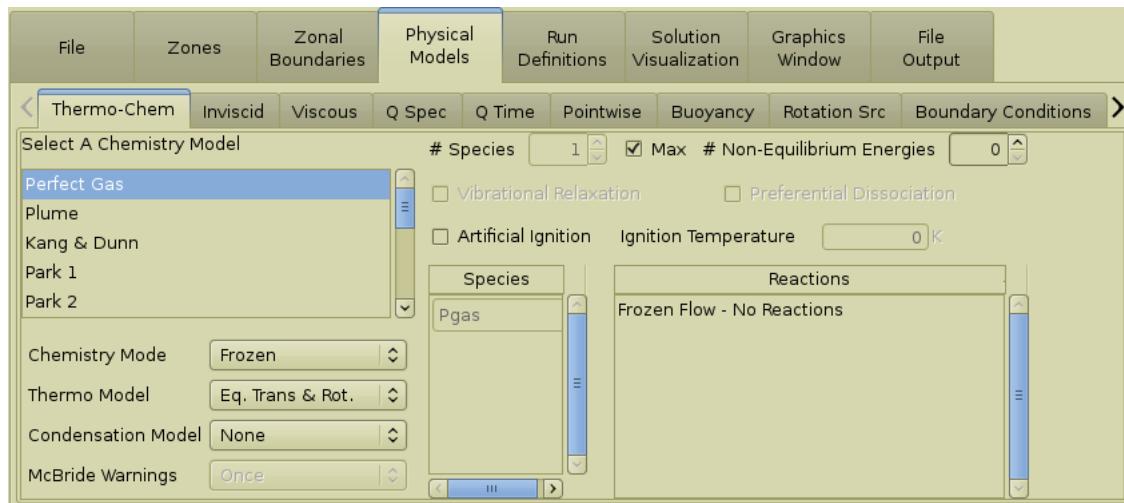


Figure 11.1: The Navier-Stokes physical model frame.

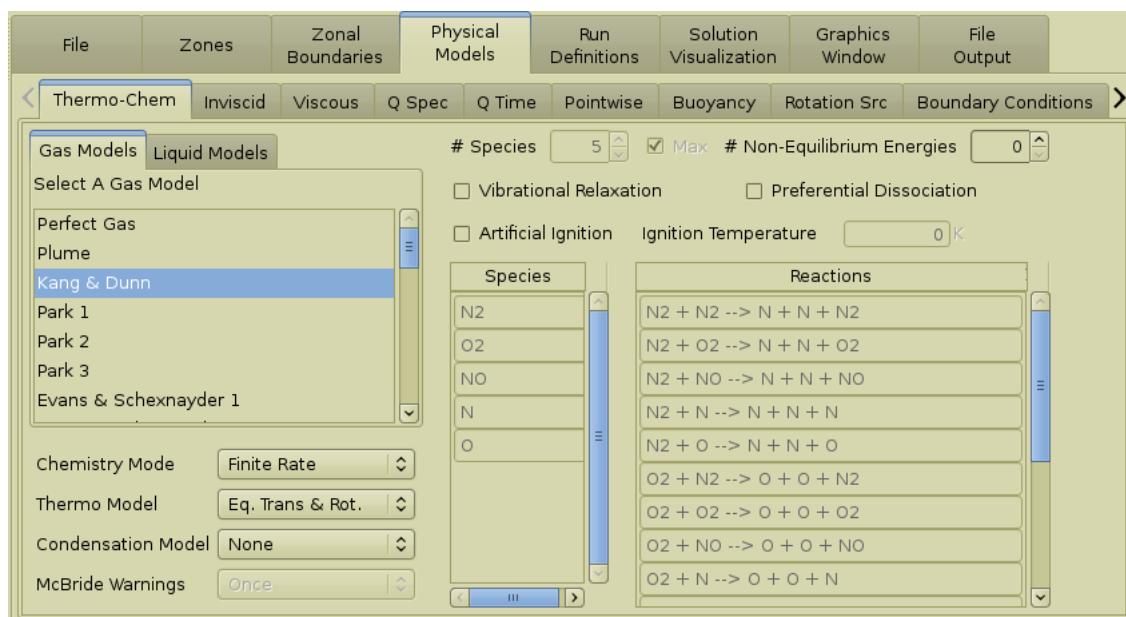


Figure 11.2: The Thermo-Chem tab.

11.2 Thermodynamics and Chemistry

The thermodynamic and chemical modeling of the flow field is controlled in the **Thermo-Chem** tab shown in Fig. 11.2 on pg. 166. The data from this section is interconnected with the **Q Spec** tab. The species selected from the chemistry model are displayed in both the **Thermo-Chem** and **Q Spec** tabs. Changing the chemistry model will have a large effect on the **Q Spec** tab since it can change the number of species in the solution.

11.2.1 Chemistry Model Scroll List

The available chemistry models are displayed in scroll lists depending on the model type. Currently either gas or liquid models are supported with the Navier-Stokes solver. There can only be one chemistry model for each physical model, which is selected by picking one of the supported models in either the gas or liquid model list. The list of models come from the database file (default is installed as `aerosoft/share/thermochemdb.tcd`). New models can be created using the *GAS*Pex chemical database GUI.

If a liquid model is selected, *GAS*Pex will automatically adjust the following in the physical model:

- The chemistry mode is set to frozen, which means there will be no reactions taking place. Chemical reactions are not allowed for liquid flows at this time.
- The thermodynamic model is set to a valid option for liquid models (equilibrium liquid).

- The number of species is fixed to one. Only one liquid species is supported when using a liquid model with the Navier-Stokes solver.
- The inviscid (convective) flux is set to Artificial Compressibility in order to solve the incompressible Navier-Stokes equations.
- If solving for laminar or turbulent flow, the viscosity and conductivity models are set to options specific for liquids.

11.2.2 Number of Species in the Model

Most chemistry models contain more than one specie. The number of species in the model is shown in the **# of Species** type-in box. In most cases, all the species in the model should be included in the simulation. This is enforced if the **Max** check box is selected. If the number of species in the physical model is deliberately set to a number less than the total number in the chemistry model, then the species ordered first in the list are included in the physical model. For example, the **Kang & Dunn** chemistry model contains five species: N_2 , O_2 , NO , N , and O . If the number of species is set to 3, then only the first three species in the list will be modeled in the flow (*i.e.*, N_2 , O_2 , NO).

The **Species** scroll list displays the selected species. The species in the scroll list can be changed by either selecting a new chemistry model or by changing the **# of Species** type-in box. The scroll list itself cannot be edited. Species can be created or modified using the **GASPer** chemical database GUI.

For liquid models, chemical reactions are not supported. Because of this limitation, a single liquid species can be used to represent the fluid mixture and the number of species should always be one.

11.2.3 Charge Balance

For flows in which ionization occur, the chemistry model may contain ions in order to capture this physical phenomena. One of the products of ionization is free electrons (denoted by the $e-$ specie name). When ions are present in the chemistry model along with the electron specie, the **Charge Balance** option will be available. When this option is selected, the assumption is made that charge equilibrium will occur within each cell. This charge equilibrium will be imposed by adjusting the electron specie density at the end of each iteration cycle.

GASPer assumes that the electron specie will always be the last entry in the species list. So when creating a chemistry model with an electron, ensure that it is last in the specie list.

11.2.4 Artificial Ignition

In cases where a combustion process is being simulated, the user may want to model fuel ignition and burn. This may present a problem if the flow temperatures are not high enough to cause combustion to start. In these instances, the user may opt to use the **Artificial Ignition** option. When this option is selected (in conjunction with finite-rate or equilibrium chemistry), the computed reaction rates are no longer based on the flow temperature, but on the **Ignition Temperature**. This option should only be used for a set number of cycles in order to get the combustion process started. After the reactions start, the artificial ignition should be turned off and the ongoing reactions should sustain themselves. Long term use of this option may cause numerical and physical instabilities.

11.2.5 Chemistry Mode

As the temperature of a non-ideal gas increases (*e.g.*, because of post-shock compression or viscous dissipation) two effects become important. First the vibrational mode of any polyatomic molecule becomes excited and contributes to the internal energy and thus the specific heat. Also, as molecules begin colliding at a more frequent rate and at higher energies, chemical reactions occur. The first effect is thermodynamic and the second chemical. In *GAS*Pex, the thermodynamic model describes the internal-energy state of the molecules in the flow and is selected using the **Thermo Model** option menu (see Sec. 11.2.6 on pg. 168). The chemistry model describes which molecules are present and how quickly they react together as a function of temperature. This data is largely empirical and comes from various sources of experimental literature.

For air at atmospheric pressure, dissociation of diatomic oxygen occurs around $T = 2000$ K with complete dissociation occurring around $T = 4000$ K. Diatomic nitrogen begins to dissociate at $T = 4000$ K and ionization occurs at temperatures above $T = 9000$ K. These reactions can occur very slowly, very quickly or at a finite rate.

*GAS*Pex can simulate frozen, equilibrium and finite-rate flow by selecting the appropriate choice in the **Chemistry Mode** option menu. Frozen flow calculations can be performed for any chemistry model and specie number. To use the **Equilibrium** or **Finite Rate** options, the number of species must equal the maximum number for the chemistry model. An additional requirement for finite rate chemistry is the reaction data, available via the chemical database. For equilibrium chemistry the reaction rates are set to infinity.

11.2.6 Thermodynamics Model

Thermodynamic models may be selected with the **Thermo Model** option menu. At temperatures away from absolute zero, we can justifiably assume that the molecules in a gas are in translational and rotational equilibrium. For air, this assumption is valid up to around $T = 800$ K. When the gas temperature increases to significantly high temperatures, real-gas effects appear as the molecular vibrational mode becomes excited. *GAS*Pex gives the user several thermodynamic models to choose from in the **Thermo Model** option menu. The

available models are described below (see the *GASpex* technical manual for a more in-depth discussion of each).

Equilibrium Statistical Mechanics With the **Eq. Stat. Mechanics** option, species are assumed in translational, rotational and vibrational equilibrium. The internal energy is calculated based on equilibrium statistical mechanics [1].

Gordon/McBride Curve Fits The species curve fits from Gordon and McBride [2] are used to compute Cp/R , $h_0/(RT)$ and s/R with the **Gordon/McBride** option. Energy modes are modeled implicitly through the curve-fit data as a function of temperature. This curve fit supports up to four temperature ranges. Curve fit data must be specified for each specie before using this option. This can be done using the *GASpex* chemical database GUI. If the flow temperature goes below or above the curve fit range, the temperature used to evaluate the curve fit will be clipped to the lower or upper temperatures accordingly to prevent extrapolation.

In Version 4.1, the nomenclature of the LeRC curve fits was changed to reflect the authors of the papers that originally presented the curve fits. Therefore, what used to be called the LeRC curve fits are now referred to as the Gordon & McBride curve fits. The name change reflects the change from NASA Lewis Research Center to NASA Glenn Research Center. The curve fits come from what is now called the NASA Glenn Chemical Equilibrium Application Program (CEA). More information on this program can be found at <http://www.grc.nasa.gov/WWW/CEAWeb>.

Equilibrium Translation and Rotation This is the simplest modeling option (**Eq. Trans. & Rot.**) which assumes each species is in translational and rotational equilibrium. No vibrational energy contributions are included. This is valid at low temperatures.

Equilibrium Liquid This option is used for liquid models and computes the internal energy using the specific heat data from the database.

11.2.7 Condensation Model

GASpex has the ability to model water condensation when a valid chemistry model is selected that supports condensation. To support condensation, the chemistry model must have water species defined for each water droplet size to be solved for. The water droplet species have the follow syntax, `H2O_[r=size]` where size is the droplet radius in meters. For example, a model may have a number of droplet species ranging from `H2O_[r=1.0e-04]` to `H2O_[r=1.0e-10]`. The water condensation model is currently coded for H_2O only.

For more information on the water condensation model in *GASpex*, please refer to the following references: Eppard *et al.* [3], Masuda *et al.* [4], Perrell *et al.* [5], and Perrell *et al.* [6]. An example test case is also available upon request.

11.2.8 McBride Warnings

When using the Gordon/McBride thermodynamics model, there are possible issues that might arise. One of the main issues that may occur is when the flow temperature goes outside the curve fit range. When this happens, the temperature used to evaluate the curve fit will be clipped to the lower or upper temperature range accordingly. The user may want to be alerted when issues with the curve fit occur. The **McBride Warnings** provides the user with a number of communication options when issues occur. Setting the warning to “Once” will print the first warning to output (*i.e.*, screen, shell, queue output), but suppress the rest. Using “Some” will display the first 100 warnings and “All” will display all warnings. If no warnings are desired, then select “None”. Remember, this option only impacts how warnings are displayed, but will not affect how the curve fit data is evaluated by the solver.

11.2.9 Available Chemistry Models With *GASPer*

GASPer comes with a chemical database filled with various chemistry models. The user can add or modify these models using the *GASPer* chemical database GUI. The database GUI can be opened separately or from within the input deck GUI (see Sec. 5.1 on pg. 69 for more information). A brief description of the gas models, as well as the number of species and reactions, are given next.

Simple Air Chemistry Models

The most commonly used models for air calculations at relatively low temperatures are

Perfect Gas Air modeled as a perfect gas. (1 species, 0 reactions)

Kang & Dunn Kang, et al. air model [7]. (5 species: N_2 , O_2 , NO , N , O and 17 reactions)

The best chemistry model for a simulation depends on both the gas composition and the temperatures expected. Fig. 11.3 on pg. 171 shows the composition of equilibrium air over a range of temperature.

Park’s Air Chemistry Models

The air chemistry models of Chul Park at NASA Ames Research Center have been implemented into *GASPer* and their names are

Park 1 Simplest Park air model which neglects ionization reactions [8]. (5 species: N_2 , O_2 , NO , N , O and 17 reactions)

Park 2 Onset-of-ionization Park air model [8]. (7 species: N_2 , O_2 , NO , NO^+ , N , O , e^- and 18 reactions)

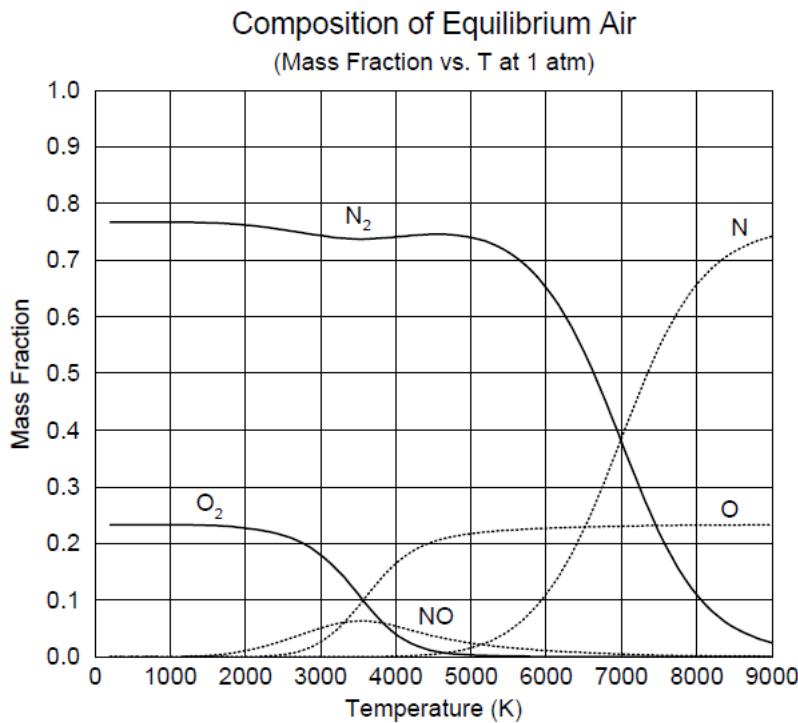


Figure 11.3: The equilibrium composition of air at standard pressure.

Park 3 Park's complete air model with full species list and reaction set [8]. (11 species: N_2 , O_2 , NO , N_2^+ , O_2^+ , NO^+ , N , O , N^+ , O^+ , e^- and 47 reactions)

Park 4 Park's updated air model with complete species list and reaction set [9]. (11 species: N_2 , O_2 , NO , N_2^+ , O_2^+ , NO^+ , N , O , N^+ , O^+ , e^- and 49 reactions)

Park 5 Park's updated air model with complete species list and a reduced reaction set [9]. (11 species: N_2 , O_2 , NO , N_2^+ , O_2^+ , NO^+ , N , O , N^+ , O^+ , e^- and 38 reactions)

Park 6 A Park air model similar to Park 2 but with reaction rates from reference [9]. (7 species: N_2 , O_2 , NO , NO^+ , N , O , e^- and 21 reactions)

The last three models are updated versions of Park's original air model. The new models include more coefficients (5 instead of 3) for the calculation of the equilibrium constant and upgraded forward reaction rates. The first two Park air models are subsets in species of the complete model (**Park 3**) with the first being applicable at low temperatures. The safest use of the second Park air model is to run the complete model and determine the significance of the neglected species (*i.e.*, N_2^+ , O_2^+ , N^+ , and O^+) which ionize last.

Evans and Schexnayder's Hydrogen-Air Models

Some of the most successful Hydrogen-Air models in *GASPex* are those attributed to John Evans and Charles Schexnayder [10] of NASA Langley Research Center. Their chemistry model names are

Evans & Schexnayder 1 Eight-reaction subset of Evans and Schexnayder Hydrogen-Air model [10]. (7 species: N_2 , O_2 , H_2 , OH , H_2O , O , H and 8 reactions)

Evans & Schexnayder 1X A more complete reaction set of the previous model. (7 species: N_2 , O_2 , H_2 , OH , H_2O , O , H and 32 reactions)

Evans & Schexnayder 2 The full species model Evans and Schexnayder Hydrogen-Air model [10] with a reduced reaction set. (12 species: N_2 , O_2 , H_2 , NO , OH , NO_2 , HO_2 , H_2O , HNO_2 , N , O , H and 25 reactions)

Evans & Schexnayder 2X A more complete reaction set of the previous model. (12 species: N_2 , O_2 , H_2 , NO , OH , NO_2 , HO_2 , H_2O , HNO_2 , N , O , H and 102 reactions)

Drummond's Hydrogen-Air Models

Several of Drummond's Hydrogen-Air models [11, 12, 13] have been implemented in *GASPex* and are

Drummond 1 7-reaction subset of Drummond Hydrogen-Air model [11]. (7 species: N_2 , O_2 , H_2 , OH , H_2O , O , H and 7 reactions)

Drummond 1X A more complete reaction set of the previous Drummond 1 model. (7 species: N_2 , O_2 , H_2 , OH , H_2O , O , H and 19 reactions)

Drummond 2 Complete Drummond Hydrogen-Air model [11]. (9 species: N_2 , O_2 , H_2 , OH , HO_2 , H_2O , H_2O_2 , O , H and 18 reactions)

Drummond 2X A more complete reaction set of the previous Drummond 2 model. (9 species: N_2 , O_2 , H_2 , OH , HO_2 , H_2O , H_2O_2 , O , H and 50 reactions)

The Baulch Chemistry Models

The Baulch models [14, 15] include the trace elements of air, namely Argon and carbon dioxide, as well as a hydrogen fuel. The models are

Baulch 1 First Baulch model [14]. (18 species: CO , H_2 , N_2 , NO , O_2 , OH , CO_2 , H_2O , Ar , H , N , O , CH_4 , HCN , NH , NH_3 , N_2O , HCO and 16 reactions)

Baulch 2 Second Baulch model [15]. (18 species: CO , H_2 , N_2 , NO , O_2 , OH , CO_2 , H_2O , Ar , H , N , O , CH_4 , HCN , NH , NH_3 , N_2O , HCO and 93 reactions)

Chemistry Models for Argon

Three Argon models [16, 17] are available in *GASPE* and go by the following names.

Glass Glass Argon Plasma model [16]. (3 species: Ar, Ar^+, e^- and 2 reactions)

Igra Igra Argon Plasma model [17]. (3 species: Ar, Ar^+, e^- and 1 reaction)

Argon-Freon Plasma Argon-Freon model. (4 species: Ar, CCl_2F_2, N_2, O_2 and no reactions)

Chemistry Models for Ethylene

The Baurle Ethylene model [18] is a three-step chemistry model for modeling ethylene (C_2H_4) in air. The model consists of 7 species and 3 reactions. The model species are: $C_2H_4, O_2, N_2, H_2, CO, CO_2, \text{ and } H_2O$.

11.3 Inviscid Information

The **Inviscid** tab is shown in Fig. 11.4 on pg. 174, and is used to set the inviscid (convective) flux scheme, spatial accuracy, and limiters for higher-order reconstruction. The inviscid information is used for every fluid flow computation performed by *GASPE*.

There are two tabs in this section corresponding to what type of mesh is being solved on. If the problem setup is using structured grids (ordered cells with a logical coordinate system), then the **Structured Data** tab is applicable. For unstructured grids (*e.g.*, prisms, pyramids, tetrahedras, etc) the **Unstructured Data** tab should be used to set the parameters.

11.3.1 Global/Marching Strategy

One of the most effective features of *GASPE* is the space marching capability. If a flow is supersonic and aligned with one of the grid coordinate directions, then the flow may be efficiently marched downstream. For marching a viscous flow, the parabolized Navier-Stokes equations are solved.

The **Global/Marching Strategy** options reflect whether a marching or global solution strategy will be used. The **Global Iteration** option implies that *GASPE* will solve for a solution over the entire zone (or zones) simultaneously. This is necessary for subsonic and transonic problems where the characteristic waves are carrying information in all directions (due to the elliptic nature of the governing equations).

If space marching, then one of the appropriate marching directions (either i, j , or k) should be selected. The marching direction should be the same as the primary flow direction. The space marching feature is only available for structured meshes.

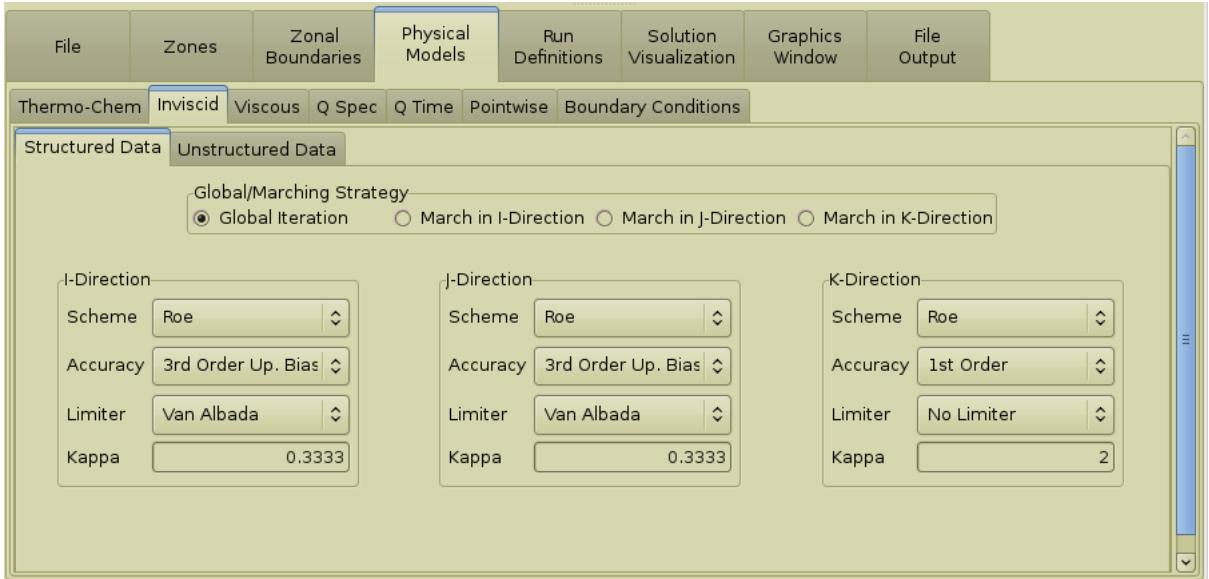


Figure 11.4: The Inviscid tab where convective flux schemes, accuracy, and limiters are set.

11.3.2 Inviscid Flux Schemes

The user may select from a number of inviscid flux options using the **Scheme** option menu. The fluxes are chosen based on the computational coordinate direction (i,j,k for structured grids) and may be mixed (in most cases) as the user sees fit. The number of active inviscid flux schemes should correspond to the dimensionality of the problem. For example, only one flux direction (of three) should be active for a one-dimensional problem (e.g., shock tubes). The only exception is for axi-symmetric problems where all three directions should have an active flux since the fluxes in the third direction do not cancel each other when summing the fluxes around each control volume.

A brief summary of each flux option in *GAS*Pex is now given. The list of flux options are identical for both structured and unstructured meshes.

No Flux Select the No Flux option for one- or two- dimensional problems where you wish to ignore the flux contribution in a given coordinate direction. For example, this is normally applied to the sides of a two-dimensional problem.

Van Leer Van Leer's flux-vector splitting has proven to be the most robust option. We recommend Van Leer's flux for blunt body flows when other flux schemes appear unstable. If you are modeling viscous flow, this flux is NOT recommended for resolving the viscous boundary layer (direction of viscous gradients) in combination with first order spatial accuracy.

Roe Roe's scheme is based on characteristic wave disturbances and by design can capture stationary discontinuities exactly. The Roe flux-difference splitting scheme is less dissipative than Van Leer, and is therefore a better choice for boundary layer flows. This

option represents the standard Roe formulation. Standard Roe has been known to have the "carbuncle" problem for blunt body flows. The Harten correction (Roe w/Harten) is intended to prevent this phenomenon from happening.

Full Flux The full flux scheme should always be used in the space marching direction. The GUI will automatically select the full flux option for the marching direction when a marching strategy is used. In most other cases this flux will result in numerical instability.

Roe w/Harten The Harten entropy fix is available to correct the lack of dissipation at sonic points and in some cases at stagnation lines. Standard Roe has been known to have the "carbuncle" problem in certain cases. The Harten fix is intended to prevent this phenomenon from happening.

AUSM+ The AUSM+ scheme [19] is a blending of flux-vector and flux-difference splitting. Flux-vector splitting schemes (like Van Leer) are efficient to solve, while the flux-difference splitting schemes (like Roe) have the advantage of increased accuracy. AUSM+, which stands for Advection Upstream Splitting Method, tries to be numerically efficient, while attaining a high level of accuracy. The AUSM+ scheme can resolve a stationary normal shock or contact discontinuity and has the positivity-preserving property.

Roe w/Precondition A preconditioned Roe scheme is available for low-speed flows. This variation of Roe prevents the governing equations from going singular as the Mach number approaches zero. As the Mach number goes supersonic, the standard Roe flux is regained. When this flux option is selected, an additional input will appear concerning the use of the precondition matrix (see below for more details).

Weiss & Smith This option is based on the preconditioned flux scheme by Weiss and Smith [20]. The scheme is very similar to the Roe with precondition flux. The major difference is that temperature is one of the solution variables instead of species density. Currently this option is limited to frozen flow. This flux option cannot be combined with any other flux (except the No Flux option). When this flux is selected, an additional input box will appear for **Density Update** (see below).

Artificial Compressibility This option solves the incompressible Navier-Stokes equations using the method of artificial compressibility. The density is assumed constant and will not change, thus chemistry cannot be combined with this option. This is also the default option when using a liquid model. This flux option cannot be combined with any other flux (except the No Flux option). When this flux is selected, an additional input box will appear for **Beta** (see below).

Yee ATM/STVD This is a low-dissipative, high-order shock-capturing flux scheme in which the user controls the amount of dissipation. This flux option is intended for

use with DES simulations (which require low dissipation). The value for the **Yee Dissipation Constant** (an input appears when this flux is selected) should be between 0 and 1. A value of 0 means no dissipation and is essentially a central difference flux scheme. A value of 1 will contain the most dissipation. The default is 0.2 which is good for DES flows without shocks or discontinuities.

HLLE+ An improved Einfeldt's Harten-Lax-van Leer (HLLE) Godunov-type solver [21]. This flux difference splitting scheme attempts to eliminate the erroneous dissipation of other HLLE schemes. Developed originally for use with high-Mach number flows, this scheme can provide more grid independence than the Roe flux schemes. The HLLE+ scheme incorporates a pressure-gradient based switch to transition to the more dissipative HLLE scheme in the presence of strong shocks. The HLLE+ Dissipation Constant, Delta has a default value of 5.

Roe with Preconditioning Options

When the **Roe w/Precondition** flux option is selected, two additional inputs will appear in the Inviscid tab labeled **Include Precondition Matrix** and **Use Modified CFL**. There are three parts to preconditioning. The first is modifications to the standard Roe flux. This will be in effect when the Roe with precondition flux option is selected. The second part is the preconditioning matrix which is used to form the system matrix for implicit schemes. The user has the option of including this second part which consists of the matrix. In theory the precondition matrix aids in solution convergence and makes for a globally consistent precondition scheme. Experience has shown that inclusion of the precondition matrix may cause convergence issues for some flows. For example, airfoil calculations may require that the precondition matrix NOT be included (option set to off). The reasons for this are not yet clear, but until AeroSoft can investigate and research the matter further, the user will have this option. So from a practical standpoint, try running your case with this option on. If the solution will not converge or appears to give an unphysical answer, then turn this option off and re-run the simulation.

The third modification that preconditioning does is to the CFL definition. Preconditioning is effective due to its ability to redefine the characteristic wave speeds. The CFL definition in *GAS*Pex uses the maximum local wave speed to back out the local time step. The user has the option of including the modified wave speed or not by using the **Use Modified CFL** option. In most cases, the user should include the preconditioning of the CFL number, but in some cases better stability is obtained by turning this option off.

Weiss & Smith Options

When using the Weiss and Smith flux option, the user has the choice of freezing the density (since it is no longer a solution variable) or updating it based on the equation of state. This selection is made using the **Density Update** option menu.

Artificial Compressibility Options

When the Artificial Compressibility scheme is selected, the user has several additional options to choose from. These include the value of β and whether or not to solve the energy equation.

A numeric input box for β will appear when the artificial compressibility flux is selected. The value of β will impact both the convergence and accuracy of the solution. A good estimate of β is 5 times the freestream velocity squared ($5V^2$). The units of β are therefore energy (velocity squared). In general, a larger beta will tend to improve accuracy, while a smaller beta will improve the convergence.

The default setting for the artificial compressibility scheme is to solve the continuity and momentum equations, but not the energy equation. Because of the assumption of constant density, the energy equation is de-coupled from the momentum and continuity equations. Therefore it is optional for the user to solve. By not solving the energy equation, temperature effects are ignored. If heat transfer is important or needs to modeled, the user should couple the energy equation with the other fluid equations. By coupling the energy, the solver will directly solve for temperature (*i.e.*, temperature becomes a primitive variable).

Yee ACM/STVD Options

When the Yee flux scheme is selected, an input will appear which allows the user to input the dissipation constant. This is input by setting the **kappa** value. The value should be between 0 and 1. A value of 0 means no dissipation and is essentially a central difference flux scheme. A value of 1 will contain the most dissipation and be equivalent to the Roe scheme.

In practical terms, start with a high value (near 1) and begin to lower it until instabilities arise. Since each application is unique, the optimal value for **kappa** will also be unique and need to be determined.

HLLE+ Options

When the HLLE+ flux scheme is selected, an input will appear which allows the user to input a scheme switching constant. This is input by setting the **Delta** value. Increasing the value of Delta results in the more dissipative HLLE scheme being applied to a larger portion of the domain. Smaller values of Delta result in less dissipation.

11.3.3 Spatial Accuracy for Structured Data

In a finite-volume code, the unknowns are stored as cell averages and *GASPerx* uses these cell averages to reconstruct the pointwise field. Why is the pointwise field needed? The state variables at the cell faces are needed to calculate the flux at the cell faces. The higher-order pointwise reconstruction in *GASPerx* is performed using one of two methods. The first is the MUSCL (Monotone Upstream-centered Schemes for Conservation Laws) approach and the second is WENO (Weighted Essential Non-Oscillatory).

Depending on which spatial accuracy method the user selects, the GUI will reflect different input options. For example, using an option involving the MUSCL scheme for **Accuracy** will yield additional options like **Limiter** and **Kappa**. This is in contrast to when a WENO option is selected, which will display options like **Characteristic variables**, **Uniform mesh**, and **Limit reconstruction**. These input parameters will now be discussed in the following sections.

Accuracy Options for MUSCL

If you wish to use the MUSCL method, then you must select one of the following options for **Accuracy**: **1st Order**, **2nd Order Fully Upwind**, **2nd Order Upwind Biased**, **3rd Order Upwind Biased** or **other**. If the physical model is performing space marching, then there are only two spatial accuracy choices for the marching direction (both of which use the MUSCL method): **1st Order** and **2nd Order Fully Upwind**. This is because only up-wind states are available for the reconstruction.

Each of the previously mentioned spatial accuracies corresponds to a value of κ in the MUSCL formulation. This value is displayed in the GUI (when using a MUSCL type spatial accuracy) and is denoted by the name **Kappa**.

Any value of κ between the range of -1 and 1 is treated as a higher-order reconstruction. If κ is outside this range, the reconstruction is first order. A selection of $\kappa = +1$ yields an arithmetic mean of the adjacent cells and no upwind information enters the reconstruction. A choice of $\kappa = -1$ is the second-order, fully upwind scheme. Choosing $\kappa = 0$ yields a linear interpolation between the upstream and downstream cells. Finally, setting $\kappa = 1/3$ yields a third-order (in one-dimension), upwind-biased reconstruction at the cell face. The best flow fidelity (using the MUSCL scheme) for global calculations occurs with $\kappa = 1/3$.

If you would like to use a value of **Kappa** (κ) other than one of the default values, click in the type-in box and enter a value between -1 and 1. The spatial accuracy will change to **Other**. Note that all the relevant reconstructions are included in the option menu.

MUSCL Limiter Options

For every CFD calculation, a balance exists between accuracy, convergence and monotonicity. Solutions may possess any two of these quantities but not all three to the fullest extent. For example, a monotone, accurate solution will have convergence difficulties, and a monotone, first-order solution will converge fairly easily. The monotonicity of the flow solution is controlled by limiters when using a MUSCL type spatial accuracy option.

If the spatial accuracy is first order, limiters are not needed and the only option available in the **Limiter** option menu is **No Limiter**. For higher order global computations, the user can choose from the following list of limiters: **Van Albada**, **Min-Mod**, **Spekreijse-Venkat**, **Modified ENO**, and **SuperBee**. If discontinuities are absent in a solution, no limiting may be acceptable to obtain a solution; however, with shocks, expansions, and contact discontinuities numerical limiters are usually needed to obtain valid well-behaved solutions.

The different options for structured mesh flux limiters are given below.

Van Albada Van Albada's limiter [22] spreads discontinuities the most yet tends to converge well for most problems.

Min-Mod The min-mod limiter [23] clips reconstructions outside the bounds of a cell-face's neighbors and can cause residual limit cycles.

Modified ENO The modified ENO limiter [24] uses the smaller of two gradients to achieve second-order accuracy. The non-linearity inherent to the gradient selection is necessary to overcome Godunov's monotonicity theorem. Clipping is done where appropriate in the modified ENO limiter whereas the classical ENO scheme does not clip the gradients.

SuperBee Roe's SuperBee limiter [25] lies along the upper limit of the second-order TVD region and typically resolves contact discontinuities well.

Spekreijse-Venkat The Spekreijse-Venkat limiter [26] is tuned to $\kappa = 1/3$.

In the space marching direction (where a full flux must be used), the limiters take on a different meaning than described above for high-order accurate calculations. In the marching direction, the limiters prevent non-physical negative quantities (*e.g.*, pressure and density) that can result from a second-order, fully upwind extrapolation. (Note: The only appropriate high-order choice in the marching direction is second-order, fully upwind.)

The method to prevent a negative pressure or density after interpolation is called **catastrophic limiting** because if uncorrected the iteration process will bomb catastrophically, usually in calculating the speed of sound at the cell face. Listed by increasing level of control, the catastrophic limiters are given below.

Catastrophic P, Rho Reverts to first-order accuracy if either the extrapolated pressure or mixture density is negative.

Catastrophic P, Rhoi Reverts to first-order accuracy if either the extrapolated pressure is negative or any extrapolated specie density is more negative than -10^{-6} .

DQ Max This limiter will allow less than a 25% variance from the first-order interpolation to the interpolated values for all variables.

WENO Options

If the user wishes to use the WENO (Weighted Essential Non-Oscillatory) scheme, four options are available for **Accuracy**. These are: **r=2 WENO**, **r=3 WENO**, **r=4 WENO**, and **3rd order OWENO**. The WENO schemes use a non-linear stencil-selection algorithm to generate truly high-order accurate solutions in multiple dimensions. Reconstructions are done by weighting each of the possible high-order stencils (that is, from fully upwind to fully downwind). In smooth regions, these weights produce the highest accuracy possible for the given number of cells in the stencil. However, when an $O(1)$ discontinuity exists among them, the weights effectively remove the reconstructions that contain the discontinuity. The

computational advantage of the WENO scheme will be most apparent for time-accurate simulations of flows with significant, smooth, high-frequency phenomenon. Examples might include computational acoustics, vortex interactions and turbulence modeling.

The order of accuracy for WENO schemes is designated in the interface through the letter **r**. The **r=2 WENO** option corresponds to two possible stencils ($i-1, i$) and ($i, i+1$) to reconstruct the flow at one Gauss point on each cell face. The algorithm aspires to third-order accuracy in smooth regions of the flow and second-order accurate when a discontinuity is present. Various studies in the literature have shown that a first-order reduction of the error occurs when shocks exists in the flow.

The **r=3 WENO** option includes all stencils from $i-2$ to $i+2$ resulting in a possible fifth-order accuracy in smooth regions. Otherwise, the accuracy reduces to third order. This option reconstructs the flow at four Gauss points per cell face.

The **r=4 WENO** option includes all stencils from $i-3$ to $i+3$ and results in seventh-order accuracy in smooth regions and fourth order otherwise. This option also reconstructs the flow at four Gauss points per cell face.

The **3rd Order OWENO** option alters the weights in the **r=4 WENO** scheme to minimize dissipation for short-length waves. The scheme aspires to best resolve high-frequency waves which travel long distances in the flow.

Spatial accuracy options involving one of the WENO schemes mentioned above allow for a reconstruction of either primitive or characteristic variables. An example of the primitive variables would be density, velocity and pressure. The characteristic variables correspond to the variables that diagonalize a system through the left/right eigenvectors. Reconstruction of the characteristic variables is recommended for most applications (**Characteristic variables selected**). Changing to primitive variables (that is, turning this option off) should not significantly help diverging calculations.

Cell averages can be written mathematically in a uniform computational space using a Jacobian transformation. If this Jacobian is smooth, high-order accuracy can be achieved using the WENO schemes by approximating the Jacobian and metrics at the Gauss points. The pointwise Jacobian is reconstructed using the cell volume as a cell average of the Jacobian. If the **Uniform mesh** option is selected, then the reconstruction of the flow neglects this reconstruction and assumes the mesh is uniformly spaced. A case that runs well using **3rd. Order Upwind Bias** will have almost identical behavior to the **r=2 WENO** scheme with uniform mesh activated. Activating this option improves the stability of the WENO scheme at the expense of formal high-order accuracy.

When the **Limit reconstruction** option is selected, the pointwise WENO correction is not allowed to exceed 80% of the cell-center value. That is, when the reconstruction is less than 20% of the cell average or greater than 80% of the cell average, limiting is applied. The min-mod limiter is used for limiting. This option may be necessary when running WENO simulations where the residual diverges even for a uniform-mesh approximation.

11.3.4 Spatial Accuracy for Unstructured Data

When an unstructured mesh is used in the fluid flow solver, the reconstruction of state variables at cell faces is limited to either first or second order schemes. The following options are currently available.

First Order The first order scheme is identical to that used in structured grids. The cell center values on each side of the cell face are used for the left and right state values.

Second Order (Gauss Divergence) A second order spatial reconstruction is obtained using the cell center values plus the cell center gradients on each side of the cell face. In this option the cell gradients are computed using the Gauss (Divergence) Theorem. This closely follows the work of Barth [27].

Second Order (Least Squares) A second order spatial reconstruction is obtained using the cell center values plus the cell center gradients on each side of the cell face. In this option the cell gradients are computed using a least squares method in which a plane is fitted through the cell center value along with each of the surrounding cell centers. This in theory is more accurate than the Gauss Divergence option, but requires more computational time and memory.

For the second order options, limiting is normally recommended. This ensures robustness of the flow solver since the quality of the grid may differ greatly from one problem to the next.

The options for limiters are **Barth-Jespersen**, **Venkat (baseline)** and **Venkat (uniform flow correction)**. The limiter by Barth and Jespersen [27] is similar to the Min-Mod limiter and the Venkat limiters (name after Venkatakrishnan [28]) are similar to the Van Albeda limiter for structured grids.

Any of the limiters will help control solution oscillations and numerical stability. The **Barth-Jespersen** option may cause residual chatter while the **Venkat** limiters may be slightly more dissipative.

The **Venkat (uniform flow correction)** limiter is identical to the baseline Venkat option except it tries to turn off the limiter in regions of uniform flow. This improves convergence behavior by reducing the potential of limit oscillations.

With unstructured grids, poor quality cells can sometimes wreak havoc on convergence and be a source of local instability. To help prevent this, the user can force any “low quality cell” to be first order. This is done by selecting the **Apply First Order to Low Quality Cells** option. The user can determine the number of low quality cells that *GAS \mathcal{P} ex* computes by displaying the grid information in the **Grid Mgt** section (see Sec. 8.4 on pg. 93).

11.4 Viscous Information

The **Viscous** tab shown in Fig. 11.5 on pg. 182 controls the modeling of the viscous (diffusive) fluxes. Including the viscous flux terms for fluid flow simulations is optional. The default

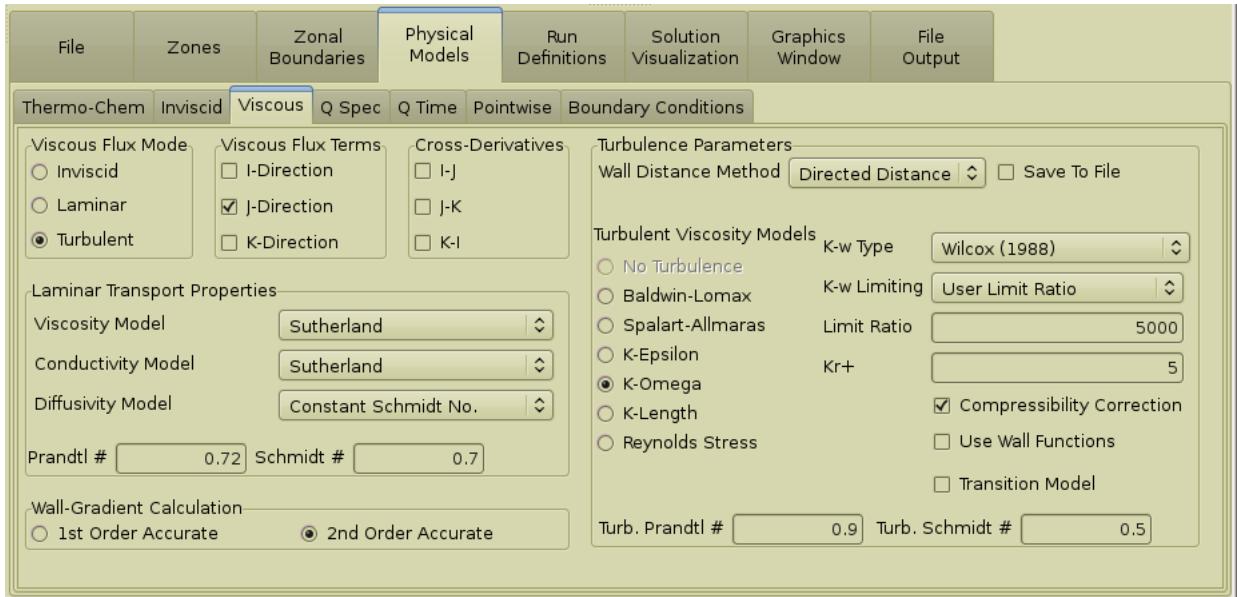


Figure 11.5: The Viscous tab

setting in *GASPer* is not to include the viscous terms (this results in an inviscid simulation). If the user models the flow as viscous, then some of the modeling options that are set in the **Viscous** tab are:

- The choice of laminar or turbulent flow
- For structured meshes, the choice of thin-layer or full viscous gradients (unstructured is always full viscous)
- The laminar transport properties, including viscosity, thermal conductivity and diffusivity models
- The wall gradient accuracy
- The turbulence model and turbulence related parameters

11.4.1 Viscous Flux Mode

Inviscid, laminar or turbulent flow can be modeled by selecting the appropriate radio button beneath the **Viscous Flux Mode** label. If the **Inviscid** option is selected, then none of the parameters in the viscous tab have significance. If the flow is modeled as **Laminar**, then the transport models may be selected, but not the turbulent parameters. For structured meshes, one or all of the **Viscous Flux Terms** should be selected when using laminar or turbulent flow modeling. Finally, all the elements of the viscous tab have significance if the flow is turbulent.

11.4.2 Viscous Flux Terms (Structured Mesh)

With either the **Laminar** or **Turbulent** viscous-flux mode selected, the **Viscous Flux Terms** check boxes may be selected in any combination. The most significant direction for wall bounded flows is normal to a no-slip surface. If a no-slip surface exists, for example, on a $i=1$ grid line (the $i\theta$ boundary), then the viscous flux should be selected in the direction normal to the **I-Direction**. *GASpex* will then calculate viscous fluxes on every constant- i cell face.

Viscous flux terms cannot be included in the space marching direction. If marching is selected (from the **Inviscid** tab), the corresponding viscous flux direction will be unselectable.

If the mesh is unstructured, the **Viscous Flux Terms** do not apply. The viscous gradients on an unstructured mesh are computed in every direction.

11.4.3 Cross-Derivative Terms (Structured Mesh)

Viscous fluxes are computed using flow gradients computed at a face center. In *GASpex*, the gradients are composed of “thin-layer” terms and “cross-derivative” terms. For a majority of flows, the thin-layer terms represent the gradient values well. But for certain flows (or grids that have poor orthogonality) the cross-derivative terms are necessary. For example, in flows with separation or recirculation regions, the user should include the cross-derivative terms that apply. If you are unsure if the cross-derivative terms are needed, then simply include them to be safe.

Cross-derivatives may be selected when any two viscous flux directions are selected. For example, the **I-J** cross derivatives may be included if the viscous terms are being calculated in both the **I-Direction** and the **J-Direction**. This would then be a complete Navier-Stokes calculation in the $i-j$ plane. If you select a cross-derivative combination without selecting the two viscous flux directions, the needed flux terms will be checked for you. Cross-derivatives cannot be included in the space marching direction.

If the mesh is unstructured, the **Cross-Derivatives** do not apply. The viscous gradients on an unstructured mesh are complete (equivalent to thin-layer plus cross-derivative gradients on a structured mesh).

11.4.4 Viscous Gradient Method (Unstructured Mesh)

If the mesh is unstructured, the user can select between two face-based gradient methods. The first method is **Least Squares**. This method is very similar to the second order spatial accuracy method used in the reconstruction of face-based variables. For this method, a least squares fit is made from each cell surrounding the face, as well as each cell adjacent to those two cells. This normally results in a stencil composed of a large number of cells. To help prevent the method from being too dissipative, the two cells adjacent to the face are weighted in order to have more influence on the gradient value.

The second option is **Composite**. This method breaks the face gradient into a normal and tangential component. The normal component is based on the two cells (points) adjacent to

the face, while the tangential gradient is constructed from the cell center gradients (computed for each adjacent cell). The gradient is therefore broken down into a normal component and a tangential component.

Both methods yield a complete or full gradient at the cell face. This is in contrast to the structured data options which allows the user to include either the thin-layer terms by themselves or both thin-layer with the cross-derivative terms (which represents the full gradient).

11.4.5 Laminar Transport Properties

For viscous problems (modeling either laminar or turbulent flow), the user must select an appropriate viscosity model, conductivity model, and diffusivity model. The available choices for each option menu will depend upon the chemistry model selected in the **Thermo-Chem** tab.

Viscosity Model

The viscosity model is used to select the transport model for laminar viscosity. This is applied when either laminar or turbulent flow is modeled. Each species in a chemistry model must contain the necessary data to support the same laminar-viscosity model. With a viscosity for each species, *GASpex* calculates the mixture laminar viscosity using the Wilkes' mixing model [29]. A brief description of each model is given below.

Sutherland's Viscosity Law The **Sutherland** viscosity model corresponds to using the Sutherland [30] curve fits for species laminar viscosity. Like the Blottner curve fits, the viscosity of each species is computed through a common equation. The Sutherland model is default and should always be supported for a specie in the chemical database.

Svehla/Gordon/McBride Curve Fits The **Svehla/Gordon/McBride** option uses curve fits, which are used to calculate all of the transport properties. The curve fits come from what is called the NASA Glenn Chemical Equilibrium Application Program (CEA). Curve fit data must be specified (in the chemical database) for each specie before using this option.

Collision Integrals Collision integrals are based on either Gupta [31] or a formulation based on Lennard-Jones data. Therefore either Gupta [31] or Lennard-Jones data must be specified in the chemical database for each specie before using this model. The option is appropriate for high-speed flows with non-equilibrium effects. When this option is selected, the **Collision Int. Model** option menu selects the specific integral to use.

Liquid Least Squares Curve Fit The **Liquid LSC** provides a polynomial curve fit for liquid viscosity. This is the only valid option when using a liquid model.

More information on the above models is available in the *GASPer* technical reference manual.

Thermal-Conductivity Model

The **Conductivity Model** option menu is used to select one of six thermal-conductivity models: **Eucken's Relation**, **Sutherland**, **Constant Prandtl No.**, **Collision Integrals**, **Svehla/Gordon/McBride**, or **Liquid NSRDS**. Each species in a chemistry model must use the same thermal conductivity model. If a species in the selected chemistry model does not support a certain thermal-conductivity model, then that model option will not be selectable. With species thermal conductivities, *GASPer* calculates the mixture thermal conductivity using Wilke's mixing model (except for the Collision Integral option).

Eucken's Relation The species thermal conductivity is calculated with **Eucken's Relation** which is a function of the species laminar viscosity and thermodynamic properties. Since Eucken's relation [1] does not use any species dependent coefficients, the thermal-conductivity model is valid with any chemistry model.

Sutherland's Conductivity Law The **Sutherland** thermal-conductivity model calculates the species conductivity using the Sutherland [30] curve fit which is a function of the temperature. Since not all species have Sutherland coefficients, not all chemistry models support this option.

Constant Prandtl Number When using **Constant Prandtl No.**, the Prandtl number is assumed constant for each species. When this conductivity model is chosen, the **Prandtl #** type-in box is used to input the Prandtl number for use in *GASPer*. For a Perfect Gas chemistry model, we suggest a number around $Pr = 0.72$. Using the Prandtl number, the species specific heat (C_p), and the species viscosity (μ), the species thermal conductivity (k) can be calculated using $k = (\mu C_p) / Pr$.

Collision Integrals This option uses collision integrals to calculate the mixture thermal conductivity. Either Gupta [31] or Lennard-Jones data must be specified in the chemical database for each specie before using this option. When this option is selected, the **Collision Int. Model** option menu selects the specific integral to use.

Svehla/Gordon/McBride Curve Fits The **Svehla/Gordon/McBride** option uses curve fits to calculate the mixture thermal conductivity. The curve fits come from what is called the NASA Glenn Chemical Equilibrium Application Program (CEA). Curve fit data must be specified (in the chemical database) for each specie before using this option.

Liquid NSRDS The **Liquid NSRDS** represents a liquid curve fit based on the National Standard Reference Data Series (NSRDS). This is the only valid temperature based curve fit when using a liquid model.

More information on the above models is available in the *GASPer* technical reference manual.

Diffusivity Model

The **Diffusivity Model** determines the species diffusion to be used with the current physical model. Mass diffusion occurs when gradients of the mass fractions occur in the flow. *GASPer* uses a reduced form of the Stefan-Maxwell equations, known as Fick's law of diffusion, by assuming the gas behaves as a binary mixture.

Using Fick's law, a binary diffusion coefficient must be determined. There are several models implemented in *GASPer* for determining this coefficient.

Constant Schmidt No. This diffusivity model is supported for every chemistry model in *GASPer*. With this model the Schmidt number is assumed constant for all diffusion processes and is entered using the **Schmidt #** type in box. Using the definition of the Schmidt number, the binary diffusion coefficient, Di , is determined as $Di = \mu / (\rho Sc)$ where Sc is the Schmidt number. Note that for turbulent flow, the turbulent Schmidt number enters the diffusivity as $Di = \mu / (\rho Sc) + \mu_t / (\rho Sc_t)$.

Collision Integrals Collision integrals are based on either Gupta [31] or a formulation based on Lennard-Jones data. Therefore either Gupta [31] or Lennard-Jones data must be specified in the chemical database for each specie before using this model. A modified Fick's Law is used for the diffusion flux in which the sum of the diffusion fluxes for all species sum to zero. The option is appropriate for high-speed flows with non-equilibrium effects. When this option is selected, the **Collision Int. Model** option menu selects the specific integral to use.

Effective, Effective w/Pressure The last two options for diffusivity represent a modeling improvement over Fick's law. The **Effective** diffusion model accounts for individual species diffusion coefficients and the **Effective w/Pressure** model does the same, but also adds pressure-driven diffusion effects. Both options require data for the effective diffusivity model for each specie before using this option. This data is specified in the *GASPer* chemical database.

More information on the above models is available in the *GASPer* technical reference manual.

Collision Integral Model

The collision integrals are based on either Gupta or Lennard-Jones formulations. Gupta data is internally set in the database at this time and is valid for a few models such as Park. The Lennard-Jones data can be entered via the database and is used to compute the collision integrals and the effective diffusivity models.

The **No Ion Correction** is the base collision integral formulation. For ion-ion, ion-electron, and electron-electron collisions, two additional methods using shielded Coulomb cross sections are available for computing the collision integrals. The **Coulomb Potentials** option models Coulomb potentials through the curve fit given in Wright [32]. The last one (**Electron Pressure**) is from Gupta [31] and uses the electron pressure, $p_e = n_e kT$, to calculate a modified collision integral. More detailed information on the above options are available in the *GASPerx* technical reference manual.

11.4.6 Wall-Gradient Calculation

For consistency, the accuracy of the wall-gradient should always be set to **2nd Order Accurate**. The viscous derivatives will then be calculated with a one-sided, second-order accurate difference formula at a physical boundary. The other option is first order accurate, which may provide more numerical stability in some cases.

For an unstructured mesh, the wall gradient is always second order and the **Wall-Gradient Calculation** option is not used.

11.4.7 Turbulence Parameters

The turbulence parameters are active when the **Turbulent** flux mode has been selected. The user must then choose between a zero, one, two-equation, or seven-equation turbulence model. The Baldwin-Lomax model is the default zero-equation model. There is a single one-equation model, Spalart-Allmaras [33], several two-equation models, and a Reynolds Stress model to choose from. Depending on which turbulent model is chosen, there may be additional information to set-up for the model.

If an unstructured mesh is being used, the Baldwin-Lomax model is not a valid choice and will not be selectable. This model is only applicable for structured grids.

Wall Distance Method

A number of turbulence models in *GASPerx* require the distance to the nearest wall for each cell in the grid domain. When required, *GASPerx* will compute this distance based upon the method selected. Since some of the methods are CPU intensive (and done once at the start of the solver), the user has the option of saving the wall distance value in the grid file. If saved, the solver would then read in the value from the grid file upon any future restarts of the solver. If the distance is saved to the grid file, the user can remove it using the **Delete Wall Distance** option in the **Grid Mgt** tab (see Sec. 8.4.5 on pg. 95). A summary of each wall distance method is now given.

Directed Distance The default method is **Directed Distance**. This method is very inexpensive and normally does not require saving to file. For this method, the distance to the nearest surface along a grid line is computed as the wall distance. The method is restricted to a zone such that only surfaces defined by the zone are used. Therefore

the method quickly loses accuracy when the grid domain becomes complex with numerous zones and wall surfaces. The method also assumes that grid lines are fairly orthogonal to solid walls. Since this method traces grid lines to the solid wall, it is not a valid method for unstructured grids.

Closest Point (Brute) For each cell in the grid domain, every solid wall grid face is gone through in order to find the closest wall distance. While this option is very accurate, it can be very time consuming upon start-up if the problem is very large.

Closest Point (Optimal) The **Closest Point (Optimal)** is similar to the option above, except it tries to be more efficient. For each cell in a zone, only the surface points that lie in that zone or in the surrounding zones are used in the wall distance search. While this method can also be expensive to perform, it can significantly reduce CPU times over the brute force option.

ADT Search Method This method uses an alternating digital tree (ADT) method to locate the closest point to the wall. This method should be faster than the closest point method and more accurate than the directed distance method.

Turbulence models that require a wall distance include the Spalart-Allmaras model, Menter's SST (k-omega) models, the LES model, Chien's k-epsilon model, Lam-Brem's k-epsilon model and the k-length models. All other turbulence models ignore the wall distance option because a wall distance is not required.

Turbulent Prandtl Number

The turbulent Prandtl number is used to calculate the turbulent thermal conductivity from the eddy viscosity and specie density. The turbulent Prandtl number enters the energy equation controlling heat flux from turbulent gradients. Experiment has shown that typical values of the turbulent Prandtl number should be $Pr_t = 0.9$.

Turbulent Schmidt Number

The turbulent Schmidt number is used to calculate the mass diffusion coefficient from the eddy viscosity and specie density. Experiment has shown that typical values of the turbulent Schmidt number should be in the range $Sc_t = 0.5 - 0.7$.

Baldwin-Lomax Model

The Baldwin-Lomax turbulence model [34] is an algebraic eddy viscosity, zero-equation model. This is the only zero-equation model in *GASpex*. Several advantages of the model are its computational efficiency and robustness. This model works best in wall bounded flows with favorable pressure gradients. As the flow physics become more complicated, as well as the geometry of the model being tested, the performance of this turbulence model greatly decreases. The model is not reliable for separated or free-shear flows.

Modeling information due to a no-slip wall cannot pass across zonal boundaries for this model. Because of this, the model has possible limitations near zonal boundaries. When using this model, pay close attention to how the problem zones are set up or decomposed.

If an unstructured mesh is being used, the Baldwin-Lomax model is not a valid choice. This model is only applicable for structured grids. All other turbulence models are valid choices when using an unstructured mesh.

Spalart-Allmaras Model

The Spalart-Allmaras [33] (SA) turbulence model is a one-equation model assembled using empiricism and arguments of dimensional analysis, Galilean invariance, and selective dependence on the molecular viscosity. The model is applicable to wall bounded flows, as well as free shear flows. The model does a good job with far-wake and mixing-layer flows. For plane-jet and radial-jet applications, one of the two-equation turbulence models is recommended.

When the Spalart-Allmaras model is selected (see Fig. 11.6 on pg. 190), several other user inputs appear in the turbulence parameters box. The first is the type of model. The options here are **Baseline** and **DES**. The baseline represents the original model formulation based on the 1994 paper [33] with one exception. Instead of the original \tilde{S} definition, the updated term used by Spalart [35] is used.

The Detached Eddy Simulation (DES) model allows for Large Eddy Simulation (LES) in areas away from solid walls, and reduces to the standard RANS formulation in the vicinity of walls. The DES model should only be used for a time dependent calculation and requires a very fine grid in order to resolve the flow eddies (see Sec. 3.11.7 on pg. 49 for tips on running DES problems).

A **Limiting** option menu allows the user to limit the ratio of eddy viscosity to laminar viscosity. During the transient solution of a steady state calculation and around shocks or other flow discontinuities the eddy viscosity may become numerically unstable. By limiting the viscosity ratio, a computation may be more numerical stability. Depending on the aggressiveness of the CFL number and the strength of discontinuities, limiting may or may not be needed to reach a steady state solution. Ratio values of 2000, 500, and 100 are suggested in the option menu (appropriate for boundary layer flows), but the user may input any viscosity ratio using the **Limit Ratio** input box and selecting **User Limit Ratio** from the option menu.

Another input parameter for the Spalart-Allmaras model is the **Compressibility Correction**. Without this correction, mixing rates in compressible shear layers are over-predicted. The correction increases the dissipation of turbulence kinetic energy, thus reducing the spreading or mixing rate. It will not have an impact on low-speed flows and in theory should not affect the boundary layer. But for high-speed flows, some authors have reported a decrease in skin friction and heat transfer results due to having the compressibility correction on. Because of this, the user should only use this option when the flow is supersonic and a free-shear, mixing layer is expected.

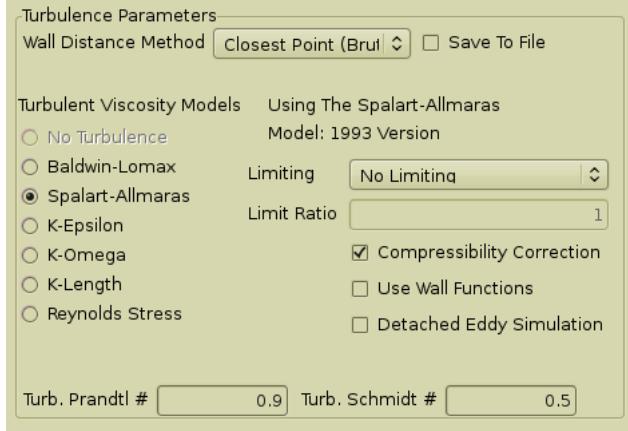


Figure 11.6: The Spalart-Allmaras turbulence modeling parameters.

If wall functions are desired, the user may select the **Use Wall Functions** check box. Wall functions are used when there is insufficient grid to resolve the laminar sublayer and log region. A logarithmic law for the velocity profile is used to attain the correct friction velocity and is valid for zero or favorable pressure gradient layers. The y^+ value for the nearest cell to the wall should be less than 100 when using wall functions. A y^+ should be less than 1 when wall functions are not used. Use wall functions only when the grid is insufficient to resolve the boundary layer.

In version 5.1.2, the DES model was updated to delayed DES (or DDES) based on Spalart *et al.* [36]. The update suppresses DES and extends RANS behavior in boundary layers.

K-Epsilon Models

With the selection of the k -epsilon model, three additional input parameters appear. See Fig. 11.7 on pg. 191 for a close-up of the k -epsilon parameters. There are three variations of the k -epsilon model to choose from using the **K-e Type** option menu. The baseline k -epsilon model is referred to as **Baseline (no wall damping)** and is intended for free-shear flows and requires the use of wall functions for wall bounded flows since it contains no wall damping terms. The **Lam-Bremhorst** [37] and **Chien** [38] models are applicable for both wall-bounded and free-shear flows. In general, k -epsilon models are good for plane and radial jet problems. The various models do not perform well for strong adverse pressure gradients, and are therefore not recommended for flows with separation present.

The Chien model has several limitations that need to be pointed out. Due to its formulation, it requires the value of y^+ (which is dependent on terms at the wall). The value of y^+ can be computed within a zone that has a solid wall, but not for those zones removed from the wall. Because of this limitation, the Chien model should only be used when the zone layout supports it. That is, the user must ensure that zones adjacent to walls are large enough to resolve the necessary turbulence. Another artifact of having a y^+ dependency is that this model is not support for unstructured grids at this time.

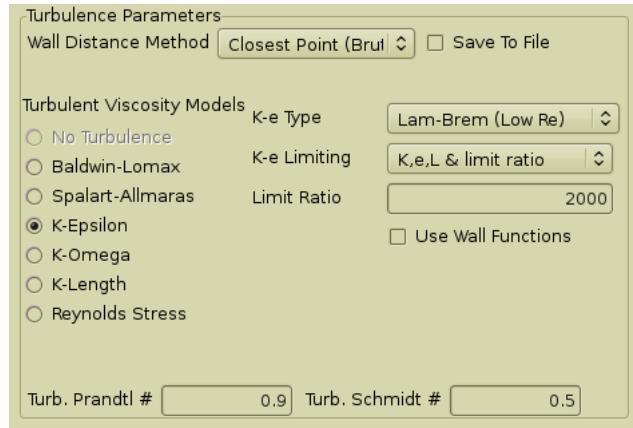


Figure 11.7: The K-Epsilon turbulence modeling parameters.

Two limiting schemes for k -epsilon may help temporal convergence of the often-stiff governing equations. By limiting the values of k and epsilon to minimum and maximum values dependent upon free-stream and local conditions, stability is added to the system of equations during the time integration process. For many cases once a physical solution begins to converge, this limiter may be removed.

In the first limiting scheme (**K, e, L & limit ratio**), k and epsilon are not allowed to have values below 10^{-8} . If the time-integration scheme yields an update that violates this limit, the average of the turbulence kinetic energy from the surrounding cells is used. In addition, the mixing length is not allowed to become lower than 2.5 times the distance from the nearest wall [39]. This effectively limits the dissipation rate. In order to limit the values of k and epsilon from producing an eddy viscosity value that is very large, a limit ratio is applied. The limit ratio is the ratio of eddy viscosity to laminar viscosity. The user must supply this value in the **Limit Ratio** input box. The value of epsilon will be adjusted to prevent the local eddy viscosity value from exceeding the specified ratio value. The default value is 10000, which may be high for wall bounded flows. The user may need to lower this value during the initial start up and then raise the value as the flow develops (see Sec. 3.11.1 on pg. 46 for more tips on solution limiting).

The second limiting scheme (**K, e & limit ratio**) is similar to the first except that the restraint on the mixing length is lifted. Use of a k -epsilon minimization is critical to obtaining solutions, particularly those started from free-stream values.

Wall functions can be applied to any of the k -epsilon turbulence models, but is only recommended for use with the baseline k -epsilon turbulence model or on coarser sequence levels for the other models. When turned on, the wall function will enforce a shear-stress flux determined from the relations of Launder and Spalding at the boundary faces of no-slip walls. In general, wall functions apply when the law of the wall is valid (*i.e.*, no separated flows or flows with large adverse pressure gradients) and are dependent upon the point above the wall where the wall functions are used.

K-Omega Models

The k -omega turbulence model by Wilcox [40, 41] tends to be more accurate for boundary layers with adverse pressure gradients when compared to other one and two-equation models. The k -omega model does not require wall functions (optional) and does an adequate job for various types of free shear flows.

When the **K-omega** selection is made, additional parameters appear for the model. This can be seen in Fig. 11.8 on pg. 194. There are a number of k -omega models in *GASpex*. The first represents the 1988 Wilcox k -omega model. For free shear flows, Wilcox made improvements to the 1988 model which resulted in the 1998 model [41]. For wall bounded flows, the '88 and '98 models should be identical, while for free shear flows the 1998 model has better prediction of spreading rates. In 2006, Wilcox released yet another version of the 1988 model which again improved modeling for free shear flows in addition to strongly separated flows. This variation is referred to as the 2006 model [42].

Wilcox also developed a model for low Reynolds number flows. This is the **Low Re (1998)** option [41].

The Menter SST model [43] is a blend of Wilcox's '88 model and the k -epsilon model. This model tries to apply Wilcox's '88 model to the inner wall regions of a boundary layer and a transformed k -epsilon model for the outer boundary layer regions and the free shear layers. Besides the base SST model, there are several additional variations of the model.

One variation is for detached eddy simulation (DES) and is referred to as **SST/DES** in the list for k -omega model types. The DES model attempts to resolve large eddies in areas away from solid walls, and reduces to a standard SST formulation in the vicinity of walls. The DES model should only be used for time dependent calculations and will require a very fine grid in order to resolve the flow eddies (see Sec. 3.11.7 on pg. 49 for tips on running DES problems).

In version 5.1.2, the DES model was updated to delayed DES (or DDES) based on Sainte-Rose *et al.* [44] and Spalart *et al.* [36]. The update suppresses DES and extends RANS behavior in boundary layers.

Another variation is Menter's scale-adaptive simulation (SAS) model [45, 46]. This model variation is very similar to DES but has less dependency on the grid spacing by using a von Karman length scale for switching from steady to unsteady RANS. This option is labeled **SST/SAS** in the list of model types.

The only Large Eddy Simulation (LES) model in *GASpex* is the **BSL/LES** model [47, 48]. This is a hybrid LES/RANS method that uses a blending function to allow for LES in the outer portion of the boundary layer and traditional RANS near the wall. The model is applicable to high-speed flows and requires a very fine grid and time-dependent simulations to run.

Similar to the k -epsilon and Spalart-Allmaras turbulence limiting, the k -omega models also have the ability to limit the ratio of eddy viscosity to laminar viscosity. This is intended to add stability to the system of equations during the time integration process. While the solution evolves with time toward a steady (or unsteady) state, the values of k and omega may result in large, unphysical values for the eddy viscosity. By limiting the eddy viscosity

value, the solution can get past the transient stage and reach the final solution where limiting may not be needed. In certain cases with strong shocks or expansions, limiting may need to remain on for stability. The limiting is applied to cells locally (uses local values for the viscosity).

The user can select from three pre-defined limit ratios or enter a user defined value using the input box. The higher the limit ratio, the less limiting will occur. For example, a limit ratio of 100 means that the eddy viscosity is restricted to 100 times the local laminar viscosity value. The limiting works by changing the value of omega in order to get the target eddy viscosity value (see Sec. 3.11.1 on pg. 46 for more tips on solution limiting).

When using one of the no-slip boundary conditions, the wall roughness can be specified using the **Kr+** input box. **Kr+** represents the average height of sand-grain roughness elements. A value of **Kr+ = 1** is equivalent to a smooth wall. As **Kr+** increases, the wall roughness will increase as well. The default setting is 5.

The next input parameter for the k-omega model is the **Compressibility Correction**. Without this correction, mixing rates in compressible shear layers are over-predicted. The correction increases the dissipation of turbulence kinetic energy, thus reducing the spreading rate. It will not have an impact on low speed flows and in theory should not affect the boundary layer. But for high speed flows, some authors have reported a decrease in skin friction and heat transfer results due to having the compressibility correction on. Because of this, the user should only use this option when the flow is supersonic and a free-shear, mixing layer is expected.

If wall functions are desired, the user may select the **Use Wall Functions** check box. Wall functions are used when there is insufficient grid to resolve the laminar sublayer and log region. A logarithmic law for the velocity profile is used to attain the correct friction velocity and is valid for zero or favorable pressure gradient layers. The y^+ value for the nearest cell to the wall should be less than 100 when using wall functions. A y^+ should be less than 1 when wall functions are not used. Use wall functions only when the grid is insufficient to resolve the boundary layer.

A **Transition Model** option is available when the **K-w Type** is set to Menter SST. This model solves two additional equations for intermittency and transition onset Reynolds number. These terms are then used in the turbulence model source terms to control the eddy viscosity [49, 50].

K-Length Model

In addition to the k -epsilon and k -omega formulations, *GASpex* supports several $k - l$ models. These are the Smith (1994) model [51] and Goldberg (2005) model [52]. These models solve for the turbulent kinetic energy and a turbulence characteristic length scale. They also require the distance to the nearest wall (which is also required of several other models like Spalart-Allmaras, Menter's SST and k -epsilon).

Turbulence limiting similar to the Spalart-Allmaras and k -omega models is available for the $k - l$ models. This involves limiting the local ratio of the eddy viscosity to the laminar

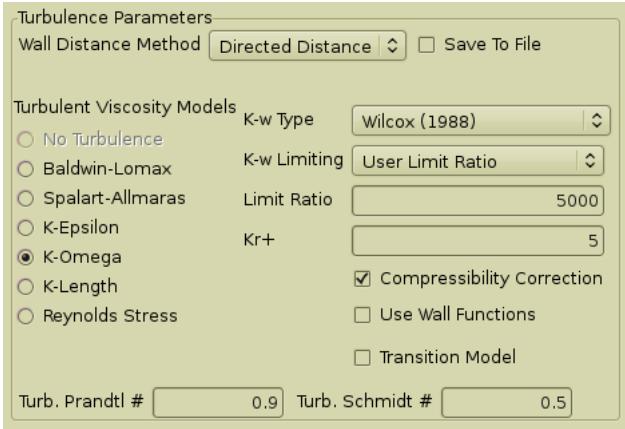


Figure 11.8: The K-omega turbulence modeling parameters.

viscosity.

Reynolds Stress Model

While zero, one, and two-equation models make use of the Boussinesq eddy-viscosity approximation, the Reynolds Stress model does not. The Boussinesq approximation assumes that the principal axes of both the Reynolds stress tensor and the mean strain-rate tensor are coincident everywhere in the flow. The assumption weakens in flows with sudden changes in the mean strain rate, flows with strong curved surfaces, flows with separation and flows with three-dimensional features. A Reynolds stress model, in theory, will circumvent the deficiencies of the Boussinesq approximation.

The Reynolds Stress model implemented in *GASpex* is the Wilcox Stress-Omega model [41]. This model uses the turbulence frequency (or dissipation rate, omega) to compute the dissipation. The pressure-strain correlation uses that of the LRR model and requires no wall reflection term which can be due to a wall-reflection effect. Thus, the model does not require a wall distance term.

Wilcox designed the model to perform similar to the 1998 *k*-omega model. The omega equations are very similar, along with the wall boundary condition for omega which can be used to determine wall roughness. Because of this, the Reynolds Stress model has the same input for **Kr+** as the *k*-omega models for no-slip boundary conditions. **Kr+** represents the average height of sand-grain roughness elements. A values of **Kr+ = 1** is equivalent to a smooth wall. As **Kr+** increases, the wall roughness will increase as well.

Other input parameters for the Reynolds Stress model include limiting of the viscosity through the omega term, and a compressibility correction for mixing layers. The compressibility correction is from Wilcox and follows the same form as that used for the *k*-omega models. Limiting of the viscosity may be beneficial for the initial phase of a run, but should be turned off if possible for the final solution. See the description for limiting under the Spalart-Allmaras model for more information.

In general, Reynolds stress models are not as practical as one and two-equation models due to the extra computational cost associated with them. A Reynolds stress model solves six equations for the Reynolds stress tensor and another equation for the dissipation. While in theory a Reynolds stress model should perform better than a first order turbulence model, there is no guarantee.

11.5 Q Specification

For a fluid flow physical model, the Q spec contains information such as temperature, pressure, density, velocity, laminar and turbulent viscosities, and Mach number. The user only needs to enter a handful of parameters (such as pressure, density, and Mach number) in order to set up a **Q Spec**. These parameters are then used to compute the remaining flow variables.

The **Q Spec** contains both input and computed quantities for the user. Data that the user can input will be covered first, followed by information that is displayed for the user (non-editable).

The flow data is divided into two sections, state variables and flow variables. For the state variables, the user will input data such as temperature, pressure, mixture density, and specie density. The **Input state as** option menu allows the user to select which state variables to input. Any remaining state variables will then be computed by the GUI and displayed.

If the selected chemistry model contains a single specie, then only three combinations for entering the state variables exist. These are: “Temp & Pressure”, “Temp & Density”, and “Press & Density”. Two of the three state variables must be specified by the user. The third state variable is then computed using the other two.

If the physical model contains multiple species (*e.g.*, the Kang & Dunn chemistry model), then the user must input two of the following state variables: temperature, pressure, or mixture density. Because multiple species exist, the user must also specify one of the following for each specie: density, mass fraction, or mole fraction. Individual specie data is entered into the **Specie Properties** table.

When selecting the thermodynamic setting (the **Thermo-Chem** tab), the user may use the option to model the non-equilibrium vibrational energy for certain species. If this is the case, the user will have the option of specifying the vibrational temperature (**Tvib**) for each non-equilibrium vibrational specie. In order to do so, the user must de-select the **Lock Vib = T** option and enter the temperature data into the **Vib Specie** table. The default setting is to set the vibrational temperature equal to the flow temperature.

The second category of input data is for the flow variables. This includes the flow speed and direction, as well as turbulence variables. The different options for specifying the flow variables are now explained.

The first input from the user is the flow speed. This is done in the **Flow speed** frame with the option of entering either the Mach number or velocity magnitude. The speed of sound is also displayed here, but is for display purpose only (non-editable).

The flow angle is input directly under the flow speed. This input is specified by entering the x,y,z components (direction cosines) of the velocity vector. The components can be normalized by clicking on the **Normalize cosines** button. For example, if you enter $u/|V| = 1$, $v/|V| = 1$, $w/|V| = 0$ and then click **Normalize cosines**, the normalized x and y velocity components will be $u/|V| = 0.707107$ and $v/|V| = 0.707107$, corresponding to a 45° angle of attack in the x - y plane. An option to input the actual flow angle is also available by pressing the **Input angle** button. The user can then input the flow angle with respect to the X-Y, Y-Z, or X-Z plane. The flow angle is always relative to the first coordinate given (*e.g.*, x in the “X-Y” plane type).

For inviscid, laminar, zero and one-equation turbulence calculations, the **Q Spec** is completely specified by entering the quantities discussed above. For two-equation turbulence models or the Reynolds Stress model, the user must specify the turbulence intensity along with either the eddy viscosity to laminar viscosity ratio or the turbulent length scale. The input selection is made with the **Input turb as:** option menu located inside the **Turbulent input** frame.

The turbulence intensity (“Turb. Intensity, T.I.”) should be specified in the appropriate input box and is used to set the turbulent kinetic energy (TKE) value. The turbulent length scale or viscosity ratio (ratio of eddy viscosity to laminar viscosity) is used to set the epsilon or omega value (depending on turbulence model type). These parameters (“Turb. Intensity”, “Visc Ratio”, “Length scale”) control the initial turbulent kinetic energy and dissipation rate/frequency in the flow field as follows

$$\text{Turbulent kinetic energy } k = (V^2 \cdot Ti^2)$$

$$\text{Dissipation rate } \epsilon = k^{3/2}/l_t$$

$$\text{Dissipation frequency } \omega = k^{1/2}/(0.09 \cdot l_t)$$

$$\text{Eddy viscosity } \mu_t = (C_\mu \cdot \rho \cdot k^2)/\epsilon$$

In the above equations, Ti is the turbulent intensity, l_t is the turbulent length scale, and C_μ is a proportionality constant ($C_\mu = 0.09$ for all turbulent models except for $k - l$ which has $C_\mu = \sqrt{2}/18^{1/3}$).

For zero and one-equation turbulence models, there are no required input parameters. The turbulence viscosity is initialized to 1/10 the laminar viscosity for the Spalart-Allmaras one-equation turbulence model.

The remaining parameters are computed from the input parameters discussed above and displayed for the user. The values are non-editable. The viscosity, thermal conductivity and Reynolds number per unit length are functions of the density, velocity, and temperature. The viscosity and thermal conductivity are also dependent on the transport model selected in the **Viscosity** tab.

The thermodynamic quantities are primarily functions of the temperature and the chemistry model. The mixture gas constant (R), the mixture specific heat (C_v) and ratio of specific heats (γ) have dependencies on the thermodynamic model selected in the

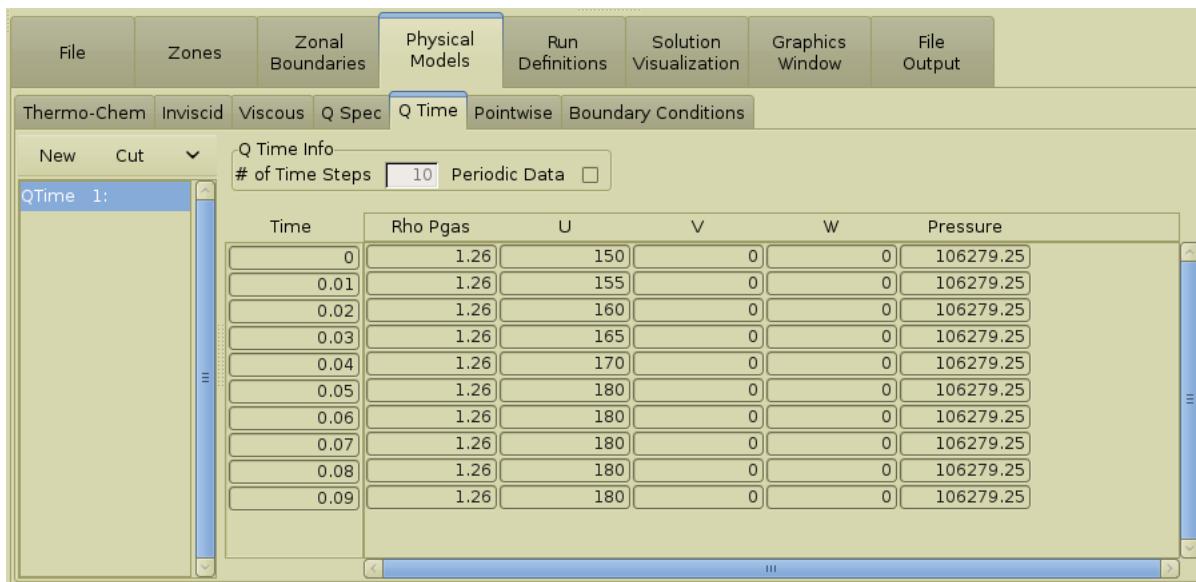


Figure 11.9: A sample **Q Time** tab with data.

Thermo-Chem tab. The total temperature and pressure are functions of the Mach number. Changing the temperature or pressure may change some of the data.

11.6 Q Time

The **Q Time** tab can be best described as a time varying **Q Spec**. But unlike the **Q Spec** where you have multiple flow variables defined, the **Q Time** is limited to one of three types: primitive variables, wall temperature, and back pressure. The **Q Time** allows the user to have time varying **Q** values, which may be necessary for proper implementation of time dependent calculations. The **Q Time** is only used as a reference for boundary conditions.

A sample **Q Time** tab is shown in Fig. 11.9 on pg. 197 where the **Q Time** is set up for primitive variables. When a **Q Time** is created, a table will appear displaying the data. The first column in the table will always represent the time in the current unit system. The remaining columns will depend on the type of data being specified. The figure has the primitive variables and gives a table with columns for density, u, v, w-velocity, and pressure. The units on the primitive variables, as well as the temperature and pressure data types, are consistent with the input deck unit system specified in Chap. 5 beginning on pg. 69.

Above the table is listed the number of time steps and a check box for periodic data. For data to be periodic, the first and last row of data should be identical and the time for the last row should represent the period.

11.6.1 Creating a **Q Time** Data Set

Initially there will be no **Q Time** data sets. The user must create each data set and can have any number of them. To create a **Q Time** data set, follow the steps below.

1. Select the **New** option. Selecting this will bring up the window shown in Fig. 11.10 on pg. 199.
2. Follow the four steps to create a new **Q Time** data set. The first step has you select the type of data you wish to use. The choices include primitive variables, wall temperature, and back pressure.
3. For the second step in the dialog, enter the number of time steps. This will be the number of rows in the **Q Time** table.
4. Next, decide whether the data is periodic or not. If the data is periodic in time, select the check box.
5. Finally, Decide how you want to initialize the data. Your choices are to read the data from a file or use a **Q Spec** to set all the values. If you read data from a file, you should select the units type and then press **Read Data**. See Sec. 11.6.2 on pg. 198 for more information on entering data using a file. If you want to initialize the data using a **Q Spec**, select which **Q Spec** and press **Initialize Using QSpec**.

After the data is read in from a file or initialized using a **Q Spec**, the pop-up window will disappear and a table will appear in the **Q Time** section.

11.6.2 Reading **Q Time** Data From A File

To read in **Q Time** data from a file, follow the steps in Sec. 11.6.1 on pg. 198 to create a new **Q Time** data set. When it comes to initializing the data, select the **Read Data** button. Be sure to select the correct unit system of the file before pressing the **Read Data** button. The file that you select should have the data in ASCII format. Each line in the file should contain the data values, starting with the time. The file should not have any headers, only data. As an example, a data file is shown below for the primitive variables density, u, v, w-velocity, and pressure. The first number of each row represents the time.

0.0	1.53	150	0	0	116763
0.1	1.53	160	0	0	116763
0.2	1.53	170	0	0	116763
0.3	1.53	180	0	0	116763
0.4	1.53	190	0	0	116763
0.5	1.53	200	0	0	116763
0.6	1.53	190	0	0	116763
0.7	1.53	180	0	0	116763

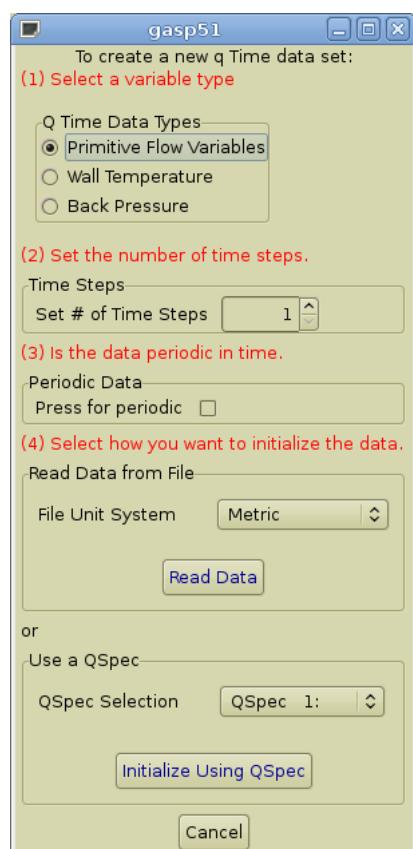


Figure 11.10: The window used to create a new Q Time data set.

0.8	1.53	170	0	0	116763
0.9	1.53	160	0	0	116763
1.0	1.53	150	0	0	116763

11.6.3 Q Time and Boundary Conditions

The purpose of **Q Time** data is to be a reference for boundary conditions in time-dependent flow simulations. Boundary conditions can be referenced to either a **Q Spec**, a **Q Time** or **Pointwise** data (see Sec. 10.9.3 on pg. 160 for more information).

Inside the **Boundary Conditions** tab, each surface family has a boundary condition assignment, a split/full flux setting, and then a Q Source assignment. Not all boundary conditions need a Q Source, but for the ones that do, you may select the **Q Time** option if one or more **Q Time** data sets exist. Once the **Q Time** option is selected, the **Q Time** option menu (in the last column) will become active and the appropriate **Q Time** can be selected.

We will take for example the **PBack Subsonic Outflow** boundary condition. This BC requires a back pressure to be specified and will use either a **Q Spec**, a **Q Time**, or **Pointwise** data to attain the pressure value. If a **Q Time** exists for either back pressure or primitive variables, then the **Q Time** option for “**Q Source**” can be selected. Once **Q Time** is selected, go to the **Q Time** column and select the correct **Q Time** from the option list. For example, a **Q Time** would be beneficial if the problem were time dependent and the back pressure varied with time.

11.6.4 Q Time and Turbulence

When using **Q Time** with a turbulence model, some considerations need to be taken into account. The one and two-equation turbulence models have state variables included in the primitive variables. So when creating a **Q Time** with the primitive variable type, the correct turbulence variable needs to be included in the data set. For two-equation turbulence, the extra primitive variables will be the turbulent kinetic energy (TKE) and either epsilon, omega, or length. For Spalart-Allmaras, the extra variable is a viscosity type variable.

Depending on the turbulence model and boundary condition, the turbulence variables may or may not ever be used. For example, if the user sets the turbulence boundary condition to **1st order extrapolation**, the turbulence variables in the **Q Time** data set will not be used.

11.6.5 Q Time Select List

The user may display different **Q Time** data sets using the select list (located at the left of the **Q Time** tab). For example, selecting on an item in the list will change the display to reflect that item’s data. Initially there are no **Q Time** data sets when a physical model is first created. The **New** option is used to create new **Q Time**, while the **Cut** option is used to remove a data set. The **Copy** feature is used to place a copy of the **Q Time** onto the clip

board. Once the clip board has a copy, the **Paste** option can be used to create a new **Q Time** based on the saved copy.

11.7 Pointwise Data

For fluid flow physical models, there are three basic types of pointwise data. These are: primitive variables, wall temperature, and intermittency. A pointwise data set will contain at least one of these types. Combinations of the data types are also possible. For our example, the wall temperature data type was selected.

See Sec. 10.8 on pg. 153 for a general discussion of pointwise data.

11.7.1 Pointwise Data Types

There are a number of data types associated with a pointwise data set. The data types presented here are unique to the Navier-Stokes flow solver.

The first data type is Primitive Flow Variables. The Primitive Flow Variables will always consist of specie densities, velocity components (u, v, w), and pressure. In addition to these, the primitive variables may also include specie non-equilibrium vibrational energy, and/or turbulence quantities like the turbulence kinetic energy. How the physical model is set up will determine what primitive variables are needed for the data set. In any case, the variables will be listed at the top of the table, where each variable will have a column for input.

For the Wall Temperature type, the data set will consist of a single column of data for temperature. Wall temperature values are commonly used with the **No Slip T=T_{wall}** boundary condition.

For the Intermittency type, input requires a single value for each face (similar to wall temperature). The intermittency values are used in conjunction with the turbulence model to specify transition from laminar to turbulent flow. Most values for the intermittency should be between 0 and 1, where 0 indicates a laminar flow and 1 indicates fully turbulent flow. These values are used to scale the eddy viscosity inside the flow solver.

There are also combinations of the previous types. Since each surface can only be set up once for pointwise data, this ensures that the user can input all the data needed.

11.8 Boundary Conditions

For the fluid flow physical model, boundary conditions are assigned for the basic flow (labeled as **Flow B.C.** in the table header), as well as for the turbulence equations (**Turbulence B.C.**). The turbulence boundary conditions are only selectable if turbulence models are used in the physical model. See Sec. 7.1.3 on pg. 80 for setting up a BC surface family.

If the physical model uses a turbulence model involving one or more turbulence equations, then the user may have the option of setting the **Turbulence B.C.**. The turbulence BC option menu will only be active if the **Flow B.C.** supports it. In other words, some flow boundary

conditions only have one possible turbulence boundary condition (like no-slip conditions), while other conditions have several valid BC's for turbulence models (*e.g.*, Fixed at Q). The possible turbulence boundary conditions are explained below (see Sec. 11.8.2 on pg. 207).

11.8.1 List of Fluid Flow Boundary Conditions

Below is a list of boundary conditions currently available in *GASpex* for Navier-Stokes modeling. A more technical write-up of the boundary conditions can be found in the *GASpex* technical reference manual.

Fixed at Q Sets all boundary values according to a Q Spec, Q Time, or Pointwise data. This condition is normally used at supersonic inflow boundaries.

1st Order Extrapolation Does a first order extrapolation from the interior cells for the boundary values. This condition is normally used at outflow boundaries in which the flow is supersonic.

2nd Order Extrapolation Does a second order extrapolation from the interior cells for the boundary values. The exiting flow should be void of any shocks or expansions to use this BC.

Riemann Subsonic In/Outflow Riemann invariants from characteristic theory are satisfied to determine the flow and sound speed at the boundary. The boundary condition adjusts itself depending on the direction of the flow (into or out of the boundary). The condition is normally used at far-field boundaries in sub-sonic, exterior flows.

P0-T0 Subsonic In/Outflow Applies either the “P0/T0 Subsonic Inflow” or “PBack Subsonic Outflow” boundary condition on a boundary face depending on the flow direction (into or out of the boundary).

Non-reflective In/Outflow The non-reflective boundary condition propagate continuous acoustic and convective waves across inflow and outflow boundaries without generating non-physical, reflective signatures. The boundary conditions implement the finite-wave model of Atkins and Casper (AIAA-93-0152). The wave equations for isotropic, compressible flow are solved to determine a consistent boundary state. The end states to the wave problem correspond to a fixed, far-field value and a reconstruction from the interior data. The extension to multiple dimensions assumes a one-dimensional wave system normal to the far-field boundary.

Fixed Mass Subsonic Inflow For subsonic inflow boundaries, where the mass flow is held constant over the surface based on the “Q Source” data. The “Q Source” data provides the density and velocity magnitude, which combined with the inflow area gives a mass flow. The boundary condition preserves total temperature and sets the velocity normal to the inflow surface.

P0-T0 Subsonic Inflow For subsonic inflow boundaries, the total pressure and total temperature at the boundary is set as a function of the “Q Source” Mach number. This condition is good for inflow boundaries with a HIGH sub-sonic flow (the P0-Riemann Subsonic Inflow is recommended for low sub-sonic conditions).

P0-Riemann Subsonic Inflow For subsonic inflow boundaries, the total pressure and total density at the boundary is set using Riemann invariants from characteristic theory. This boundary condition is recommended over the previous P0-T0 Subsonic Inflow. A “Q Source” is required for the boundary condition. This condition is good for sub-sonic inflow boundaries in which the flow is normal to the boundary surface (such as an internal duct problem). It is also a good choice for low sub-sonic inflows.

Incompressible PO Inflow For incompressible, subsonic inflow boundaries, the total pressure at the boundary is set as a function of the “Q Source” Mach number. This BC should only be used with the “Artificial Compressibility” inviscid flux option.

PBack Subsonic Outflow For outflow boundaries, the back pressure is set according to the “Q Source” selection. All other boundary values are extrapolated from the first interior cell. This condition is normally used when the pressure at an exit boundary is known and the flow exiting is sub-sonic.

PBack/Forced Outflow Applies the “PBack Subsonic Outflow” boundary condition and then ensures that the velocity is forced to exit the boundary.

Riemann/PBack Outflow For subsonic outflow boundaries, the pressure as specified in the “Q Source” selection is used to set the second ghost cell, along with the internal entropy, total temperature and flow angle. For the first ghost cell flow values, Riemann variables, internal entropy, and the flow angle is used.

Relax P Inflow For use with sub-sonic inflow boundaries. Pressure is extrapolated and the remaining boundary values are set according to the “Q Source” selection.

Relax P Inflow/Pback Outflow Applies the “PBack Subsonic Outflow” boundary condition for boundary faces where the flow is moving out of the boundary. For inflow boundary faces, pressure is extrapolated and the remaining boundary values are set according to the “Q Source” selection.

Forced Outflow This is similar to 1st order extrapolation except that the velocity is forced to exit the boundary.

Tangency An inviscid, impermeability condition designed for solid wall surfaces. This is normally used for solid walls when viscous effects are neglected (*e.g.*, inviscid flows).

Slip Velocity & Temp (High Knudsen #) For high Knudsen number flows in which the fluid may no longer be considered a continuum (transition regime to free molecular flow). A wall “slip” velocity and temperature are calculated based on local flow conditions, a user specified wall temperature, and inputs under the **Special BC Data**.

Slip Wall T=T_{wall} This boundary condition allows the user to fix the velocity and temperature at the wall using the “Q Source”. The pressure is extrapolated.

No Slip Adiabatic A viscous, solid wall boundary condition where the temperature gradient at the surface is set to zero by extrapolating the pressure and specie densities. The heat flux through the surface should therefore be zero in most cases (species diffusion, radiation, and turbulent kinetic energy at the wall may result in a finite energy flux). This condition is normally used for solid walls in which the viscous effects are simulated as either laminar or turbulent and the wall is adiabatic.

No Slip T=T_{wall} A viscous, solid wall boundary condition where the temperature is set according to the “Q Source” selection.

No Slip w/Heat Transfer A viscous, solid wall boundary condition where the wall temperature is set in order to match a user specified heat flux. The heat flux is set in the **Special BC Data**.

No Slip T=T_{wall}; Split A viscous, solid wall boundary condition where the temperature is set according to the “Q Source” selection. The boundary condition is applied to the first interior ghost cell and the flux is split at the wall.

Dual:Fixed/No-Slip A dual boundary condition that switches between two BCs depending on the value of the “Q Source” pressure. If pressure is greater than zero, the “Fixed at Q” boundary condition is used. Otherwise the “No Slip Adiabatic BC” is applied. This was added to *GASpex* to be used with pointwise or **Q Time** data in which the user either fixes the flow variables or allows the adiabatic no-slip condition to apply depending on the local pressure value input.

Dual:PO-TO In/No-Slip A dual boundary condition that switches between two BCs depending on the value of the “Q Source” pressure. If pressure is greater than zero, the “P0-Riemann Subsonic Inflow” boundary condition is used. Otherwise the “No Slip Adiabatic BC” is applied.

Dual:PBack Out/No-Slip A dual boundary condition that switches between two BCs depending on the value of the “Q Source” pressure. If pressure is greater than zero, the “PBack Subsonic Outflow” boundary condition is used. Otherwise the “No Slip Adiabatic BC” is applied.

Dual:PO-TO,Pback/No-Slip A dual boundary condition that switches between two BCs depending on the value of the “Q Source” pressure. If pressure is greater than zero, the “P0-T0 Subsonic In/Outflow” boundary condition is used. Otherwise the “No Slip Adiabatic BC” is applied.

Surface Reactions, T=T_{wall} A viscous, solid wall boundary condition where the temperature is set according to the “Q Source” selection. The normal pressure gradient

is set to zero. The species concentrations are determined by solving a coupled system of mass-balance equations. The mass balance includes the effects of diffusion as well as the surface reactions defined in the **Special BC Data** section associated with this boundary condition. If the associated Q Specification includes mass flow, the momentum equations will be solved to include a boundary blowing rate consistent with the Q Specification. Otherwise no-slip velocity is prescribed. This boundary condition is typically used for finite-rate chemically reacting flows.

Surface Reactions, Energy Bal A viscous, solid wall boundary condition where the temperature is set by satisfying an energy balance equation. Energy balance terms include the effects of convection, conduction, diffusion and radiation. The normal pressure gradient is set to zero. The species concentrations are determined by solving a coupled system of mass-balance equations. The mass balance includes the effects of diffusion as well as the surface reactions defined in the **Special BC Data** section associated with this boundary condition. If the associated Q Spec includes mass flow, the momentum equations will be solved to include a boundary blowing rate consistent with the Q Spec. Otherwise no-slip velocity is prescribed. This boundary condition is typically used for finite-rate chemically reacting flows.

Slip w/Surface Reactions, T=T_{wall} Similar to the “Slip Velocity & Temp (High Knudsen #)” boundary condition, with the exception that mass-balance equations are solved for species concentrations. The species concentrations are determined by solving a coupled system of mass-balance equations. The mass balance includes the effects of diffusion as well as the surface reactions defined in the **Special BC Data** section associated with this boundary condition. This boundary condition is typically used for finite-rate chemically reacting flows.

Porous Wall; Passive This condition models walls with small pores in which flow suction or blowing occurs due to the cavity pressure under the wall. This boundary condition requires a number of user specified parameters (set in **Special BC Data**). For more details, see the boundary condition chapter in the *GASPerx* Technical Manual.

Porous Wall; Active This condition models walls with small pores in which flow suction or blowing occurs due to a user specified mass flow. This boundary condition requires a number of user specified parameters (set in **Special BC Data**). For more details, see the boundary condition chapter in the *GASPerx* Technical Manual.

Wall Blowing/Suction This condition models a viscous or inviscid solid wall with prescribed blowing or suction. The pressure and species densities are extrapolated. The mass flow is taken from the Q Spec density and velocity magnitude values. The condition of blowing or suction is set in the **Special BC Data**.

X-Axis Axi-symmetric For use with axi-symmetric flows where the singular axis lies along the *x*-axis.

Y-Axis Axi-symmetric For use with axi-symmetric flows where the singular axis lies along the y -axis.

Z-Axis Axi-symmetric For use with axi-symmetric flows where the singular axis lies along the z -axis.

Symmetry Plane Assumes the flow is symmetric about the boundary face. There are no restrictions as to the surface orientation.

Negative Axi-symmetric Wall An axi-symmetric BC for the side-wall in the negative rotation direction. Assumes the flow is axi-symmetric, and that the singular axis is along the x axis. The angular displacement of the side walls must be exactly $-\pi/80$ or -2.25° . The acute angle in a cross flow pie wedge must be exactly 4.5° .

Positive Axi-symmetric Wall An axi-symmetric BC for the side-wall in the positive rotation direction. Assumes the flow is axi-symmetric, and that the singular axis is along the x axis. The angular displacement of the side walls must be exactly $+\pi/80$ or $+2.25^\circ$. The acute angle in a cross flow pie wedge must be exactly 4.5° .

2-Parameter Inflow This is an axi-symmetric, combustion inflow boundary condition based upon Wang, et al. AIAA 93-0111. Pressure is extrapolated using a second order stencil. The specie values are maintained at the “Q Source” mass fractions. The u-velocity depends on the internal pressure, the v-velocity is set to yield zero vorticity and the w-velocity is set to zero.

Fixed-u Outflow Similar to the 1st order extrapolation boundary condition except that the u-velocity is fixed according to the Q source selection.

Periodic Rotation: Positive Wall This condition is used with user-created zonal boundaries to do a periodic flow about some user defined, 3-D slice of axi-symmetric flow. The user must first create a zonal boundary using the two corresponding surfaces. The surfaces then should be moved out of the Pt2Pt zonal boundary folder and into the surfaces BC folder. The rotation axis and angle are then specified in the **Special BC Data** section.

Periodic Rotation: Negative Wall The corresponding negative wall to the above boundary condition. The positive and negative walls are defined by the standard right hand rule.

Zonal Boundary Interface This boundary condition assumes the surface is part of a zonal boundary and that it receives information from the adjacent zone. The boundary condition therefore does nothing.

LES: ZB w/Perturbed Velocity This boundary condition is intended for LES simulations where turbulent spots are generated at a zonal boundary. The BC overrides the velocity in order to generate random eddies in the flow.

LES: Inflow w/Perturbed Velocity This boundary condition is intended for LES simulations where turbulent spots are generated at an inflow boundary. The BC generates random eddies in the flow as well as sets the pressure and density based on the **Q Spec** values.

Fluid/Solid Interface For conjugate heat transfer problems, this boundary condition sets the energy balance for heat transfer between the solid and flowfield. This boundary condition should only be used on zonal boundary surfaces in which one surface belongs to a fluid zone and the other belongs to a solid zone. Recall that a zonal boundary can be overridden as a BC inside the Pt2Pt ZB folder of the tree view.

11.8.2 List of Turbulent Boundary Conditions

When setting up a fluid flow physical model and a turbulence model is used, the user may need to set a boundary condition for the turbulence model. Depending on the fluid flow boundary condition, the turbulence option menu may or may not be selectable. If the turbulence boundary condition option menu is selectable for a surface family, the user will need to select a BC from the following list.

Default Setting Each turbulence model will have a default setting for inflow and outflow boundary conditions. This condition sets the default. The actual default conditions are given in the *GASPEX* technical reference manual.

Extrapolate The turbulence parameters are set at the boundary using first order extrapolation.

Fixed at Q The turbulence parameters are set at the boundary using the “**Q Source**” values for turbulence.

Inflow/Outflow This boundary condition is a combination of Extrapolate and Fixed at **Q**. The turbulence parameters are set at the boundary using “**Q Source**” values if the flow is entering the domain, and set to first order extrapolation if the flow is exiting the grid domain.

11.8.3 Special BC Data

In general, the most common use of **Q Spec** data is to specify parameters for the boundary conditions. Not all boundary conditions require information from the user, but for the ones that do, the **Q Spec** supplies much of this information. But for some boundary conditions, unique parameters are required for input. In these cases, the input parameters are listed in the **Special BC Data** tab (inside the Boundary Condition section) for the user to specify.

All the data in this section is available for user input. The data is grouped into frames in which the boundary condition takes on the frame title. For example, the **No Slip w/Heat Transfer BC** data frame corresponds to the “No Slip w/Heat Transfer” boundary condition.

Tool tips (pop ups that occur when the mouse pointer hovers over the text) can also be used to determine which boundary conditions the data belongs to.

No Slip w/Heat Transfer BC Applies to the following BCs:

- No Slip w/Heat Transfer
- Surface Reactions, Energy Bal.

The user provides the heat flux on a boundary. Positive values represent heat transfer into the domain.

Slip Velocity & Temperature BC Applies to the following BCs:

- Slip Velocity & Temp (High Knudsen #)
- Slip w/Surface Reactions, $T=T_{\text{wall}}$

Reflected Molecules is the fraction of diffusely reflected molecules with values ranging between 0 and 1. **Thermal Accommodation** is a coefficient that is used in computing the slip wall temperature with values ranging between 0 and 1. **dT/dS Creep** represents the thermal creep effects along the wall.

Generalized Surface Reactions BCs Applies to the following BCs:

- Surface Reactions, $T=T_{\text{wall}}$
- Surface Reactions, Energy Bal
- Slip w/Surface Reactions, $T=T_{\text{wall}}$

Surface Reactions is used to select which, if any, surface reactions to apply. The surface reactions are created inside the database and stored as part of the chemistry model. Surface reactions can be viewed or edited using the *GASPer* database GUI (see Sec. 3.3.3 on pg. 34). **Surface Emissivity** is the effectiveness of the surface in emitting thermal radiation. Emissivity values range from 0 (no emission of radiation) to 1 (black-body radiation).

Mass Injection/Extraction BCs Applies to the following BCs:

- Surface Reactions, $T=T_{\text{wall}}$
- Surface Reactions, Energy Bal
- Wall Blowing/Suction

Wall Velocity Type controls whether the wall velocity is blowing, suction, or zero (no wall velocity).

Porous Wall (passive or active) BC Applies to the following BCs:

- Porous Wall: Passive
- Porous Wall: Active

Porous hole diameter is the size of the holes perforating the surface. **Plate thickness** is the thickness of the solid surface. **Porosity** is the ratio of total hole area to porous surface area. **Hole Factor** is the correction factor for non-cylindrical holes.

Porous Wall (active only) BC Applies to the following BC:

- Porous Wall: Active

Porous mass flow is the user specified mass flow rate where a positive value indicates injection. **Cavity Temperature** is the temperature of the reservoir located under the surface. **Relax Factor** is an under-relaxation factor used for convergence stability.

Periodic Rotation BC Applies to the following BCs:

- Periodic Rotation: Positive Wall
- Periodic Rotation: Negative Wall

Rotation angle is the angle of the “pie slice” or angle between the negative and positive walls. **Rotation axis** is the axis about which the rotation occurs. This must be one of the three primary axis.

All Solid Wall BCs Applies to the following BCs:

- Tangency
- No-Slip Adiabatic
- No-Slip $T=T_{wall}$
- No-Slip w/Heat Transfer
- Slip Wall $T=T_{wall}$
- Surface Reactions, $T=T_{wall}$
- Surface Reactions, Energy Bal
- Slip w/Surface Reactions, $T=T_{wall}$

Enforce Wall Rotation Velocity is used to apply the rotation rate (p,q,r) on solid surfaces using this Q Spec. **Fixed Frame (Wall Velocity = 0)** forces the wall velocity to be zero.

LES/DES Initialization Flags Applies to the following BCs:

- LES: ZB w/Perturbed Velocity
- LES: Inflow w/Perturbed Velocity

Reference TKE is a reference value for the turbulent kinetic energy. **Reference Length** is a reference length scale for turbulent eddies.

The technical *GASpex* manual has a chapter on boundary conditions which goes into detail on most BCs offered in *GASpex*. The user is encouraged to read that chapter to learn what the specific inputs are for boundary conditions such as “Slip Velocity and Temperature”, “Porous Wall” and others displayed in the **Special BC Data** section.

11.9 Buoyancy Sources

The **Buoyancy** section allows the user to include gravity effects. Gravity effects are most commonly seen in low-speed or stagnant flows with heat addition. Buoyancy forces come about by density variations in the flowfield. Lighter gas (lower density) rises while heavier gas (higher density) fall, thus causing free convection to occur. If forced convection (pressure driven flows) is present, buoyancy effects become negligible very quickly. The Buoyancy property section appears as shown in Fig. 11.11 on pg. 211.

By default, modeling of buoyancy forces is turned off. Turning on or off the buoyancy model is accomplished using the **Have Buoyancy** toggle selection. When the buoyancy model is off, all other options in the Buoyancy section are disabled.

GASpex supports two models for buoyancy. The models are selected using the **Buoyancy Method** option menu. With each model selection, the buoyancy source term used in the momentum equations is displayed to the right of the option menu. The first model is based on the actual solution specie density and is called **Flow Density (Eqn. of State)**. The source term for this model is $\vec{g}(\rho - \rho_{ref})$. If the density varies within the flow-field (density is not constant), then the user can use this option. The user must then input a reference density using the **Reference Density** input.

The other buoyancy model (**Thermal Expansion Coeff.**) assumes the flow is solved using a constant density solver (*i.e.*, incompressible Naiver-Stokes solver). The source term is $\rho\vec{g}\beta(T - T_{ref})$. In this case, the buoyancy force is computed using the thermal expansion coefficient for the fluid. The thermal expansion coefficient is input through the **Thermal Expansion Coeff** input (β in the source term). The user must also input a reference temperature with this model using the **Reference Temperature** input. The reference temperature is normally taken as the mean fluid temperature. The model using a thermal expansion coefficient may be used for variable density flows as well (*i.e.*, compressible flow flux options).

For each model, the user must input the gravity vector. This is done by entering the gravity value in the **Gravity Acceleration** input and the direction in which gravity is acting in the **Gravity Direction (inertial)** input. The gravity direction must be a unit vector. The user can force the direction vector to have a magnitude of unity by pressing the **Normalize** button.

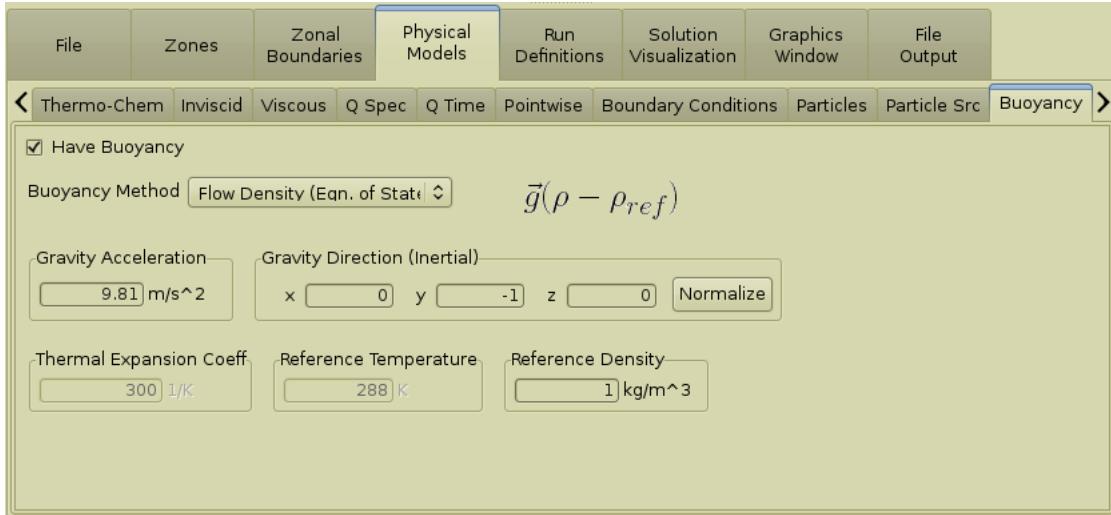


Figure 11.11: The Buoyancy tab for a fluid flow physical models.

11.10 Rotation Src

The **Rotation Src** tab was included for applications which involve rotation, but are solved for in a fixed frame of reference. Common applications are turbo machinery and helicopter rotor simulations.

In an inertial reference frame (*e.g.*, one in which the axis is moving with the body) the forces due to rotation are:

Coriolis: $-2\rho\vec{\omega} \times \vec{v}_{rel}$

Centrifugal: $-\rho\vec{\omega} \times (\vec{\omega} \times \vec{r})$

In the above equations, \vec{v}_{rel} is the relative velocity expressed as $\vec{v}_{rel} = \vec{v} - (\vec{\omega} \times \vec{r})$, \vec{r} is the distance vector, and $\vec{\omega}$ is the rotation vector. In *GASPEx* we solve for the absolute-flow variables (*e.g.*, a non-inertial frame) such that the momentum source term becomes

$$-\rho(\vec{\omega} \times \vec{v})$$

Therefore, when using the rotation source feature in *GASPEx*, the following things happen inside the code:

- Source terms are added to the momentum equations which account for the centrifugal force $(-\rho(\vec{\omega} \times \vec{v}))$.
- A wall velocity is added to the solid wall boundary conditions. This allows the impact of rotation to be taken into account in the boundary conditions.
- The grid metrics are modified to account for the “moving grid” effect. The grids are stationary inside *gaspex*, but the face metrics need to account for the simulated rotation.

To include the rotation source as described above, the user must select the **Have Rotation Source** option in the **Rotation Src** tab. This will in turn make the **Angular Rate** inputs available for the user to edit. The angular rates are the components of the rotation vector $\vec{\omega}$.

One final comment needs to be made on using a rotation source. If a rotation source is used along with an enforced wall rotation velocity (set inside the **Special BC Data** using the **Enforce Wall Rotation Velocity** option), the wall velocity will be set to the value used in the **Special BC Data** section, and not based on the **Angular Rate** input.

11.11 Coupled PhysMods

A Navier-Stokes solution may have coupling with a number of other physical models. In general, this is supported when the coupled solutions are located on the same grid system. A separate interpolation coupler must be used when solutions reside on different, overlapping grids. Not all couplers have a corresponding interpolation coupler at this time. A description of each possible coupling is given below.

Coupler For Spray This couples a Lagrangian spray physical model with the current Navier-Stokes model. A spray model is used to simulate particulates (either liquid or solids) entering and interacting with the flow field. Setting up a spray coupler will add source terms to the Navier-Stokes equations. That is, the impact of the spray on the fluid is modeled. For more information on modeling a Lagrangian Spray, see Chap. 13 beginning on pg. 225.

A physical model coupler is created using the **New** button and selecting one of the coupler types from the list that appears. The user must then select a physical model from the **Couple with Physical Model** input. The selected physical model should correspond to the type of coupler created.

11.12 User Src

A user defined source term can be added to the governing equations using the **User Src**. The source term is added on a zone by zone basis in which the user specifies a unique file for each zone. The contents of the source file must be consistent with the governing equations being solved.

To add a source to the model, the user first selects **New** from the source list. Each new source will correspond to one zone. If more than one zone needs a source term added, then use the **New** button to create additional sources. The **Cut**, **Copy**, and **Paste** work on the list entries in the same manner as other lists in the *GASPEX* GUI.

Once a new source entry has been created, select an individual zone from the tree view and press the **Change To Selected** button. This sets the source to the selected zone.

The actual source terms are provided to *GASPer* in the form of either a Plot3D, ASCII file or column data, ASCII file. The Plot3D file will follow the standard function file format for a single zone. The format is as follows:

```
nZone
iDim-1  jDim-1  kDim-1  nSoln
All source terms for nSoln=1 at cell centers
.
.
.
All source terms for nSoln=2 at cell centers
.
.
.
Repeat for remaining solutions in system
```

where “nZone” is the number of zones and should be set to 1 and “nSoln” is the number of solution variables for this physical model. The dimensional size of the zone is given by “iDim-1 jDim-1 kDim-1” which is the number of cell centers in each coordinate direction. The file format can be reviewed from the GUI by pressing the **File Format** button.

The column data format is intended to support the ASCII line output option from *GASPer*. This allows user defined variables to be used to create the source file from within the GUI. The format contains two header lines, followed by column data for each cell in the zone.

For the Navier-Stokes physical model, the source term should be given in metric SI units and have the corresponding units.

Mass Continuity Equations: kg/m^3

Momentum Equations: N/m^3

Non-Equilibrium Vibration: W/m^3

Energy Equation: W/m^3

Spalart-Allmaras Equation: $kg(m^2/sec)/(m^3sec)$

TKE Equation $J/(m^3sec)$

Dissipation of TKE (ϵ) Equation $kg(m^2/s^3)/(m^3sec)$

Turbulence Frequency (ω) Equation $kg(1/sec)/(m^3sec)$

Turbulence Length Equation $kg * m/(m^3sec)$

Turbulence Stress Equations $J/(m^3sec)$

11.13 Changing Species

The number of species can be changed by adjusting the **# of Species** setting or by changing the selected chemistry model in the **Thermo-Chem** tab. Changing either the specie name or the number of species will prompt the solution conversion dialog when a solution exists.

In general, *GASPEX* will check for matches between the old and new solution variables and copy specie values when matches are found. The check for a match is based on the specie name. If a new specie does not have a match from the old solution, the new specie value is set to zero. If one or more species are removed, or do not have a match during the conversion process, the mixture density is still preserved by normalizing the existing specie values. In this way the mixture density will always be preserved.

The user should never change chemistry models where there are no common specie names. For example, you should not go from the **Kang & Dunn** chemistry model to **Perfect Gas**, since the *Pgas* specie is not part of the **Kang & Dunn** model. There are two exceptions to this rule.

The first exception to this rule is in converting from a one specie model to another one specie model. In this situation, the specie values are copied over even though the specie names may differ. The second exception is when converting from the *Pgas* specie to a chemistry model which contains both *N₂* and *O₂* species. *GASPEX* has been programmed to break the *Pgas* density up so that the mole fraction of *N₂* is 0.79 and that of *O₂* is 0.21. In this way a user can begin a problem with perfect gas and then later switch over to an air chemistry model.

11.13.1 Changing Non-Equilibrium Vibrational Energies

The non-equilibrium vibrational energies can be changed inside the **Thermo-Chem** tab. The user can either add or remove non-equilibrium vibrational energies by adjusting the **# of Species with Non-Equilibrium Energies** setting.

When going from no vibrational energies to a specified number, the new variables will be filled with equilibrium values based on the initial **Q Spec** setting. When changing the vibrational energy terms where some exist in the solution, *GASPEX* will check for matches and copy these values over. Otherwise the vibrational energy is filled with initial equilibrium values.

11.13.2 Changing Turbulence Models

The turbulence model can be changed inside the **Viscous** tab. When going from laminar flow to using a multi-equation turbulence model, *GASPEX* will initialize the turbulence variables with the **Q Spec** information specified in the **Initialization** tab under **Zones**. If the user is going from a multi-equation turbulence model to a zero-equation model or laminar flow, then the turbulence variables are simply removed from the solution.

Any conversion from a two-equation model to another two-equation model will preserve

the turbulence kinetic energy and eddy viscosity values. If you wish to initialize the two-equation model to the **Q Spec**, then change the model to laminar first, and then switch it back to the desired model. Once the solution is converted, the old solution is deleted. Conversions which involve the one-equation turbulence model will always result in initializing the turbulence variables to **Q Spec**.

Chapter 12

Physical Model: Solid Thermodynamics

12.1 Introduction

The solid thermodynamics (solid heat transfer) physical model is used for computing the temperature distribution inside solid materials. The governing equation for this physical model is the energy equation which is used to model heat conduction within a solid. Temperature is the primitive (solution) variable.

The ST model can be used as a stand-alone solver or can be coupled to the Navier-Stokes solver to perform conjugate heat transfer (CHT). CHT is used when the heat transfer across a solid/liquid boundary is unknown and needs to be computed. This is commonly used when a flow field directly impacts the heating or cooling of a solid material. See Sec. 3.10 on pg. 46 for setting up a conjugate heat transfer problem.

The default physical model in *GASpex* is the Navier-Stokes model for fluid flows (see Chap. 11 beginning on pg. 165). To create a solid thermodynamics (ST) physical model, the user should use the **New** physical model command as explained in Sec. 10.4 on pg. 149. An example solid thermodynamics physical model is shown in Fig. 12.1 on pg. 218.

Note:

The solid thermodynamics physical model is currently not available for unstructured grids.

12.2 Solver Overview

The solid thermodynamics (ST) physical model is used to describe the heat conduction physics inside a solid material. The physical model is associated with the ST solver which computes the temperature field. The governing equation used by the ST solver is the energy equation. More details of the solver is provided in the *GASpex* Technical Reference Manual in the “Solid Thermodynamics Solver” chapter.

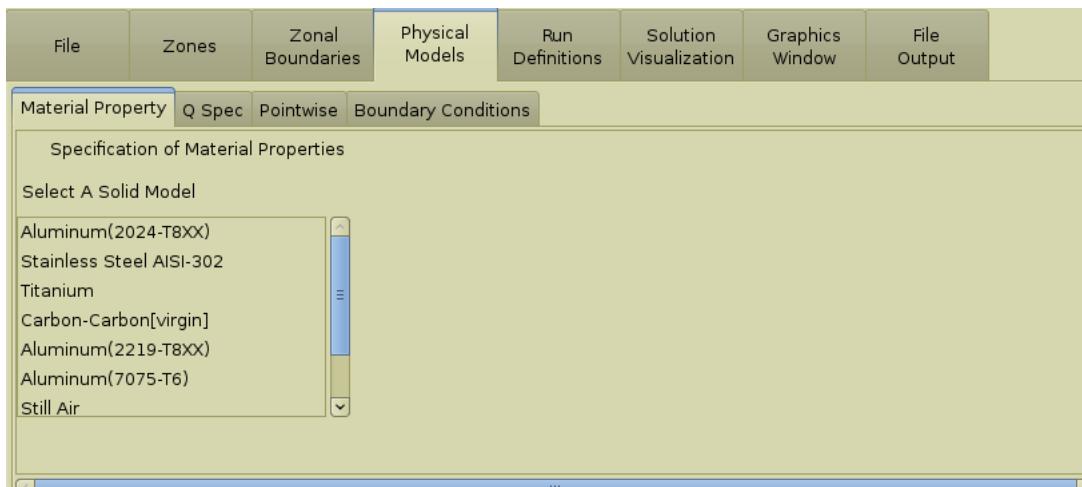


Figure 12.1: The solid thermodynamics (ST) physical model frame.

12.3 Material Property

The **Material Property** tab is used to select the type of material associated with the physical model. The material property section appears as shown in Fig. 12.2 on pg. 219. The material property selection is done by selecting a model displayed in the solid model list. Once the user selects a solid model, the selection is complete.

The displayed solid model list is taken from the *GASPE* database. There are several solid models that come with *GASPE*. If the user would like to add solid models to the existing list, the user will need to use the *GASPE* database GUI. Solid models can be added only through the database GUI (see Chap. 18 beginning on pg. 365) and are stored in the database XML file (the default database is `aerosoft/share/thermochemdb.tcd`).

When the physical model is first created, there will be no solid model selected from the list. If no solid model is selected, a default solid model is used with properties of unity (*i.e.*, density, thermal conductivity, and specific heat are all one). So it is important that the user select a solid model from the material property data.

12.4 Q Spec

For the solid thermodynamics physical model, a **Q Spec** data tab is available for setting solid data parameters. The data is used for solution initialization purposes or for use with boundary conditions. The default solid material **Q Spec** is shown in Fig. 12.3 on pg. 220.

The first frame for input is **Material Input Parameters**. Inputs here include the material temperature and the heat flux per unit area. The heat flux input is used for certain boundary conditions, while the material temperature represents the solution unknown (primitive variable) for solid conduction problems. The temperature input is used for boundary conditions as well as for solution initialization (one **Q Spec** is always selected for solution initialization

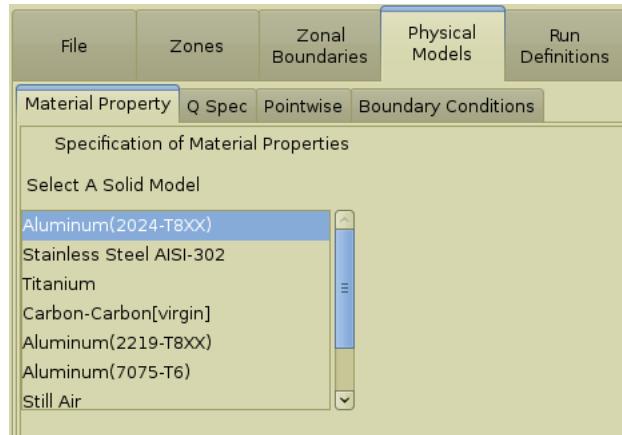


Figure 12.2: The Material Property tab for a solid thermodynamics physical model.

in each zone). Note that solution initialization is setup in the **Initialization** tab under **Zones** (see Sec. 8.2 on pg. 88).

The **Convection Input Parameters** are used with convection type boundary conditions. The equation for convective heat transfer to known surroundings is given in Fig. 12.4 on pg. 220. These boundary conditions rely on a heat transfer coefficient (h) and convective fluid temperature. These inputs are only used when convection effects are included through a BC. For example, the “**Convection**” boundary condition will use this information.

The **Radiation Input Parameters** are used with black-body radiation type boundary conditions (see Fig. 12.4 on pg. 220 for the radiation equation). For BC’s modeling radiation, the farfield temperature is required in order to determine the heat loss or gain due to radiation. The emissivity (e) is a material property which is specified in the database. An emissivity value of 1 indicates the solid acts like a blackbody in that it emits all incidence radiation.

The **Computed Parameters From Database** are for reference use only. The parameters in this frame include material density, specific heat, thermal conductivity and emissivity. Except for the emissivity, these parameters are functions of the temperature and are computed using the **Material Temperature** input. The values for these parameters are computed from the properties specified in the **GASPer** database.

12.5 Pointwise Data

The pointwise data is specific to the physical model type and the functionality will always be the same, regardless of physical model type. See Sec. 10.8 on pg. 153 for a general discussion of pointwise data.

For solid thermodynamic physical models, the pointwise data types include temperature and surface heat flux. The heat flux represents the thermal power per unit area into or out of the solid. A positive heat flux represents heat transfer into the solid, while a negative value represents a heat loss.

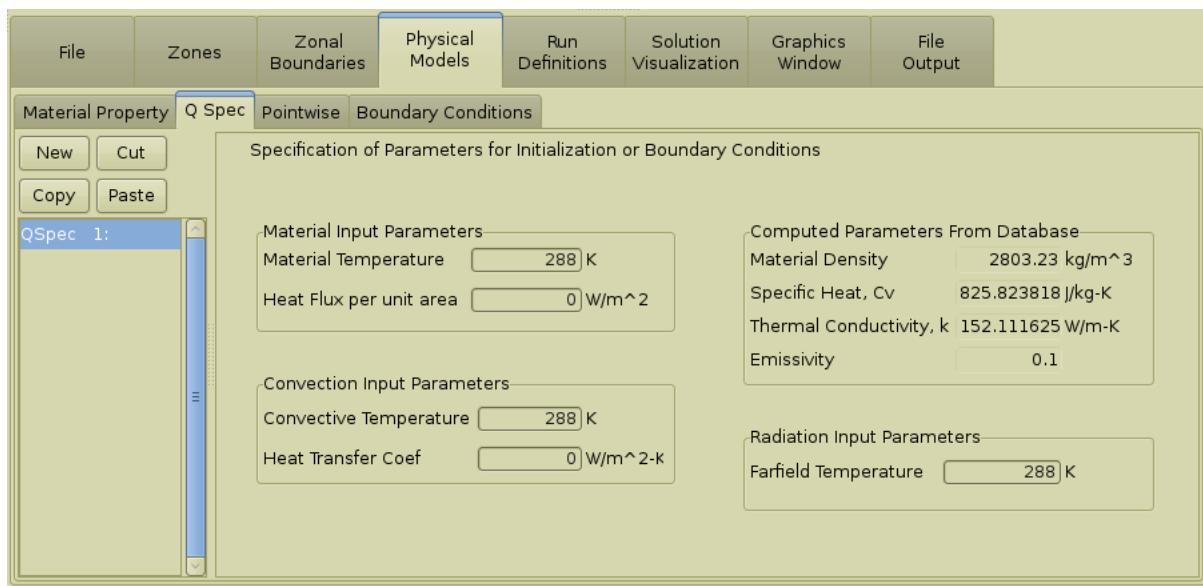


Figure 12.3: The Q Specification tab for solid materials with heat conduction.

$$\begin{aligned} \text{[Convection]} \quad q''_{wall} &= \bar{h}(T_{wall} - T_{convective}) \\ \text{[Radiation]} \quad q''_{wall} &= e\sigma(T_{wall}^4 - T_{farfield}^4) \end{aligned}$$

Figure 12.4: Equations for convection and radiation to known surroundings.

12.6 Boundary Conditions

The **Boundary Conditions** tab allows the user to assign boundary conditions to surface families. See Sec. 7.1.3 on pg. 80 for the details of setting up a BC surface family.

12.6.1 List of Solid Material Boundary Conditions

Boundary conditions for the heat conduction solver are different from other solvers in *GASpex*. So when setting up a solid thermodynamics physical model, the user will need to select from the list below.

Prescribed T Sets the boundary temperature as specified by the Q source. This is normally used to specify the solid wall temperature.

Prescribed Heat Flux Sets the boundary temperature in order to match a user specified heat flux. The heat flux is set by the Q source. This is used when the wall temperature is not known, but the heat flux through the wall is.

Adiabatic Wall This is a special case of the prescribed heat flux condition in which the heat flux through the boundary is zero.

Convection A convective heat transfer coefficient and convective temperature are specified by the user through the **Q Spec**. A heat flux at each boundary face is then computed and set as in the Prescribed Heat Flux BC. This condition is used to simulate convective cooling or heating without actually modeling the fluid flow. See Fig. 12.4 on pg. 220 for the expression used to model convection.

Radiation Heat transfer at the boundary due to black body radiation is modeled with this boundary condition. The user specifies the surrounding farfield temperature (given in the **Q Spec**) and the material emissivity (set in the database). If the farfield temperature is greater than the surface temperature, there is heat transfer into the surface, while a lower farfield temperature represents a energy loss at the surface. See Fig. 12.4 on pg. 220 for the expression used to model radiation.

Heat Flux + Convection + Radiation This boundary condition is a combination of the Prescribed Heat Flux, Convection, and Radiation conditions described above.

Extrapolate T Performs a first order extrapolation from the interior cells in order to set the boundary value for temperature. This condition is normally used for the side walls of a two-dimensional problem.

Extrapolate Heat Flux This boundary condition tries to make the boundary heat flux equal to the heat flux of the first interior cell face.

Symmetry Assumes the boundary surface is a symmetry plane. The temperature profiles in the ghost cells are therefore a reflection of the interior cells.

Fluid/Solid Interface For conjugate heat transfer (CHT) problems, this boundary condition sets the energy balance for heat transfer between the solid and flow. The energy balance includes the Prescribed Heat Flux specified in the Q Source as well as radiation effects (as applied in the Radiation BC). This boundary condition should only be used on zonal boundary surfaces in which one surface belongs to a fluid zone and the other belongs to a solid zone. Recall that a zonal boundary can be overridden as a BC inside the Pt2Pt ZB folder of the tree view. See Sec. 3.10 on pg. 46 for setting up a CHT problem.

Solid/Solid Interface For conductive heat transfer problems in which two materials are used with different thermal conductivities. In this case, the surfaces between the two materials should be setup as a zonal boundary, and then set to this boundary condition in order to compute the correct heat flux through the zonal boundary. Recall that a zonal boundary can be overridden as a BC inside the Pt2Pt ZB folder of the tree view.

Zonal Boundary Interface This boundary condition assumes the surface is part of a zonal boundary and that it receives information from the adjacent zone. The boundary condition ensures that the information passed across a zonal boundary is not overwritten.

12.7 User Src

When running the solid thermodynamics model, a user defined source term can be added to the solid energy equation using the **User Src**. The source term is added on a zone by zone basis in which the user specifies a unique file for each zone. The contents of the source file must be consistent with the governing equations being solved.

To add a source to the model, the user first selects **New** from the source list. Each new source will correspond to one zone. If more than one zone needs a source term added, then use the **New** button to create additional sources. The **Cut**, **Copy**, and **Paste** work on the list entries in the same manner as other lists in the *GASPE* GUI.

Once a new source entry has been created, select an individual zone from the tree view and press the **Change To Selected** button. This sets the source to the selected zone.

The actual source terms are provided to *GASPE* in the form of either a Plot3D, ASCII file or column data, ASCII file. The Plot3D file will follow the standard function file format for a single zone. The format is as follows:

```
nZone
iDim-1  jDim-1  kDim-1    nSoln
All source terms for energy equation at cell centers
.
.
.
```

where “nZone” is the number of zones and should be set to 1 and “nSoln” is the number of solution variables for this physical model (nSoln = 1 for solid thermodynamics)). The dimensional size of the zone is given by “iDim-1 jDim-1 kDim-1” which is the number of cell centers in each coordinate direction. The file format can be reviewed from the GUI by pressing the **File Format** button.

The column data format is intended to support the ASCII line output option from *GASPE*. This allows user defined variables to be used to create the source file from within the GUI. The format contains two header lines, followed by column data for each cell in the zone.

For the solid thermodynamic physical model, only a single equation is solved and corresponds to the conservation of energy. The source term is therefore added to the energy equation and should be given in metric SI units of W/m^3 .

Chapter 13

Physical Model: Lagrange Spray

13.1 Introduction

The Lagrange spray physical model is used to model multi-phase, particle flow in *GASPer*. A brief discussion of multi-phase is given here, followed by specific details for each of the modeling tabs.

This physical model is intended to be coupled with a Navier-Stokes model. The NS model provides the flow field in which the solid or liquid particles interact.

The default physical model in *GASPer* is the Navier-Stokes model for fluid flows (see Chap. 11 beginning on pg. 165). To create a spray physical model, the user should use the **New** physical model command as explained in Sec. 10.4 on pg. 149.

Note:

The spray physical model is currently not available for unstructured grids.

13.2 Solver Overview

In the context of the *GASPer* flow solver, multi-phase flow consists of a mixture of carrier fluid (generally a gas) and particulate (either solid particles or liquid droplets). The carrier phase is computed using an Eulerian analysis via the Navier-Stokes fluid flow model. That is, *GASPer* assumes the carrier fluid is a continuum. The particulate phase is treated as Lagrangian (a discrete number of particles).

The multi-phase implementation in *GASPer* is derived from an Euler-Lagrange approach, also known as a *discrete-element model*. Here, the trajectory of the particulate phase is analyzed using Newton's law of motion for each particle or collection of particles. The position, velocity, size and temperature are known along a certain path through the domain. *GASPer* implements the Particle Trajectory Approach, where the flow field is held constant while the particle trajectory is computed until the particle either leaves the domain, evaporates to negligible size, or reaches a prescribed time step. The inter-phase

mass, momentum and energy transfer are computed as each group of particles is integrated through an individual fluid control volume.

The Lagrange spray physical model is used to describe solid or liquid particles that are released into a flow field. The physical model provides data to the Lagrange particle solver which computes the path of each particle and keeps track of properties such as the temperature, velocity, size, and mass. More details of this solver is provided in the *GASpex* Technical Reference Manual in the “Particulate Modeling” chapter.

13.3 Boundary Conditions

The **Boundary Conditions** tab allows the user to assign boundary conditions to surface families. See Sec. 7.1.3 on pg. 80 for setting up a BC surface family.

For the spray model, there is no **Q Source** option. Therefore the inputs for Q Source, Q Spec and Q Time are not selectable.

13.3.1 List of Spray Boundary Conditions

Boundary conditions for the Lagrange spray solver are different from other solvers in *GASpex*. So when setting up a spray physical model, the user will need to select from the list below.

Inflow/Out The surface is assumed to be open such that particles can enter or leave freely.

Symmetry The surface is assumed to be a symmetry plane. This BC is implemented the same as a wall with pure reflection (see next BC).

Wall (Pure Reflection) The surface is assumed to be a solid wall. If a particle collides with the wall, it will reflect off the wall without any losses. The angle of reflection is computed directly from the angle of incidence and the particle speed will remain the same.

Wall (Tabakoff Reflection) The surface is assumed to be a solid wall. If a particle collides with the wall, the collision is based on the work of Tabakoff [53] and may be more realistic than a pure reflection. For this BC, particles will have some sort of energy loss due to wall contact. This will result in a different speed and angle of reflection than the pure reflection case.

13.4 Coupled PhysMods

In most all cases a Lagrange spray solution will have coupling with a Navier-Stokes physical model. At this time, this is the only available coupler for the spray model. The coupling is described below.

Coupler For Fluid Flow (N-S) This couples a Navier-Stokes physical model with the current Lagrangian spray model. Setting up a fluid flow coupler will allow the particles to respond to the carrier fluid (*i.e.*, the Navier-Stokes solution). This allows the impact of the fluid on the particles to be modeled, which includes conservation of energy and momentum of the particles. Without the coupler, the particles would not change direction or energy. Note that a corresponding coupler should be set up in the Navier-Stokes model in order to take into account the impact of the particles on the fluid flow.

13.5 Particle Src

The **Particle Src** tab allows the user to define where particles emerge in the flow, as well as the initial particle characteristics. The particles must originate from some location in the flow domain, which in *GASPer* will be a surface group that the user selects. If more than one source exists, the user has the flexibility to define multiple particle sources.

On the left-hand side of the **Particle Src** section is located the select list (see Fig. 13.1 on pg. 229 for a sample **Particle Src**). Using the **New**, **Copy**, **Cut**, and **Paste** options the user can manage multiple particle sources. For example, a new source is added to the select list by pressing the **New** option. A particle source can be removed and placed on the clip board using the **Cut** option. **Paste** will create a new source identical to the one in the clip board, while **Copy** will make a copy of the particle source and place it in the clip board.

The **Temporal Injection Type** describes how a particle enters the computational domain with respect to time. For steady state problems, the user specifies how much time particles are allowed to travel during the iteration cycle, also called the Lagrangian integration period. For steady flow simulations, the integration period is set in the solver info of the run definition (see Sec. 15.4.4 on pg. 256 for setting the integration or update time). For time accurate problems, the integration period is equal to the physical time step. In either case, particles are assigned a time that corresponds to a value between the start and end times for that cycle (integration period). The user has the choice of releasing particles all at once at the start time, or randomly throughout the travel time.

For each particle source the user creates, a surface family must be assigned. This is done by selecting a surface folder (not the actual surface, but a folder) in the tree view. With the surface folder selected (highlighted), the user must press the **Change To Selected** button under the **Surface Family** header. When the flow solver is run, *GASPer* will model particles originating (injecting) from all surfaces inside the selected folder.

The **Particle Seed Location** specifies how particles are distributed on the surface where they are being injected from. The user may want to randomly place the particles or use an ordered method. When **Random** is selected, a random number generator is used to specify the starting location for each particle. Each time particles are released the location will be random such that there is no repetition. With the random option, the user will set the number of clusters to be released each integration period (which normally occurs each iteration cycle). This is done using the **Clusters/Step** input, which is typically in the

thousands. Releasing more clusters per step will increase accuracy, but at the cost of more computational work.

The **Ordered** option will cause the same number of particles to be released from each surface. The user must then set the **Clusters/Face** input. In order to evenly distribute particles over each face, the **Clusters/Face** input represents a matrix distribution. The default value of 1 indicates that a single cluster is released from the center of each face. A value of 2 corresponds to a 2×2 distribution (total of 4 clusters per face). Similarly, a value of 3 would be a 3×3 distribution (total of 9 clusters per face). Using an ordered distribution will cause the particle tracking to be repeatable since clusters will be released from the same location each step.

The **Particle Size At Injection** allows the user to select the initial particle size distribution. The options here are **Constant** and **Gaussian**. The constant option will set all particle size diameters equal to the Sauter Mean Diameter (**smd**) input value at the time of injection. The Gaussian option will vary the particle sizes using a Gaussian curve with the Sauter Mean Diameter as the average.

The **Particle Injection Pattern** allows the user to select particle angles upon injection. The options here are **Uniform** and **Spray**. The uniform option sets the direction for all particles entering the surface the same. The actual direction of the particles is set using the **Injection Direction**.

For the spray option, the particles are directed primarily in the **Injection Direction**, but with some variation in order to produce a “spray” type pattern. The maximum half angle of the pattern is user specified using the **Half Angle** input. While the maximum angle is set, the actual angle will vary between 0 and the half angle. The angle for each cluster is randomly set, preventing the spray pattern from being repeatable each step.

As mentioned above, the **Injection Direction** specifies the direction cosines of the particles as they are injected or released into the computational domain. The **Half Angle** input is only used with the spray injection pattern.

The remaining inputs are done under the **Particle Properties** heading. The inputs are:

Velocity Mag. This input is for the initial velocity magnitude of the injected particles. The direction is set based on the **Particle Injection Pattern**.

Temperature Temperature of particles as they enter the flow domain. After entering the computational domain, the local particle temperature may vary as it interacts with the surrounding flow.

smd The Sauter Mean Diameter is a characteristic length used to describe the distribution of particles in a spray. In defining the Sauter Mean Diameter, consider that an atomizing spray includes a range of droplet sizes. Sauter Mean Diameter is one way to express the average droplet size in terms of the average ratio of volume to surface area of the droplets. Since it deals with surface area, Sauter Mean Diameter is a good way to describe a spray that is to be used for processes involving evaporation. Sauter Mean Diameter is the diameter of a hypothetical droplet whose ratio of volume to surface

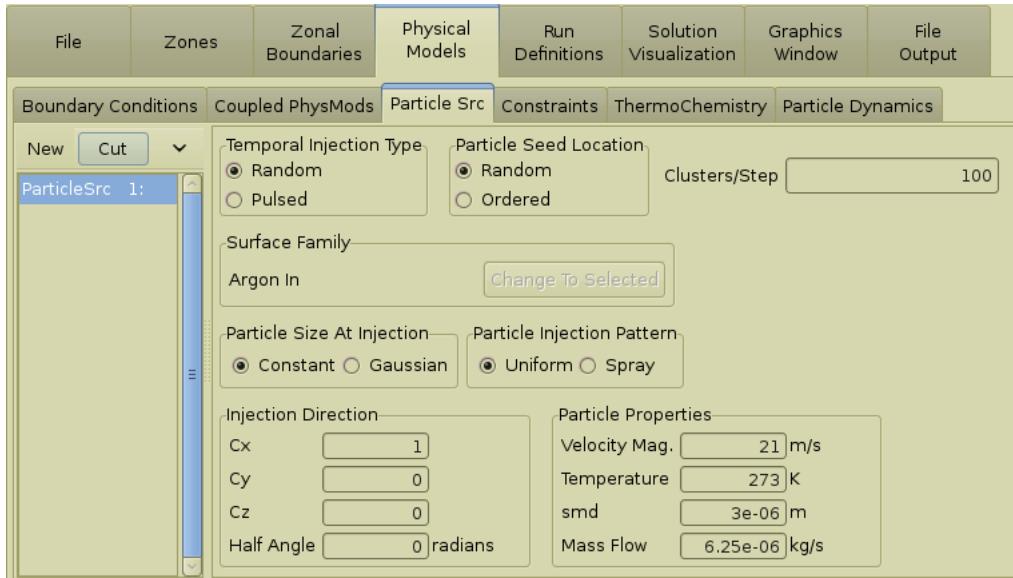


Figure 13.1: The Particle Src tab used for setting up source information for particles in multi-phase flow applications.

area is equal to that of the entire spray. Each cluster of particles integrated in *GASPE*x is allowed to have a different size and mass. The size of each particle cluster is assigned according to a probability density function (PDF) which when integrated, will have the given SMD.

Mass Flow The amount of mass entering the domain per time for the current source of particles.

Note: The number of particles per cluster will be calculated as a function of the number of clusters per step, the particle mass flow, the particle size/mass, and the update time (steady state) or physical time (time accurate).

13.6 Constraints

There are several constraint options for modeling particles. The constraints are normally enforced when solving on two-dimensional or axi-symmetric grids. The first constraint limits the path of particles along a specific grid coordinate. This constraint is found under the **Logical Constraint** header. The default constraint is **None**, which removes all grid coordinate constraints. Selecting for example the **I-Direction** constraint will prevent particles from moving in the *i*-direction.

The second constraint allows the user to constrain particle movement in physical (XYZ) space. Under the header **Physical Constraint**, the user can select a constraint in one of the

three primary axis. For example, selecting the **X-Direction** constraint will prevent particles from moving in the *x*-direction.

13.7 ThermoChemistry

In this section the user selects the type of particles to be modeled. The two choices are liquid particles and solid particles. If the **Liquid** selection is made, a liquid model should be selected from the **Liquid Models** list. If the **Solid** selection is made, a corresponding solid model needs to be selected.

Under the spray type, the liquid and solid lists are provided. Select the appropriate tab to display the corresponding list. Each list contains the available models that are read in from the *GAS*Pex database, which contains a number of models by default. The user may view, edit, add or remove liquid and solid models using the *GAS*Pex database GUI. This list cannot be edited from the **ThermoChemistry** tab.

At the bottom of the **Liquid Models** frame is an option for evaporation. If the liquid particle can undergo evaporation and the user would like to model this physical phenomena, then the **Evaporation** toggle should be turned on. There are several restrictions in using the evaporation option. First, the particle type must be a liquid. Second, the name of the liquid model must correspond to a gas species name in the coupled Navier-Stokes chemistry model (selected in the Navier-Stokes **Thermo-Chem** tab). When a particle evaporates, the liquid becomes a gas and the mass gets transferred over to the equivalent gas species assuming a coupler is being used (see Sec. 13.4 on pg. 226). *GAS*Pex matches the liquid particle to the gas species using the liquid model name. The third requirement for using evaporation is valid vapor pressure and heat of vaporization data for the liquid model. This data must be entered and stored in the *GAS*Pex database for the liquid model.

The **Solid Models** list are the same ones used for the solid thermodynamics (heat conduction) solver. With a solid particle, evaporation is not a modeling option.

When the spray model is first created, there are no selected liquid or solid models. The user must select a model by clicking on one of the model names in the list. If a model is not selected, the flow solver will stop with a warning upon running. The user may change models by simply clicking on another model name.

The last piece of data the user must select is the **Density Model**. For both solid and liquid particles, the database must contain a density curve fit. There are two curve fits available for density: NSRDS and LSRC. The NSRDS is a curve fit from the National Standard Reference Data System. Data for the curve fit can be found in the public domain. The LSRC is a least squares curve fit used for curve fitting user defined data. Only curve fits with data in the database are selectable.

13.8 Particle Dynamics

There are currently two drag models implemented for the particulates. The first is based on Stokes flow (*i.e.*, very low Reynolds number) and assumes the particle is traveling with a velocity near the carrier gas. This is normally the case for most spray injections.

The second is a model by Loth [54] which accounts for compressible and non-continuum effects. This should be used when the particulates are travelling at a velocity which is much different than the carrier gas.

If the user wants to model the impact of gravity on the particles, then the **Have Gravity** option should be selected. The user must then input the gravity acceleration and normalized force vector (input through the **gx**, **gy**, **gz** inputs).

13.9 Output Files

When the Lagrange spray solver is run, an output file is automatically generated called **sprayLgrInfo.dat**. This output file contains information about the particle tracking where the contents of the file are as follows.

Time The physical time for simulations run with temporal accuracy.

Cycle This is the current iteration or cycle number.

Mass In (kg) This is the total amount of mass in kilograms that the particles entering the computational system represent.

Clusters In This is the total number of clusters injected into the grid domain. Recall that particles are grouped into clusters, and its the clusters that are tracked by the solver.

Clusters Out This is the total number of clusters that exited the grid domain. Note that some clusters evaporate (liquid particles) or entering into a recirculating loop and may never exit the grid domain.

Clusters Removed The number of clusters removed from the system due to evaporation, a recirculation loop, or numerical error.

Chapter 14

Physical Model: Mechanical Stress

14.1 Introduction

The mechanical stress (MS) physical model is used for computing the stress and strain inside solid materials, as well as any displacements that may occur. The mechanical stress can be used to estimate stress fracture limits and contributions to the refractive index.

The MS model can be used as a stand-alone solver in which body forces are applied to create force induced stress. The model can also be coupled to the solid thermodynamics solver to compute thermally induced stresses.

The default physical model in *GASPer* is the Navier-Stokes model for fluid flows (see Chap. 11 beginning on pg. 165). To create a mechanical stress physical model, the user should use the **New** physical model command as explained in Sec. 10.4 on pg. 149.

Note:

The mechanical stress physical model is currently not available for unstructured grids.

14.2 Solver Overview

The mechanical stress (MS) model in *GASPer* is used to compute the stress and strain that arises in a solid material due to either external body forces or internal thermal gradients. An assumption made about the material is that it is assumed to be isotropic with the following elastic stress model.

$$\sigma_{ij} = \lambda e_{kk} \delta_{ij} + 2G e_{ij} - (3\lambda + 2G)\alpha \delta_{ij}(T - T_O)$$

The above expression relates stress and strain where G is the shear modulus, λ is Lame's constant, α is the temperature dependent thermal expansion coefficient, T_O is a baseline temperature, and T is the local temperature. With the exception of the material temperature, all other parameters are taken from the database solid model. A view of the stress property inputs located in the database is shown in Fig. 14.1 on pg. 234.

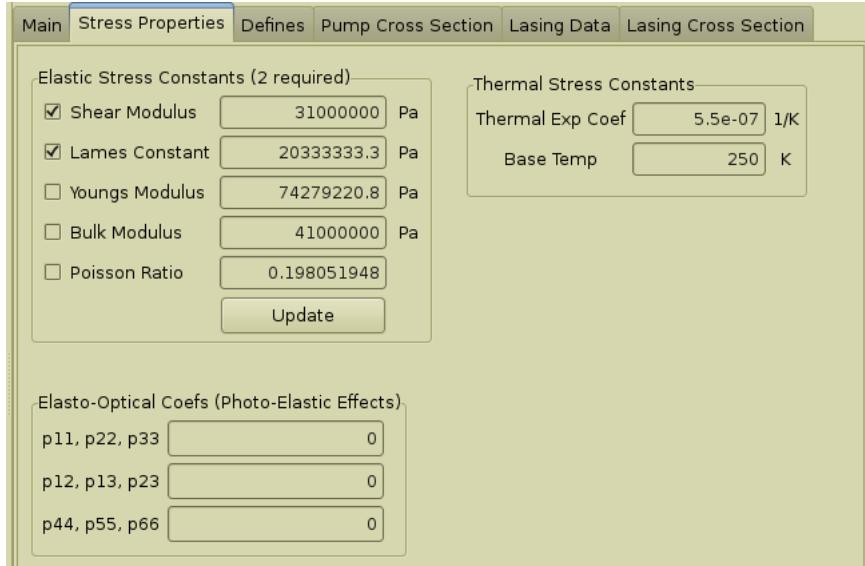


Figure 14.1: Stress properties for the database solid model (accessed using the *GASPer* Database GUI).

The strain is assumed to be small and dependent only on the stress applied to it. Making this assumption gives the following relation between the strain tensor (e_{ij}) and the displacement vector (a_i).

$$e_{ij} = \frac{1}{2} \left[\frac{\partial a_i}{\partial x_j} + \frac{\partial a_j}{\partial x_i} \right]$$

The governing equations for the mechanical stress consist of a three equation system which are solved directly for the material displacements. Once the displacements are known, the above expressions are used to compute the strain and stress.

The current stress formulation only supports steady state calculations. In order to perform time dependent simulations, additional equations for the rate of change (material velocity) would need to be solved.

14.3 Material Property

The **Material Property** tab is used to select the type of material associated with the physical model. The material property section appears as shown in Fig. 14.2 on pg. 235. The material property selection is done by selecting a model displayed in the solid model list. Once the user selects a solid model, the selection is complete.

The displayed solid model list is taken from the *GASPer* database. There are several solid models that come with *GASPer*. If the user would like to add solid models to the existing list, the user will need to use the *GASPer* database GUI. Solid models can be added

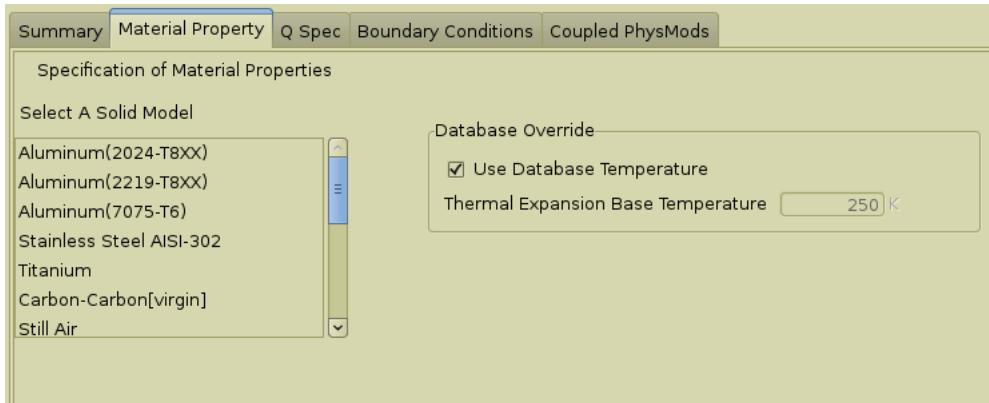


Figure 14.2: The Material Property tab for a mechanical stress physical model.

only through the database GUI (see Chap. 18 beginning on pg. 365) and are stored in the database file (the default database is `aerosoft/share/thermochemdb.tcd`).

When solving for the stress due to thermal gradients, the thermal expansion coefficient along with a base temperature are utilized. These parameters are part of the solid model and stored in the database. The user may desire to change the base temperature (T_O) value at which the thermal expansion is neutralized. Instead of modifying the database, this can be done from the GUI by turning off the **Use Database Temperature** option and entering a new thermal expansion base temperature. This allows for problem specific applications where the base material temperature may differ from that set in the database.

When the physical model is first created, there will be no solid model selected from the list. If no solid model is selected, a default solid model is used with properties of unity (*i.e.*, density, thermal conductivity, and specific heat are all one). So it is important that the user select a solid model from the material property data.

14.4 Q Spec

For the mechanical stress physical model, a **Q Spec** data tab is available for setting stress related parameters. The data in this tab is used for solution initialization purposes or for boundary conditions. The default **Q Spec** for mechanical stress is shown in Fig. 14.3 on pg. 236.

The **Displacements** frame allows the user to input material displacements in the three coordinate directions. This data is used for both solution initialization and for select boundary conditions such as “**Prescribed Q**”.

When using the “**Prescribed Force**” boundary condition, the user must provide the applied force. This is done using the **Boundary Force/area** inputs. In this frame, the user inputs the force per unit area in each of the three coordinate directions. This is then used to compute the local force on a surface.

The stress model includes the effects of thermal expansion and requires the material

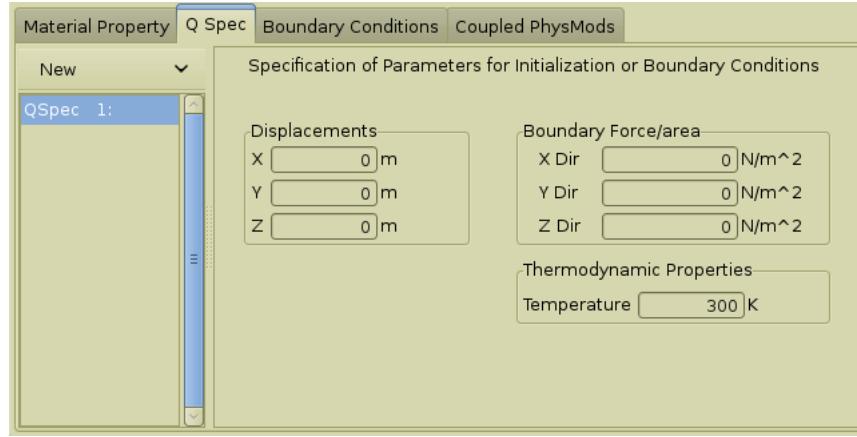


Figure 14.3: The Q Specification tab for mechanical stress model.

temperature. If the physical model is coupled to a solid thermodynamics solver (see Sec. 14.6 on pg. 237 for details concerning the coupler), the temperature is passed into the solver and is not provided by the user. Otherwise, the material temperature is set using the **Temperature** input located inside the thermodynamic properties frame. When providing the temperature, it is assumed to be constant over the entire material.

14.5 Boundary Conditions

The **Boundary Conditions** tab allows the user to assign boundary conditions to surface families. See Sec. 7.1.3 on pg. 80 for setting up a BC surface family.

14.5.1 List of Solid Material Boundary Conditions

Boundary conditions for the mechanical stress solver are different from other solvers in *GASPEX*. So when setting up a MS physical model, the user will need to select from the list below.

Prescribed Q Sets the boundary displacements as specified by the Q source (which in this model is the Q Spec).

Prescribed Force Applies boundary forces on the surface with the magnitude and direction based on the **Boundary Force/area** inputs located in the Q Spec.

1st Order Extrapolation Does a first order extrapolation from the interior cells for the boundary values. This is normally used for an unconstrained boundary.

Symmetry Assumes the boundary surface is a symmetry plane. The displacement profiles in the ghost cells are therefore a reflection of the interior cells.

14.6 Coupled PhysMods

The stress solver may be coupled with another solver that provides the solid temperature. By doing this the stress induced by thermal gradients are included in the solution. Without a coupler, the temperature is assumed to be constant. A description of the possible coupling is given below.

Coupler For Solid Thermodynamics This couples a solid thermodynamics field to the current solid model. In order to include thermal stress, a temperature field is required. By coupling this solver with an ST physical model the temperature data is provided.

Chapter 15

The Run Definitions Frame

The **Run Definitions** frame of *GASPer* is where the user defines how a problem is solved. Information such as which sequence and physical model to use, the time-integration scheme to use, and parallel settings are defined here. A sample **Run Definitions** frame is shown in Fig. 15.1 on pg. 239.

15.1 Run Definitions Select List

For simple problems there may only be one run definition, but in most cases the user will have multiple run definitions. A list of the run definitions for the problem are displayed in the **Run Definitions** select list located in the lower left corner of the GUI. An example of a

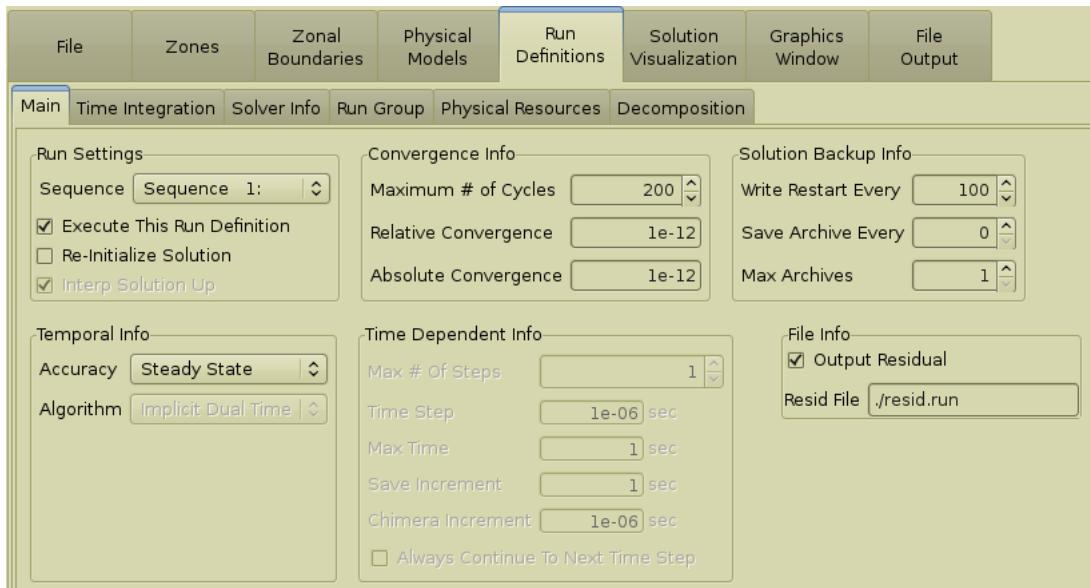


Figure 15.1: The **Run Definitions** frame of the *GASPer* GUI.

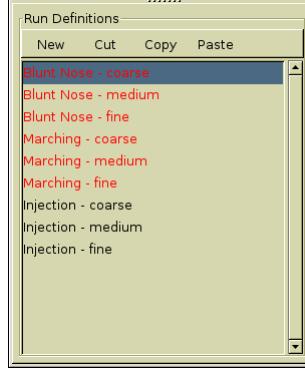


Figure 15.2: An example **Run Definitions** select list.

Run Definitions select list is displayed in Fig. 15.2 on pg. 240.

The example **Run Definitions** select list contains a total of nine run definitions and describes how to run a three-zone, three-sequence problem. The first three run definitions describe how to solve the first zone on each sequence level. The next three describe the second zone and the remaining three run definitions describe the third and final zone. The second zone is solved using space marching, while the first and third zones are solved globally. Because of this, a separate run definition was created for each zone.

A run definition is displayed in the current tab of the **Run Definitions** frame when it has been selected. The selected run definition will be high-lighted in the select list. The user may change the selected run definition by clicking with the left mouse button.

The description of each run definition in the select list can be changed or edited by double clicking on the text. Once in edit mode, the user may make desired changes by typing in new text.

The run definitions in the select list are color coded. A run definition that has red text signifies that the run definition will be executed by the *GAS*Pex flow solver (when using the auto solver). Black text means that the run definition will not be executed. See Sec. 15.2 on pg. 241 for more information on how to set a run definition for execution.

15.1.1 Creating a New Run Definition

By default a run definition will exist when you first create a new input deck. If additional run definitions are needed, one way to create a new one is using the **New** button located at the top of the **Run Definitions** select list. Pressing the **New** button will create a new run definition consisting of default values and settings. The new run definition will be added to the list following the selected run definition. Note that there must always be a run definition selected.

15.1.2 Cutting a Run Definition

To delete or remove a run definition, select the run definition from the select list and press the **Cut** button. This will remove the run definition from the list and place it in the clipboard. The run definition can be recovered using the **Paste** button. Using the **Cut** and **Paste** buttons in this fashion allows you to re-order the run definitions in the select list.

15.1.3 Copying a Run Definition

Pressing the **Copy** button will place a copy of the selected run definition in the clip-board. Both the copy and cut procedure will overwrite the contents in the clip-board.

15.1.4 Pasting a Run Definition

If a run definition exists in the clip-board, then pressing the **Paste** button will add the run definition to the select list. A copy of the run definition will still remain in the clip-board after a paste.

15.2 Main Run Definition Tab

The first tab under the **Run Definitions** frame is called **Main**. This tab contains general information about the run definition like temporal accuracy, sequence selection, and convergence information. Elements of the **Main** tab shown in Fig. 15.1 on pg. 239 are discussed below.

15.2.1 Run Settings

Sequence Selection

The **Sequence** option menu allows you to select which sequence level to execute. When there are multiple sequences, a new run definition is needed for each sequence level. Otherwise, the user must change the **Sequence** setting for each restart and have only a single run definition. See Sec. 3.7 on pg. 42 for a definition of a sequence in *GASPerx*.

A global problem (one that is not space marched) will normally have three sequences. The first sequence will represent the original grid, which is commonly called the “fine” grid. The two additional sequences are created using the sequence feature and are referred to as the “medium” and “coarse” grids. With these three sequences, the user will normally set the first run definition to the coarse grid. The user will then create a second run definition for the medium grid, and a third for the fine grid. *GASPerx* will execute each run definition in order, so the run definitions for the coarse grid are normally first in the list.

Executing the Run Definition

The next input selection under the **Run Settings** label deals with execution of the run definition. Since a problem may contain multiple run definitions (listed in the **Run Definitions** select list), you may decide to execute one or more of the run definitions each time the flow solver is called. To execute a run definition, the **Execute This Run Definition** check box needs to be selected. When a run definition is selected for execution, the title of the run definition in the select list will appear red. In this way you can see which run definitions will be executed by looking at the select list. This flag only applies to the auto launch method for the flow solver (`gaspx --solve`). See Sec. 3.3.2 on pg. 32 for more information on the auto launch method.

When the GUI is started, the first run definition that is selected to execute will automatically be displayed. In this way the user may use the selection as a way to control which run definition to display upon startup (which also sets which sequence is shown in the graphics window).

Re-initializing the Solution

The next flag under the **Run Settings** label is **Re-Initialize Solution**. This flag is used to control solution initialization. Each run definition has a sequence level and group of associated zones. If this flag is active, the solution on just those zones (and sequence level) will be initialized when the flow solver starts up. The solution is initialized according to the settings in the **Zones/Initialization** section. In this way, a user may restart the simulation for a set of zones through the flow solver. For example, after running a multi-zone problem for several cycles, a user may want to re-initialize the flow with more conservative run settings. Activating this flag accomplishes the task. Conversely, if a user would like to begin with the past solution, make sure this flag is de-activated.

Solution initialization may also be done from within the GUI. Re-initialization of individual or a group of zones can be done through the **Soln Mgt** window inside of the **Zones** section. (See Sec. 8.5 on pg. 96 for more details).

If the flow solver is being started for the first time and a solution does not exist, the solver will automatically perform the initialization. With the **Re-Initialize Solution** flag turned off, the flow solver will restart the simulation using the existing solution file. If a solution for the simulated set of zones does not exist, the flow solver will create a solution based on the initialization settings. By default the **Re-Initialize Solution** flag is turned off since it is not necessary when a CFD simulation begins.

Interpolating Between Sequences

The last flag in the **Run Settings** is for solution interpolation. By default this flag is turned on. When a run definition is operating on a sequence other than the fine grid level, the solution will be interpolated from the current sequence level to the next sequence level. This operation takes place inside the flow solver at the end of the run. The solution will only be

interpolated up to the next sequence level. For example, if the run definition is operating on the third sequence level, the solution will be interpolated up to the second sequence level. If the run definition is operating on the original grid (first sequence level), then this option will not be selectable by the user and no interpolations will occur. Solution interpolations may also be performed using the GUI inside the **Soln Mgt** section located in **Zones**.

15.2.2 Convergence Information

For a steady-state run, *GASPer* will execute the run definition until all the cycles have been completed or a residual tolerance has been met. For an implicit time-accurate run, *GASPer* will compute the solution at a time step until all the cycles are run or tolerances are met. The number of cycles (or iterations) is specified in the **Maximum # of Cycles** type-in box. A cycle is complete when each run group in the run definition is executed once.

The relative and absolute residual tolerances are specified in the **Relative Convergence** and **Absolute Convergence** type-in boxes respectively. The relative convergence is checked against the residual's root mean square (rms), after being normalized with the first residual's rms on that grid sequence. The absolute convergence is checked against the un-normalized rms residual. These tolerances are checked every run cycle. Cycles on a plane or a zone will continue until all the cycles are completed or one of the two tolerances are met.

In summary, the maximum cycle number is used differently depending on the following situations.

Steady State Cycle is the max number of iterations that will be performed if tolerances are not reached.

Implicit Time Accurate Cycle is the max number of iterations to perform each physical time step if tolerances are not reached. Implicit time accurate methods include Dual-Time Stepping and Crank Nicolson Time Stepping.

15.2.3 Solution Backup Information

The solution is written to the binary file after a specified number of cycles have been performed. This number is entered through the **Write Restart Every** type-in box. In this way, an intermediate solution can be saved to file to prevent the total loss of data in case program execution stops.

For steady state runs, this number is based on the cycle count as specified in the **Maximum # of Cycles**. In the case of transient or time-dependent runs, the number is based on the time dependent **Maximum # of Steps**.

If the user wants to save the solution and residual files for archiving purposes, this can be done using the **Save Archive Every** and **Max Archives** inputs. The **Save Archive Every** is similar to **Write Restart Every**, except that it controls how often to archive. An archive is saved inside the solutions directory in a sub-directory of its own. The maximum number of

archives to save is specified with the **Max Archives**. Once the maximum number of archives is reached, the next archive will over-write the first archive performed and so on.

The user may recover information from the archive directories either manually or using the GUI. Recovery from within the GUI is done from the File menu bar by selecting the **Recover Archive** option (see Sec. 4.2.8 on pg. 61).

15.2.4 Temporal Information

Temporal Accuracy

In a run definition, the problem is solved either as a steady-state flow or a time-dependent flow. This choice is made using the **Temporal Accuracy** option menu. The options for this menu are: **Steady State**, **1st Order**, **2nd Order**, **3rd Order**, and **4rd Order**. Options other than steady state apply to time-dependent (transient) calculations.

Temporal Algorithm

Temporal algorithms fall into one of two categories: steady state or time accurate. When **Steady State** is selected for the temporal accuracy, the user has the following options to select from.

Euler Implicit: This is the default steady state method for solving linear and non-linear partial differential equations (PDE) in *GASpex*. The Euler implicit method takes a non-linear PDE and turns it into a linear system that can be solved by one of the algorithms in the time integration (Sec. 15.3 on pg. 247). While this option is not considered time-accurate, using a constant time step produces a pseudo first-order solution.

Newton-Krylov Solver: The Newton-Krylov-Schwarz (NKS) option is a Krylov iterative method using a matrix-free approach. The solver is available through the PETSc software package.

When the temporal accuracy is not set to steady state, then a time-accurate simulation is assumed. *GASpex* offers two implicit time-accurate algorithms and one explicit. The algorithms are selected using the **Temporal Algorithm** option menu and has the following options to select from.

Implicit Dual Time Stepping: An implicit formulation which attains arbitrary time accuracy while allowing implicit treatment of the fluxes and source terms. This method incorporates a second temporal derivative (a pseudo-time derivative) for converging the root of a time-accurate discretization. Because time accuracy is maintained on the outer loop, popular implicit time-integration techniques which do not maintain temporal accuracy can be applied to the inner loop in pseudo-time. When the pseudo-time

step is set to infinity, the dual-time-stepping algorithm reduces to another popular implicit technique referred to as Newton sub-iteration. The temporal accuracy is either first, second, or third order. The pseudo-time step is set in the time integration tab (Sec. 15.3 on pg. 247).

Crank Nicolson Time Stepping: An implicit formulation that slightly differs from the dual-time stepping method. The temporal accuracy is strictly second order and in theory is less dissipative than second order dual-time stepping. Following in the steps of dual-time stepping, a pseudo-time step is used which is set using the time integration tab.

Explicit Runge Kutta: An m-stage Runge-Kutta method that offers first, second, and fourth order time-accurate solutions. As with most explicit methods, the time step has stability restrictions making the method practical for inviscid flows only.

Both the dual time stepping and Crank Nicolson options involve a two-step process. The first step is to add a time-dependent source term to the residual. This source term is what controls the time accuracy of the solution. The second step is to drive the residual to zero while holding the physical time constant. The second step is similar to converging a steady-state problem. Therefore, the dual-time-stepping and Crank Nicolson algorithms will still make use of the convergence information and time-integration tab.

The explicit Runge Kutta algorithm is based on the traditional Runge Kutta scheme. This algorithm requires the formation of the residual only (no Jacobians computed) and steps the solution forward in physical time.

Several differences should be pointed out between the time algorithms. Both the implicit and explicit algorithms can compute first-order and second-order solutions in time, but only the implicit schemes can compute third-order-accurate solutions. In a similar manner, only Runge Kutta can compute fourth-order-accurate solutions. Because of this, the GUI will not allow certain combinations. To select the Runge Kutta option, the temporal accuracy must be either first, second, or fourth order. For the dual-time scheme, the temporal accuracy must be either first, second, or third order. For Crank Nicolson, only the second order temporal accuracy is valid.

Since dual time stepping and Crank Nicolson solves an inner (pseudo time) problem, many of the input parameters used with the global, steady-state problems are used. For Runge Kutta, *GASpex* requires fewer input parameters from the user due to the simplicity of the explicit algorithm. Because of this, when the Runge-Kutta option is selected, there will be fewer input parameters selectable in the GUI. The non-selectable input parameters indicate that they are not used by the Runge-Kutta algorithm.

One final note on the temporal algorithms. Because the Runge-Kutta algorithm is explicit in nature, the maximum allowable time step will be smaller than that allowed for the implicit algorithms. But the trade off comes from the computational speed of the algorithm. The Runge-Kutta scheme will take less time to advance the solution one physical time step when compared to the other algorithms.

15.2.5 Time Dependent Information

When a run definition is solving a time-dependent problem, the **Time Dependent Info** settings are used to describe the time progression. The time-dependent information is only selectable when the **Temporal Accuracy** is set to something other than “Steady State”.

The physical time duration for the problem can be set using either the **Max # Of Steps** or **Max Time** inputs. *GAS*Pex will run until either the maximum number of time steps are done or until the maximum time is reached. Every time-accurate calculation begins with the time set to “0” seconds.

The physical time step (Δt) is set using the **Time Step** input. The problem will progress for the number of time steps or the maximum time, whichever occurs first.

When running a time-accurate problem, *GAS*Pex will save the solution and grid at $t = 0$ seconds and the time when the computation ends (at the max time or when the number of time steps is reached). In order to save the solution and grid at intermediate times, specify the time increment in which to save using the **Save Increment** input. When the physical time reaches the save increment, the solution and grid are saved to separate files in the **grids** and **solns** directories.

If the case involves overlapping Chimera grids, then the user must specify a **Chimera Increment**. Similar to the save increment, the Chimera increment will force the Chimera coefficients to be re-computed when the physical time progresses that amount. The Chimera coefficients need to be re-computed when one overlapping grid has moved enough to change which cells it needs to interpolate from and to. The default value is 1.0×10^{-6} sec, but the optimal value is problem dependent.

With each time step for the dual-time-stepping or Crank Nicolson algorithm, *GAS*Pex will compare the L_2 -norm of the residual to the relative and absolute convergence criteria provided in the **Convergence Info** section (see Sec. 15.2.2 on pg. 243). At the end of a time step, if the solution does not meet these convergence requirements, *GAS*Pex will stop execution. You can over-ride this behavior by selecting the **Always Continue To Next Time Step** check box. In this way you can force *GAS*Pex to run the specified time steps regardless of the convergence level. This option only applies to the iterative, implicit algorithms.

15.2.6 Solution Averaging

When the temporal accuracy is set to a value other than steady state (*i.e.*, a time dependant simulation), a section with the heading **Solution Averaging** is displayed. This allows the user to compute a running average of the solution. To activate this feature, select the **Compute Average Solution** option.

The averaged solution can be written to file at a user specified interval using the **Save Increment**. When the physical time reaches the save increment, the averaged solution is saved to a separate file in the **solns** directory. The file is flagged as an average by having the **.avg.sln** extension.

The average solution is computed using an exponential smoothing algorithm based on

the following formula:

$$\bar{q}^{n+1} = (1 - \alpha)\bar{q}^n + \alpha q^{n+1}$$

In the above equation, \bar{q} is the running average while q is the instantaneous solution. The user may specify the value of α using the **Alpha** input selection. The default value for α is 0.05.

15.2.7 File Information

The **Output Residual** option allows the user to select printing of the residual file. A residual value is computed for each finite volume cell and for each governing equation that is solved. The residual represents the flux balance on each finite volume cell, as well as any source contribution.

The residual for the entire run definition represents a global value for all the zones computed on. The residual is computed using the root-mean-square (RMS) value of the flux-balance residual over all the equations and cells. The residual value is one way to monitor solution convergence, as well as solution stability. A normalized value of the residual is also printed to the file. The file also contains additional information such as CPU time, parallel speed-up factor, load-balancing efficiency, percent CPU time for explicit and implicit communication, and the location of the largest residual.

The user selects the filename using the **Resid File** input. The default file name is **resid.run**. If the file does not exist, it will be created by the flow solver at the time of execution. If the file does exist, the flow solver will append to it. The file is written every iteration cycle during the run.

15.2.8 Marching Information

The last piece of information to set is applicable only to space marching. When space marching, the best initial guess in a zone may originate from an existing upstream solution. This initialization is accomplished by selecting **Pass Soln Into Zones At Start**. Additionally, you may wish to continue to the next plane even if a convergence tolerance has not been met on the current plane within the specified number of cycles. This may occur because of a limit cycle, for example. To use this option select the check box labeled **Always Continue To Next Plane**. In most circumstances you will want to stop and determine the problem if a particular plane causes convergence difficulty. The marching information is only visible when space marching is selected in the Navier-Stokes physical model.

15.3 Time Integration Run Definition Tab

The **Time Integration** tab of the **Run Definitions** frame describes the algorithms and parameters applicable to advancing a solution in time. *GASPerx* solves a time-dependent, non-linear coupled system of equations when the flow solver is executed. Except for Runge

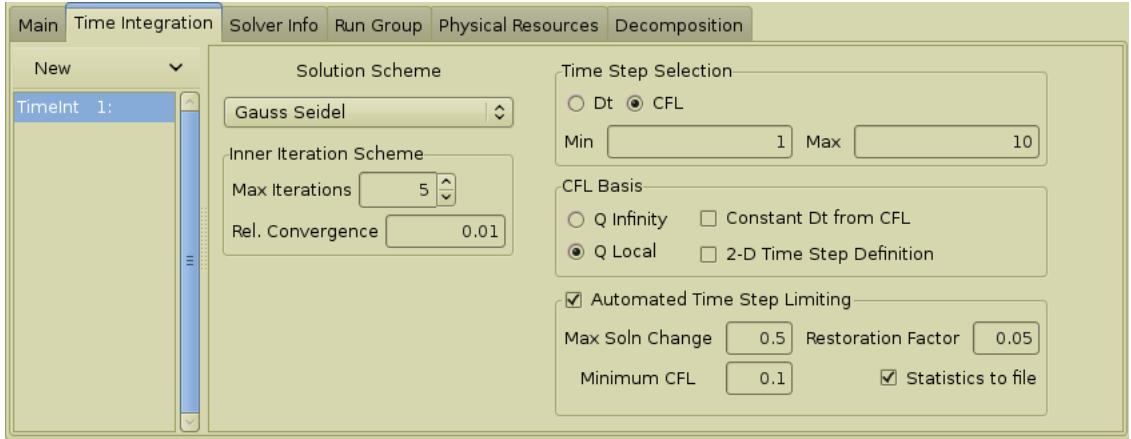


Figure 15.3: The **Time Integration** tab for a run definition.

Kutta (and the simple Euler-explicit scheme), *GASPex* solves this system in an implicit fashion by linearizing the equations. How *GASPex* then solves the linear system of equations is determined in the **Time Integration** tab shown in Fig. 15.3 on pg. 248.

15.3.1 Solver Algorithm

The options for the time integration are listed in the **Solution Scheme** option menu. In most cases where a global two or three dimensional problem is being solved, the Gauss-Seidel or PETSc GMRES options are recommended. Additional options are available for special cases such as space marching. A brief description of these algorithms are given below.

1-D LU:

This algorithm performs a complete LU decomposition on a plane, and is recommended for two-dimensional, space-marching problems. This option should not be used for 3-D marching problems or global problems. This option is only available for physical models which use a marching **Global/Marching Strategy** in the **Inviscid** tab.

2F-AF:

The name **2F-AF** stands for two-factor approximate factorization. In this algorithm, the linear system of equations are approximately factored in a spatially split fashion. The solution update is obtained by solving two block tridiagonal systems. The algorithm is designed to operate on a plane, which means **2-F AF** cannot be used for non-space marching run definitions. The primary use of this option is for three-dimensional, space-marching problems.

Jacobi:

Jacobi is an iterative technique for solving a linear system. This option is intended for use with an inner-iteration scheme and can be used with any of the run types.

Gauss Seidel:

Similar to **Jacobi**, **Gauss Seidel** is another iterative technique for solving a linear system. The algorithm implemented in *GASPerx* is a symmetric Gauss-Seidel solver which does a forward and backward sweep each iteration. This option is intended for use with an inner iteration scheme and can be used with any of the run types. The Gauss-Seidel algorithm with inner iterations is recommended for two or three-dimensional global problems.

Euler Explicit:

This option runs the simple Euler explicit time-integration scheme and is not intended for use with an inner-iteration scheme. At present this option is limited only to steady-state simulations. Note that when usng a explicit solver, the time step restrictions are more severe and a smaller time step is normally required compared to implicit solvers.

PETSc-GMRES:

The generalized minimal residual method (GMRES) is an iterative Krylov subspace method that is available through the PETSc software package. While the number of search directions are fixed internally, the user may control convergence through the inner-iteration parameters. The GMRES solver is robust and ideal for most applications.

PETSc-NKS:

The Newton-Krylov-Schwarz (NKS) option is a Krylov iterative method using a matrix-free approach. The solver is available through the PETSc software package. This solver works best with an infinite CFL/Dt and only after the solution is close to convergence (after 3 or more orders). Therefore, the NKS option should be combined with one of the more robust solvers like Gauss-Seidel or GMRES. For Navier-Stokes applications, NKS does not work well with min-mod or other non-smooth limiters that oscillate.

The Portable, Extensible Toolkit for Scientific Computation (PETSc) [55] is developed and maintained by Argonne National Laboratory. It was first made available to *GASPerx* in Version 5.2 and consists of a suite of routines used to solve partial differential equations for scientific applications. One strength of the software is its performance for large-scale applications through use of the Message Passing Interface (MPI).

15.3.2 Inner Iteration Scheme

The Jacobi and Gauss-Seidel solvers alone are not very effective time-integration strategies. However, these methods take far less CPU time per iteration than other implicit schemes, and when combined with an inner-iteration strategy, convergence rate drastically improves. The PETSc GMRES and NKS solvers are also schemes that need to use an inner-iteration strategy to improve the convergence rate.

There are two parameters for doing inner iterations. The first controls the maximum number of inner iterations and is set in the **Max Iterations** type in box. This is the number of inner cycles to be performed unless a tolerance is satisfied. The tolerance is relative to the first inner cycle and is specified in the **Rel. Convergence** type in box.

Typical values for the inner cycle number range from 3-15. The default choice when using Jacobi or Gauss-Seidel for the inner cycle number is “5” with a convergence tolerance of “0.01”. Two orders of convergence on the linear problem is usually more than adequate. The PETSc solvers are capable of converging the solution better and can support a lower tolerance. Therefore the default is increased to 10 for the inner cycle number and 1×10^{-6} for the tolerance.

15.3.3 Time Step Selection

A constant time step, used for every cell in the run group, can be applied by selecting the **Dt** radio button. A more efficient method of advancing the solution in time is to use a uniform Courant number for each cell (*i.e.*, a non-dimensional time step). In this case, each cell has a unique time step based upon the CFL number, a reference velocity, and the cell size. This option corresponds to selecting the **CFL** radio button.

For most problems, the CFL should be specified first. If no CFL can be found to converge the problem, a fixed time step may be required, though this is rare. Some very stiff problems may require the use of a fixed time step.

The time step or CFL can be held constant during the solver or allowed to change. This is specified using the option menu next to the **Dt** and **CFL** radio buttons. The options for the CFL and Dt are as follows:

Constant Value: The Dt or CFL value is held constant during the solver.

Linear Ramp: The CFL or time step will increase over the course of a run in which the value varies linearly with the cycle number between the **Min** and **Max** inputs.

Log Ramp: The CFL or time step increases over the course of a run in which the value of the time step varies in a logarithmic manner with the cycle number between the **Min** and **Max** inputs. This is beneficial if the min and max values are orders of magnitude apart.

Residual Ramp: The CFL or time step increases as the normalized residual decreases. When a simulation first starts, the normalized residual begins at unity, which corresponds to the **Min** value. As the residual decreases, the CFL or time step will increase toward the **Max** value. The max value is reached when the normalized residual reaches the **Relative Convergence**, which is set in the main section of the run definition. The increase is done in a logarithmic fashion since residual values change over orders of magnitude.

When the CFL or time step uses one of the ramping options described above, the **Min** type in box is used to specify the initial time step or CFL number for the solver. Likewise, the **Max** parameter is used to specify the final time step or CFL number for the solver. When the **Dt** radio button is selected, the input values must be given in seconds. When using the **CFL** option, the input value is a non-dimensional number.

The optimal CFL number is both algorithm and problem dependent. Typical allowable values for simple, perfect gas, inviscid flow with implicit time integration range from 0.1 to 100. As the complexity of the flow conditions, geometry, and physical model increase, so does the stiffness of the problem, and the maximum allowable CFL may be reduced.

Note: While the approximate factorization scheme (available for space marching) may be stable at CFL numbers greater than 100, optimal convergence will be achieved with a CFL number anywhere between 1 and 20.

15.3.4 CFL Basis

The **CFL Basis** parameters are applicable only when using a CFL number. With a CFL, the time step for a gas dynamic system is calculated using

$$\Delta t = \frac{\lambda \Delta_L}{c}$$

where λ is the CFL or Courant number, Δ_L is a characteristic length of the cell, and c is the fastest characteristic speed. Using the **Q Infinity** option corresponds to calculating the characteristic speed with free-stream quantities, while **Q Local** calculates the characteristic speed using the local cell quantities.

Using the free-stream quantities has been effective for calculating hypersonic blunt-body flows where the fastest characteristic speed changes orders of magnitude between adjacent cells. For most applications, the local-time-step option is preferable. Our experience has shown that high-speed, viscous flows converge well by first using local time stepping and then switching to free-stream time stepping. Often the local time stepping will converge a viscous region and the primary shock features, but the pre and post-shock time steps are orders of magnitude different causing the residual to plateau. Switching to the free-stream characteristic speed in the CFL definition equalizes the time steps around the shock and aides convergence.

An additional option is available which computes a constant time step using the CFL number. This is the **Constant Dt Based On CFL** option. Here the time step is computed using local cell quantities, and the smallest time step for the zone is used to set all the other cells. In this way, all the cells have the same time step which is allowed to vary as the solution converges.

The **2-D Time Step Definition** option is intended for use with axi-symmetric problems and is applicable only when using a CFL number. This option removes the influence of the symmetry direction from the time step calculation, providing larger time steps and faster flow evolution near the singular line.

15.3.5 Automated Time Step Limiting

The **Automated Time Step Limiting** option is intended to help improve robustness. When using either a CFL or constant Dt, this option will adjust the time step for an individual cell. The time step will be lowered if the solution changes by more than a prescribed amount (specified by **Max Soln Change**). For example, a **Max Soln Change** value of 0.5 means that the time step will be lowered if the solution changes by more than 50%. The time step limiting is applied on a cell by cell basis.

Practical values for limiting range from 5% to 50%. The default value of 0.5 (50%) yields mild limiting. To increase the limiting to help with numerical stability, lower the limit value to as low as 0.05 (5%). Values lower than this may still be beneficial, but convergence will be slowed.

If the time step has been lowered for a cell, yet the solution is no longer changing by more than the prescribed limit, the time step will be raised according to the value of **Restoration Factor**. The default for this value is 0.05, which corresponds to a 5% increase in time step each cycle.

The minimum time step for a cell will be no less than **Minimum CFL** (or **Minimum DT** if **DT** is selected). The **Minimum CFL** prevents the time step from being too small and impractical.

When using automated time step limiting, the first cycle in the flow solver will not update the solution. This allows the algorithm to determine if limiting is needed upon solver start up. If so, the time step is limited and the solution is then allowed to update on all subsequent cycles.

Note that because the limit factor is dependent upon the time integration history, and because the limit factors are not saved to the restart file, convergence characteristics can change upon restart.

When the **Statistics to file** box is selected, the number of cells limited, as well as the minimum time step for any cell in the domain will be printed to the file “dtlimit.dat” after each cycle.

15.4 Solver Info Run Definition Tab

The **Solver Info** tab of the **Run Definitions** frame describes solver specific information for the equations being solved. The default solver info is for the Navier-Stokes equations. The complete list of solvers available in GASPE are listed below. Note that each solver corresponds to one of the physical models in *GASPE*.

Navier-Stokes: Solves the Navier-Stokes equations including turbulent flow.

Solid Thermodynamics: Solves the energy equation for heat transfer inside a solid.

Lagrange Spray: Performs particle tracking of either solid or liquid particles released inside a Navier-Stokes flow field.

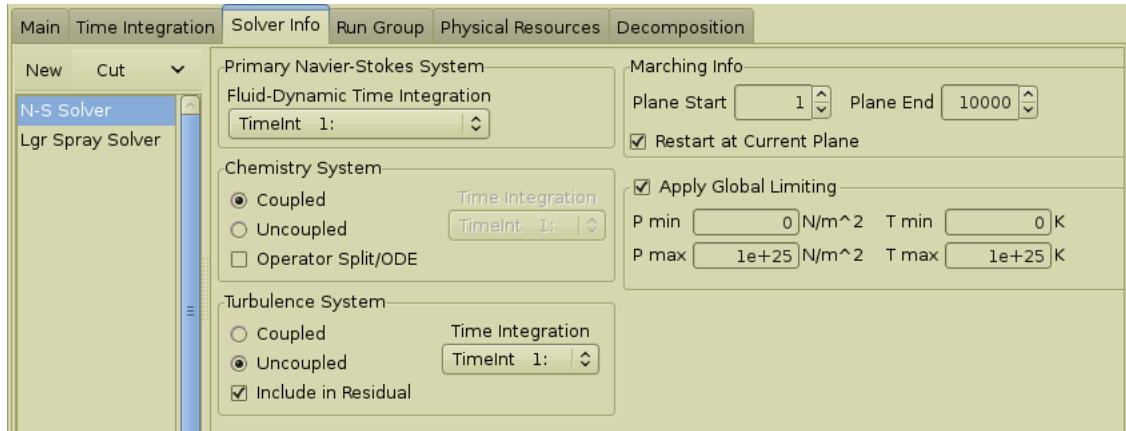


Figure 15.4: The **Solver Info** tab for Navier-Stokes.

Mechanical Stress: Solves for the displacement inside a solid material due to body forces and thermal stress.

Each available **Solver Info** will now be discussed in more detail.

15.4.1 Solver Info Select List

Each run definition can call one or more solvers (set in the Run Group). At least one solver info is required, but additional ones can be created or deleted using the solver info select list. The select list is inside the **Solver Info** tab and contains the following buttons: **New**, **Cut**, **Copy**, and **Paste**. Below these buttons is a list which contains all the solver info associated with the run definition. The solver information can be viewed in the GUI by selecting the desired entry in the list. The solver info name can be changed by double clicking on the list item and entering new text.

To create a new solver info, use the **New** option. This will show an option menu with the available solvers. Selecting on one of the solvers will add that solver info to the list. To copy an existing solver info, select it from the list, press **Copy** and then press **Paste**.

Both the **Copy** and **Cut** buttons create a duplicate of the data in the clip-board. The **Cut** button will also remove the entry from the select list. The last entry in the clip-board can be recovered using the **Paste** button.

15.4.2 Navier-Stokes Solver Info

For the “Navier-Stokes Solver”, the fluid-dynamic equations are solved in a single, fully-coupled system. However, *GASPex* also allows for the species continuity equations (Chemistry System) and the turbulence equations (Turbulence System) to be uncoupled from the primary fluid-dynamic equations and solved as separate systems. In addition to describing the uncoupling, the solver info section also specifies the time-integration scheme to be used

for each system. An example of the **Solver Info** tab for a Navier-Stokes problem is shown in Fig. 15.4 on pg. 253.

The **Solver Info** data is divided into three sections that describe the **Primary Navier-Stokes System**, the **Chemistry System**, and the **Turbulence System**. The **Primary Navier-Stokes System** frame specifies the time-integration strategy for either the fully-coupled system or the primary fluid-dynamic system (if any other system is uncoupled). The primary fluid-dynamic system consists of the mass, momentum, and energy conservation equations. Uncoupled chemistry is specified by selecting the **Uncoupled** radio button under the **Chemistry System** frame. Once selected as uncoupled, additional options then become available to describe the time integration strategy for the chemistry system. Similarly, uncoupled turbulence is specified by selecting the **Uncoupled** radio button under the **Turbulence System** frame. As before, additional options then become available to describe the time integration strategy for the turbulence system. If both the chemistry and turbulence radio buttons are selected as **Coupled** then only the single **Fluid-Dynamic Time Integration** must be specified.

The time-integration scheme to be used for the Fluid-Dynamic, Chemistry and Turbulence systems is set using the corresponding **Time Integration** option menu in each section. Parameters in the time integration definition are discussed in detail in Sec. 15.3 on pg. 247.

Chemistry System

GASpex has the ability to solve the species continuity equations together with the fluid dynamic equations as a fully-coupled system, or uncoupled as a separate system. Uncoupling a weakly-coupled system of equations offers the potential to lower computational time and memory necessary for the solution of the block-matrix system. When uncoupled chemistry is selected, the fluid-dynamic system update is followed by a non-linear residual update for the chemical system. To uncouple the specie continuity equations from the primary fluid dynamic equations, select the **Uncoupled** radio button in the **Chemistry System** section. In addition, a new time-operator splitting method has been added to help speed up flow simulations with large numbers of chemical species (greater than 15). To utilize this scheme select the **Operator Split/ODE** check box in the chemistry section. This method uses two fractional steps to update the specie densities - a transport fractional step and a reaction fractional step. The reaction fractional step uses a variable coefficient ordinary differential equation (ODE) solver to appropriately handle chemical stiffness. The transport fractional step can be solved using a low-cost Euler-Explicit scheme. To correctly utilize this option, the chemistry time integration definition should include **Euler Explicit** as the solution scheme.

Turbulence System

GASpex has the ability to solve any of the multi-equation turbulence models together with the fluid dynamic equations as a fully-coupled system, or uncoupled as a separate system. Uncoupling the turbulence equations offers the potential to lower computational time and memory necessary for the solution of the block-matrix system. When uncoupled turbulence is selected, the fluid-dynamic system update is followed by a non-linear residual update

for the turbulence system. To uncouple the turbulence equations from the primary fluid dynamic equations, select the **Uncoupled** radio button in the **Turbulence System** section. This option is recommended for most all turbulence model simulations. We also recommend using the same implicit time-integration strategy as used for the fluid-dynamic system. This is the strategy shown in Fig. 15.4 on pg. 253.

The residual calculation by default includes all the equations being solved for. If the user would like to exclude the turbulence equations in the residual value reported in the main residual file (`resid.run`), then the **Include in Residual** option should be un-selected. This only impacts the residual value as reported in the output file. The solver is not impacted by this option.

Marching Info

If space marching is being used, then the **Marching Info** settings are utilized. Space marching is specified inside the physical model under the **Inviscid** section (see Sec. 11.3.1 on pg. 173). Space marching can only be applied to a structured grid because it requires a computational-coordinate direction to march in. The direction is specified in the physical model.

When space marching, *GASpex* must know on which plane to begin and end the marching. This information is given under the **Plane Start** and **Plane End** headers. The default value for the plane start is the first plane (plane value of 1) while the default value for the ending plane is 10000 (*i.e.*, a large grid dimension). The **Plane End** value can be greater than the number of planes. Note that the plane start and plane end values must correspond to the run definition sequence level dimensions and not to the fine mesh dimensions.

The last setting is **Restart at Current Plane**. *GASpex* keeps track of which plane the solver is on when execution is stopped. If you want *GASpex* to ignore the **Plane Start** setting and resume marching on the last plane solved on, select this option. The current plane is stored in the `.aux.gsp` file.

Global Limiting

There are several types of limiting that can be used during a run (see Sec. 3.11.1 on pg. 46 for a brief summary). The global limiting option in the solver info section acts as a catastrophic limiter. It places minimum and maximum limits on the pressure and temperature values. These limits are enforced at the end of each cycle just after the solution is updated.

When limiting the pressure, the limiter is applied directly to the solution (since pressure is always one of the solution variables). Limiting the temperature is not as straight forward since it is not usually a solution variable. Instead, temperature is computed using the equation of state which involves the pressure and mixture density. Therefore, when temperature is not a solution variable (*i.e.*, a primitive variable), the mixture density is adjusted as a way of enforcing the minimum and maximum temperature settings.

To use global limiting, the user must select the **Apply Global Limiting** option. This will enable the limit inputs to be selectable.

15.4.3 Solid Thermodynamics Solver Info

When solving the heat transfer within a solid, the user will need to create a solid thermodynamics physical model. The run definition will then require a solver info for solid heat transfer labeled “Solid Thermodynamics Solver”.

For this solver, the only information needed is the time integration selection. Parameters in the time integration definition are discussed in detail in Sec. 15.3 on pg. 247. The time integration allows the user to specify either a CFL or constant time step. If a CFL value is used, the time step is computed using

$$\Delta t = \frac{\lambda(\Delta L)^2}{\alpha}$$

where λ is the CFL or Courant number, ΔL is a characteristic length of the cell, and α is the thermal diffusivity ($k/\rho C_p$).

15.4.4 Lagrange Spray Solver Info

The inputs for the particle tracking solver are available in the “Lagrange Spray Solver” Info. Details of the Lagrangian solver can be found in the *GASpex* technical manual.

The **Update Time** input is used only for steady state problems. For time-accurate problems, the actual physical time step is used. The update time is the amount of time particles are allowed to travel during an update frequency. Each time particles are released and tracked, both the new particles and the existing ones in the flow will be tracked for a set amount of time. Again, the particle update time is specified by the user for steady state flows, and the physical time step is used for time dependent flows. A large value is normally desirable for steady state flows since this allows the particle being injected to be tracked through the entire flow-field and then exit (assuming it leaves the computational domain).

The Lagrange solver uses a Runge-Kutta type time integration scheme. Particles or rays are traced through the domain until either they leave the domain or until the specified integration time (Update Time) is reached. Particles or rays still in the domain at the end of the integration time will remain at that location until the Lagrange solver is called again. The Runge-Kutta time integration uses a variable time step. The time step is adjusted locally in order to maintain stability. Because of this local adjustment, bounds are necessary to prevent too small or too big of a time step from being selected. These bounds on the time step are specified using the inputs **Minimum Time Step** and **Maximum Time Step**.

15.4.5 Mechanical Stress Solver Info

When using a mechanical stress physical model, a “Mechanical Stress Solver” Info is used to specify the time integration selection. This is the only piece of information that needs to be set for the solver information.

Recall that the time integration has the option of specifying either a physical time step or a CFL number (non-dimensional time step). The CFL is based on the local cell volume

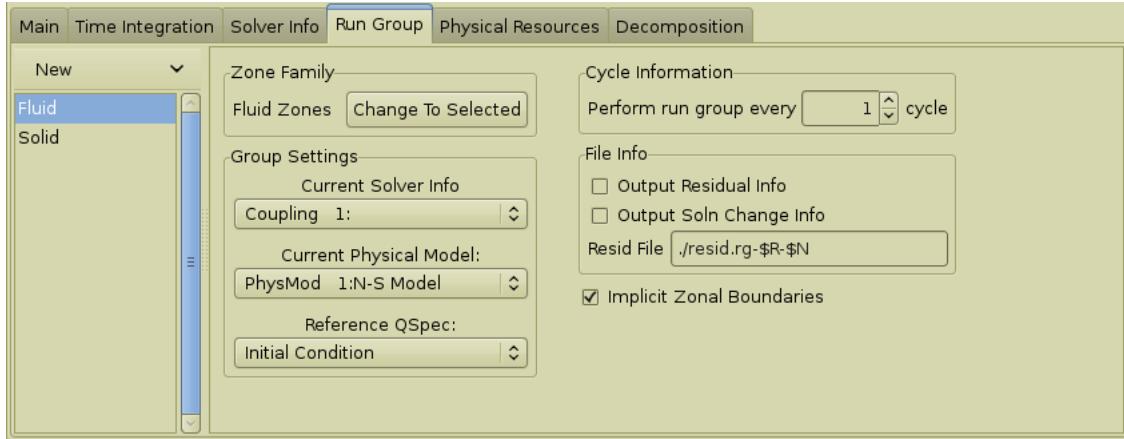


Figure 15.5: The **Run Group** tab for a run definition.

and the displacement velocity. At this time *GASPex* is not solving for the displacement velocity, but only the final steady state solution. Therefore the CFL is not applicable and the user should only use a constant time step. If the CFL option is selected, then an infinite time step is used. The stress solver is very stable and in most cases an infinite time step will not hinder convergence to a steady state.

15.5 Run Group Run Definition Tab

A run group specifies a group of zones (called a zone family), a solver info, a physical model, and some additional information. With this information, the run group tells *GASPex* the group of zones to solve on, the physical model to use and what time integration to apply (solver info).

At a minimum, each zone of interest should have at least one defined run group; however, different integration techniques can be employed for the same zone during the same cycle loop. An example of the **Run Group** tab is shown in Fig. 15.5 on pg. 257.

15.5.1 Run Group Zone Family

Each run group will operate on a zone or collection of zones which is referred to as the **Zone Family**. The zone family is set using the **Change To Selected** button. In order to set the zone family, first select a zone folder from the tree view (select the folder name). Then press the **Change To Selected** button. The name of the zone folder should appear in the **Zone Family** box.

If you wish to run all the zones, then select the **Zones** text from the tree view folder and press the **Change To Selected** button. If you have multiple zones and only wish to run a few of the zones, then you will need to create a sub-folder inside of **Zones** and place the desired zones inside the new folder. This new folder can then be set as the zone family.

15.5.2 Run Group Solver Info

The coupling strategy to be used in the run group definition is set using the **Current Solver Info** option menu. The solver info options are discussed in Sec. 15.4 on pg. 252, and describe various coupling and time integration strategies for solving the governing equations.

15.5.3 Run Group Physical Model

The physical model to be used in the run group definition is set using the **Current Physical Model** option menu. The option menu will list all the physical models that have been created. But only those physical models that are consistent with the zone family will be selectable.

Each physical model is checked against the zones in the zone family for consistency. A physical model is consistent with a zone when the zone's solution and the physical model reflect the same solution variables. For example, in a Navier-Stokes solution, the species densities, non-equilibrium vibrational energies, and turbulence variables must all be present in the zone family solution if used by the physical model. Recall that each zone's solution is based on the initial physical model setting in the **Initialization** tab (inside of the **Zones** frame). A solution based on a physical model may also be added to a zone inside the **Zones/Soln Mgt** section.

15.5.4 Run Group Reference Q Spec

With a physical model defined, a run group also needs to have a Q Spec selected. The Q Spec is set using the **Reference QSpec** option menu. The Q Spec is used as a reference to the local time step in the Navier-Stokes solver. For example, if a CFL number is specified in the time integration tab, a Q Spec is required to convert the **Q Infinity** CFL number to a time step. For most other solver types the reference QSpec is not used.

15.5.5 Run Group Cycle Information

Under normal circumstances, the run group will be applied and executed each cycle. In certain cases when multiple run groups exist, the user may want to skip a run group. This frequency is set using the **Perform run group every** input.

An example of when this might be desirable is when the Lagrange Spray solver is used. When the Lagrange solver is not performed for a given cycle, any source terms that are used in the Navier-Stokes solver are frozen until the Lagrange solver is run again.

For the Lagrange solver, the use of this input will differ slightly depending on whether the time algorithm is steady state or time-accurate. For steady state flows, the update frequency corresponds to a new set of particles being released according to the particle source information in the physical model. Both new and existing particles are then tracked for a set amount of time. Users may not want to update the particles each cycle since particle tracking is computationally expensive (for larger update times). Therefore, it is common to perform this operation after a specified number of cycles. The default value for the update

frequency is one, but users will normally want to increase this value by one or two orders of magnitude depending on the total number of cycles expected to run.

For time-accurate flows, particles are released and tracked at the beginning of each time step by default. For the **Implicit Dual Time** algorithm, the user specifies a set number of cycles to iterate on before moving on to the next time level. If the particle update frequency value is set smaller than the cycle number for each time step, the particle locations will be updated to reflect the current solution values. This may be desirable to do if the solution is changing significantly between time steps (or if a large number of cycles are being performed between time steps). If no updates are to be performed during a time step, the user should set the update frequency value to be larger than the **Maximum # of Cycles** value.

15.5.6 Run Group File Info

There are a number of solvers inside of *GASPex*, each of which solve a different set of governing equations. For example, there is a Navier-Stokes solver for the conservation of mass, momentum, and energy. There is also a solid thermodynamics solver, a stress solver for solid materials and a solver for tracking particles for example. A run definition may have multiple run groups in which different sets of equations are solved by each run group.

In the **Run Definitions/Main** section, there is an option for printing residual information based on the entire run definition. For the run group, there is a similar option for printing residual information just for the solver used by the run group. The run group residual output is turned on by selecting either the **Output Residual Info** or **Output Soln Change Info** options. The user may then specify a file name using the **Resid File** string input. The default file name is `resid.grp-$R-$N` where `$R` is the run definition number and `$N` is the run group number.

The **Output Residual Info** will force the residual value (which is computed using an L2 norm procedure) to be printed to file while the **Output Soln Change Info** will print out the L2 norm of the solution update value. The solution update is the actual change in the primitive variables for a cycle. If both options are selected, then both values will be printed to file.

Even if there is only one run group, users may want to print out the run group residual. The format will be different from the run definition residual file in that it will print out the residual value (or solution change) for each equation solved or by equation group. For example, if the Navier-Stokes solver is used and the **Group Data By Equation Type** is selected, the run group residual will print out the individual residuals for continuity, momentum, energy, non-equilibrium vibrational energy, and turbulence. In this way the user may see the break-down of the individual equations with regards to the total residual value. If the data is not grouped by equation type, then the residual or solution change data is printed for each equation solved.

15.5.7 Run Group Implicit Zonal Boundaries Setting

If the zone family defined for the run group has more than one zone, the **Implicit Zonal Boundaries** check box controls how data is passed between zones. If the check box is selected, the zone family is treated implicitly, which means that the zones in the zone family are solved as one large zone each cycle and solution update information is shared between zones within the iterative solver. Otherwise, the solution for each zone in the zone family is updated independently each cycle, and solution information is only passed among zones between cycles.

The use of this option affects two things. It will impact the convergence rate and the required memory to run the problem. With the **Implicit Zonal Boundaries** option on, convergence should be better than if the check box is turned off, but more memory will be required to run the problem. So if memory is not an issue, it is recommended that this setting be turned on when the run group has multiple zones.

15.5.8 Run Group Select List

Each run definition can have one or more run groups. At least one run group is required, but additional run groups can be created or deleted using the run group select list. The select list for the run group is inside the **Run Group** tab and contains the following buttons: **New**, **Cut**, **Copy**, and **Paste**. Below these buttons is a list which contains all the run groups associated with the run definition. The different run groups can be viewed in the GUI by selecting the desired run group description in the list. The run group description can be changed by double clicking on the list item and entering new text.

A new run group can be created using either the **New** or **Copy/Paste** buttons. A run group with default values can be created by pressing **New**. To copy an existing run group, select the run group from the list, press **Copy** and then press **Paste**.

Both the **Copy** and **Cut** buttons create a duplicate of the run group in the clip-board. The **Cut** button will also remove the entry from the select list. The last entry in the clip-board can be recovered using the **Paste** button.

15.6 Algorithms Run Definition Tab

The **Algorithms** tab will only visible if the user has created a zone algorithm (see Sec. 8.8 on pg. 106 for more information). When this is the case, the user may use this tab to specify which zone algorithms to apply at the end of the current run.

To add a zone algorithm to the run, click on the **Add Algorithm** button. Doing this will add an option menu to the table that the user can then select from.

There can be multiple zone algorithms applied at the end of a run. If the user needs to remove an algorithm from the list, then simply select the algorithm to remove and press the **Delete Algorithms** option.

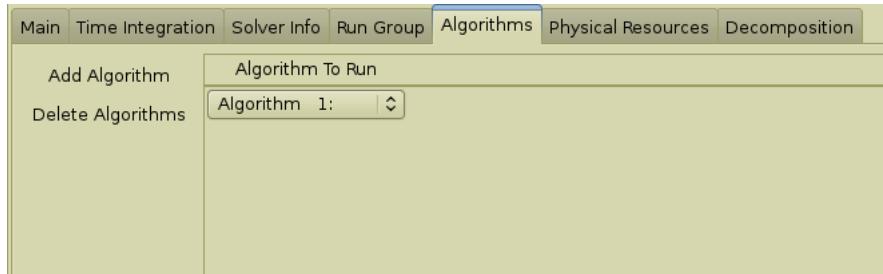


Figure 15.6: The **Algorithms** tab showing an active algorithm.

Main	Time Integration	Solver Info	Run Group	Motion Models	Physical Resources	Decomposition	
New	Host Name	Host #	Execution String	Scale Fac	Mem/CPU (MB)	nCPU	Cut
Cut	localhost	0	\$(AEROSOFT_HOME)/bin/gasp	1	1000	16	<input type="checkbox"/>

Figure 15.7: The **Physical Resources** tab for a problem that will be run on 6 processors, each using no more than 150 MB of memory.

15.7 Physical Resources Run Definition Tab

The **Physical Resources** tab of the **Run Definitions** frame is used to specify the physical resources (machines, processors, etc) that will be employed to solve the problem. It is here that the user will specify how many different resources will be used to run the problem, and the properties, such as number of processors (CPUs), memory and speed, for each of the resources. An example of a typical **Physical Resources** tab for running on a 6-processor cluster with each node having 150 MB of memory is given in Fig. 15.7 on pg. 261.

NOTE: the following sections will discuss the parameters in the **Physical Resources** tab, and their relationship to running a parallel problem, but they will not go into the details of how to run *GASPerx* on the various parallel platforms. That discussion is in the *GASPerx* installation section.

15.7.1 Resource Table

The **Physical Resources** tab consists primarily of the resource table. A single “resource” can describe a single processor machine, a multi-processor machine, or a cluster of homogeneous machines.

New Button

The **New** button is used to add new resources to the resource table. New resources are always added to the end of the table.

Cut Button

The **Cut** button is used to remove resources from the resource table. All of the resources which have the **Cut** toggle button located at the far right of the table set to true will be removed when the **Cut** button is pressed.

15.7.2 Host Name and Host Number

For most cases, the **Host Name** and **Host Number** can simply be left as the default values of **localhost** and 0. The only case where it needs to be otherwise is when using the MPICH libraries instead of the default MPI libraries. When using MPICH, there are two possible entries for the **Host Name**, either “machine name” or “machine name%*d*”. The first is self-explanatory, the second option is used to simplify the input for a cluster of homogeneous machines which all have the same machine name followed by sequential node numbers. So to run a problem on a 3-processor cluster of machines named node1, node2, node3, the **Host Name** could be set to “node%*d*”. The **Host Number** is only used when the “%*d*” **Host Name** is employed, and it represents the first number in the sequential list, or in this case 1 (for node1). To run on non-sequential nodes (e.g. nodes1-3, and nodes6-7), the user would have to use 2 resources, the first one as described above, followed by a second resource that had **Host Number** set to 6.

When using the default MPI libraries, the machine names are taken from a host file, and the only relevant **Host Name** is the first one, and it should be set to **localhost**.

15.7.3 Execution String

The **Execution String** is the name of the *GASPEX* executable for the given resource. It must be given in the form of an absolute path. The default value is “\$(AEROSOFT_HOME)/bin/gaspex” where AEROSOFT_HOME is an environment variable describing the path to the AeroSoft install directory. Currently, AEROSOFT_HOME is the only environment variable that can be used in the executable string.

Unlike the **Host Name**, the **Execution String** is important when using MPI. The ordering of the **Execution String** names in the resource table must correspond to the ordering of the nodes in the host file.

15.7.4 Scale Factor

The **Scale Fac** is used to improve load balancing on heterogeneous clusters. It defines the relative time to compute an iteration on the machine(s) in the given resource to a baseline

of 1. So if a user wanted to run a problem on two nodes where node2 was twice as fast as node1 (i.e. it would take 1/2 as long to compute an iteration on node2), then the **Scale Fac** for node1 would be 1, and for node2 it would be 0.5.

When using multiple resources, the ordering of the **Scale Factor** in the resource table should correspond to the ordering of the nodes in the host file.

15.7.5 Processor Memory

The available memory for each processor is determined by the **Mem/CPU (MB)** value. This is given in Mega Bytes and is used in the auto decomposition feature (discussed in Sec. 15.8.1 on pg. 264). If you are using processors with distributed memory, then you would give the available memory for each processor. For a shared-memory platform, divide the total memory available by the number of processors.

When using multiple resources, the ordering of the **Mem/CPU (MB)** in the resource table should correspond to the ordering of the nodes in the host file.

15.7.6 Number of Processors

The number of processors for each of the resources listed in the table are given in the **nCPU** type in. For a single shared-memory machine with multiple processors or a homogeneous cluster of nodes that take advantage of the “%d” naming convention (see Sec. 15.7.2 on pg. 262), the user can typically define all of the processors with a single resource. In such a case, **nCPU** would be set to the total number of processors to be run.

The other extreme case is where each machine requires a separate resource definition. In that case, all of the values for **nCPU** would be set to 1, and the total number of processors in the run would be the same as the number of resources. The final possibility is a mixture of the two extremes in which there are multiple resources defined with varying values of **nCPU**. In that case, the total number of processors would be the sum of all the **nCPU** values.

When using multiple resources, the ordering of the processors in the resource table should correspond to the ordering of the nodes in the host file. So if the first resource has a value of 3 for **nCPU**, then the first 3 nodes in the host file should be those 3 nodes in order. The next set of nodes in the file would need to correspond to the **nCPU** processors in the second resource, and so on.

15.8 Decomposition Run Definition Tab

The **Decomposition** tab, Fig. 15.8 on pg. 266, allows the user to perform zone decomposition, which is the process of dividing a zone up into multiple partitions. Prior to decomposition, a zone consists of a single partition. Zone decomposition is done for three reasons. The first reason is to run the problem in parallel efficiently. There must be, at a minimum, the same number of zone partitions as processors. Each processor is assigned at least one zone partition to compute. To maximize the work load on all the processors, the load size

assigned to each processor should be roughly equal. This can be accomplished using zone decomposition. The second reason to break zones up into smaller sizes is to decrease the memory use. Running more zones with explicit passing of partition boundary information will reduce the required CPU memory. The final reason is also to reduce memory, but in this case the reduction of local memory is desired for the purpose of improving CPU performance on cache-based computers and not for the purpose of memory limits.

15.8.1 Decomposition Type

There are now two algorithms for performing an automated decomposition in *GASPEX*. The first and original algorithm is chosen by selecting **Load Balance** for the **Decomp Type**. This algorithm attempts to balance the work as evenly as possible across the total number of processors chosen. This algorithm is the default and it works well for most cases. For platforms with large number of processors and in cases where the work load per zone might end up being too small compared to the communication costs, there is an alternate method which is chosen by selecting **Zones Per Proc** for the **Decomp Type**. Fig. 15.9 on pg. 266 shows the GUI layout when this method is selected. Note that the inputs have changed from the previous layout shown in Fig. 15.8 on pg. 266.

Load Balancing Automated Decomposition

When the **Load Balance** algorithm is selected, the **Auto Dec** button is designed to set-up the decomposition to give proper load balancing for parallel runs. **Auto Dec** will set the partitions (in I, J, and K for structured grids) necessary to meet the target efficiency specified by the **Target** type in. The value for the **Target** efficiency ranges from 0 to 1, where 1 represents a perfectly balanced case (or 100% efficiency). The default value is 0.90 (or 90% efficiency) which means that the best attainable parallel speed-up will be 90% of the theoretical maximum (i.e. the number of processors). So for 4 processors with a 90% efficiency, the best possible parallel speed-up would be 3.6. There are many factors involved in determining parallel speed-up and the load balancing doesn't consider every possible factor, so these numbers are meant to be a guideline, not a hard fast rule.

Most of today's parallel architectures consist of cache-based machines, and as such, the local memory sizes can have a relatively large impact on performance. The parameters **Min** and **Max** refer to the number of points in a partition and give the user more control over the final partition sizes that will result from the decomposition algorithm. This provides the user with the ability to utilize zone size as a tuning parameter to get optimum performance on a given platform. The **Min** and **Max** values represent the range of points in the largest allowable zone partition. So setting a range of 4000 and 10000 would mean that the decomposition algorithm would return a result which attempts to meet the target efficiency in which the largest zone partition has between 4000 and 10000 points in it. If the algorithm is unable to meet the target efficiency within the specified range, then the highest efficiency achieved in that range is returned instead.

The default value for **Max** is the number of total grid points for all zones in the run divided by the number of processors. This would be the number of grid points on each processor if all the zones were the same size and there were one zone per processor. The default value for **Min** is **Max** divided by 10. This essentially means the user is trying to put no more than 10 zone partitions of equal size on each processor on average. There is typically no reason to raise the value of **Max** because the resulting decompositions are not likely to be very efficient. Lowering this value has the effect of forcing the zones to be smaller which can be beneficial on cache based machines as mentioned earlier. Lowering the value of **Min** can be done for the same reason and also if the user is having difficulty fitting the problem into memory or achieving a valid target efficiency in cases where the zone sizes vary a lot. Raising the value of **Min** can have the effect of causing the algorithm to use explicit partitioning to fit into memory over breaking up the zones smaller. Explicit partitioning is an automated feature of the load balancing algorithm.

After pressing the **Auto Dec** button, new partition values will be computed, and the value of the newly computed efficiency will be displayed in the **New** efficiency display. Valid ranges are from 0 to 1. A value of -1 indicates that a valid partitioning is not possible with the current inputs. If the algorithm is unable to meet the **Target** efficiency, but can find at least one valid efficiency, then the highest efficiency achieved will be returned. The efficiency is a function of the grid sizes, the **Min** and **Max** grid point values, the **Lock** toggle button, the number of processors, the scale factor of the processors, and the memory available for each processor. The last three inputs are described in the **Physical Resources** tab section. So a value of -1 often indicates that there is not enough memory available. The user must then either change the memory input (although this value is typically fixed at the limit for the given machine), or lower the **Min** grid points value to reduce memory requirements.

The current load balancing algorithm does not take into account the possible differences in physical modeling or time integration techniques. If a user has zones with differing number of equations or significantly different time integration methods, then it may be necessary to do a manual decomposition. The best approach in such a case would probably be to first use the **Auto Dec**, and then modify some of the partition values manually to adjust for the discrepancy being sure to then **Lock** the manually edited zones so that the user values are retained.

Zones Per Proc Automated Decomposition

For some cases that involve running on large numbers of processors (1000+), the communication costs associated with having several zones on each processor can outweigh the benefit of ideal load balancing of the work. This is typically not the case and so this algorithm is not usually recommended, but it is available for those limited cases for which it might prove more effective.

Fig. 15.9 on pg. 266 shows the GUI layout when this method is selected. The **Zones/Proc** counter selects the exact number of zones that will be assigned to each processor - the default choice is 1 to get the minimum possible communication overhead. In order for this algorithm

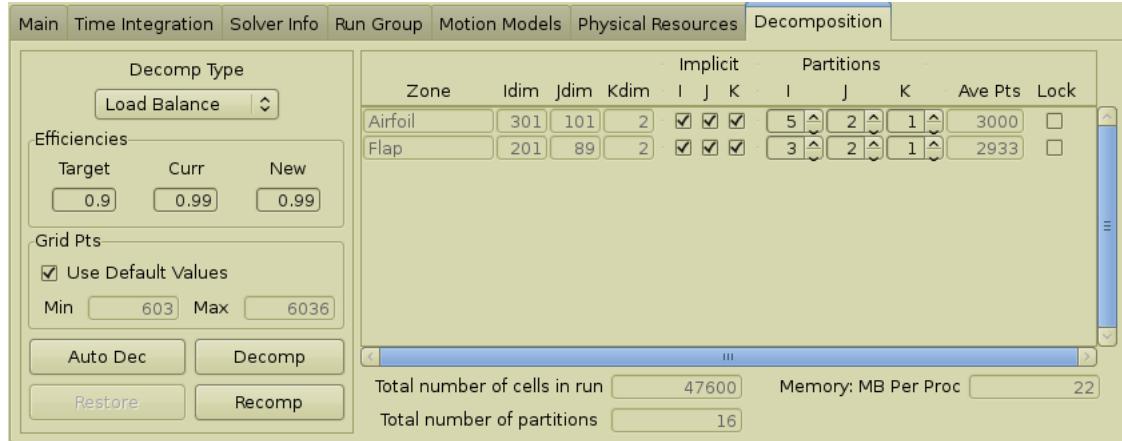


Figure 15.8: Zone Decomposition tab for Load Balancing algorithm

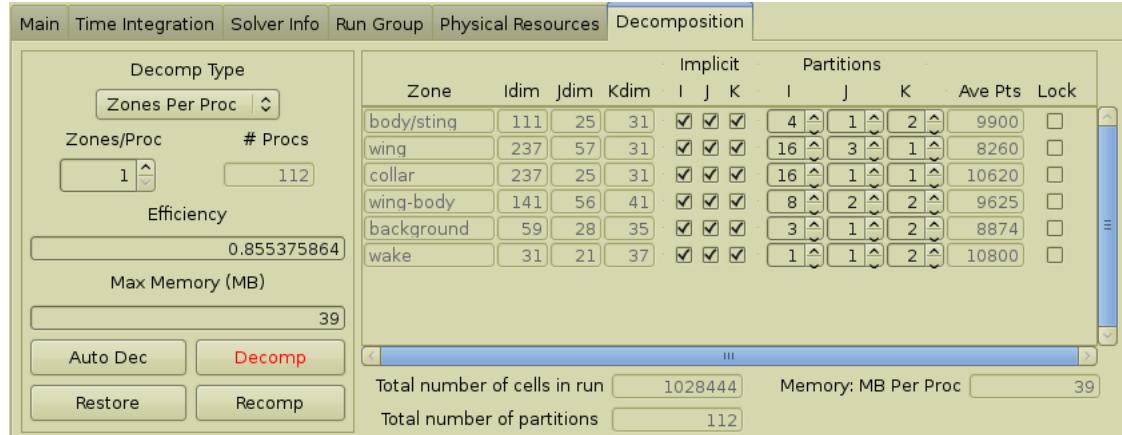


Figure 15.9: Zone Decomposition tab for Zones-Per-Proc algorithm

to be effective, the exact number of processors to be used is controlled by the algorithm and not by the user input in the **Physical Resources** section. The number of processors input by the user will be used as a guideline, and the algorithm will choose a number of processors as close as possible to this value that meets the **Zones/Proc** criteria. The **Max Memory (MB)** is a computed value that displays the amount of memory required since the memory limits in the **Physical Resources** are not used by the algorithm as they are in the load balancing algorithm. It is up to the user to determine after the algorithm has run if the amount given by the **Max Memory (MB)** is appropriate.

15.8.2 Creating a Decomposition

To actually create the new partitions (to decomp a zone), you must press the **Decomp** button. Pressing the **Auto Dec** button will only set-up (or prepare) the zones for decomposition. Notice that after you use the **Auto Dec** feature, but before you actually perform the decomp,

the changes in the decomposition are high-lighted in red. The **Decomp** button is also highlighted in red to let the user know to press it in order to have the changes stored. The reason for the two-step process is that it allows the user to preview the results of the **Auto Dec** before committing to computing the partitions.

After pressing the **Auto Dec** button, the **New** efficiency lets the user know what the resulting efficiency will be. The **Curr** efficiency lets the user know what the efficiency of the currently stored decomp is without any of the changes (which are high-highlighted in red). The **Restore** button will restore all of the input settings back to the values of the current decomp (i.e. it will change the red values back to their previously decompiled values). The **Restore** button is grayed out when all of the inputs are consistent with the current decomp.

The **Recomp** button is used to remove all of the zone partitions and recompose each zone back into a single partition. After a recomp, all of the partition values (I, J, and K for structured grids) will be set to 1. The **New** efficiency and **Curr** efficiency will also be recomputed based on these new settings. Often times they will be -1 after a recomp, because many problems which are to be run in parallel require some decomposition (zone partitioning).

15.8.3 Implicit Partition Boundaries

The **Implicit** buttons labeled “I”, “J”, and “K” for a structured grid or “**Impl Part**” for unstructured grids show the user which partition boundaries are implicitly solved and which ones will be solved explicitly. The ability to set none, some or all of the partition boundaries give the automated **Load Balance** algorithm (and the user in the case of manual decomposing) some control over the amount of memory required to run a problem. This capability is currently not implemented for the **Zones-Per-Proc** decomposition algorithm. The default for all of these values is true (or on). This means that regardless of the partitioning in the zone, the entire zone will be solved implicitly (*i.e.*, the partition boundaries are implicit). In order to achieve this, the linear solver information must be stored over the entire zone which can translate into large memory requirements for large three-dimensional problems.

This is where the **Implicit** buttons come into play. The **Load Balance** algorithm will adjust the **Implicit** settings in order to run the case within the memory specified. The user should only need to set the memory available (in the physical resources section). For example, if the load balancing sets the **Implicit** button to false in the I direction of a given zone, and that zone has 4 partitions in I, then the linear solver memory is reduced by about a factor of 4. This memory reduction is done at the expense of convergence rate. The solution would still be shared across partition boundaries between cycles as in the implicit case, but the linear solvers for the 4 partitions in our example would update the solution in each partition independently (*i.e.*, they would not share updates to the solution as they are solved). This is what we call an explicit partition boundary, and the end result is slower convergence. In the case of a structured grid, if the user does not like the direction the load balancing algorithm chose to make explicit for a given zone, the user can set the **Implicit** values in that zone along with specifying the partition values and then set the **Lock** to true. The

load balancing algorithm can then be recalculated with these fixed values. Whether or not a valid decomposition is reached with these values fixed will depend on the typical algorithm factors with the new constraint applied.

The choice of the time-integration scheme is also important when setting the decomposition parameters. The LU and 2-Factor AF methods will always have explicit partition boundaries regardless of the **Implicit** settings. So the user should consider this when running a marching problem (LU & 2-Factor AF are only selectable when marching) that is to be decomped. The Jacobi algorithm is completely independent of the decomp when all the partition boundaries are implicit (*i.e.*, a decomped run will yield the same convergence as a non-decomped run). The SGS algorithm utilizes the **Implicit** settings, but it is not completely independent of decomping because some of the solution update information is lagged in order to keep the message passing simple and efficient. The end result is improved convergence with implicit over explicit partition boundaries, but not a perfect match of decomped and non-decomped convergence rates.

Typically when setting up a problem, the user is unsure as to what the ultimate decomposition will look like. This is one of the main reasons the decomposition is set up as a two-step process. It allows for simple trial and error to get a desirable result. The following is a simple example which illustrates this.

Decomposition Example

Consider a six-zone Chimera case that is set-up to run on 6 processors with 150 MB per processor. Without making any changes to the decomposition, the zone decomposition is shown in Fig. 15.10 on pg. 269 and the **Curr** and **New** efficiency is -1 meaning that the current settings are invalid for running the case. This is due to the specified memory (150 MB per processor) and the fact that some of the zones have over 300k cells.

By pressing the **Auto Dec** button, all the zones are broken up into smaller zones or partitions. See Fig. 15.11 on pg. 269 for the result. There are several zones that also have the implicit zonal boundary passing turned off and the average zone size is around 30,000 cells. The algorithm uses explicit passing for partitions for this case in the *I* direction, so that the number of decomped zones stays within the **Min** grid points limit and a valid decomp can be achieved.

It turns out that the *I* direction is a good choice for this problem because it is the largest grid dimension for most of the zones, and the boundary layers are normal to the *K* surfaces. Suppose *K* had been chosen by the automated algorithm which would possibly be the worst choice due to the gradients in the boundary layer. The user could override this by manually setting the explicit partition in another direction for those zones as well as the number of partition boundaries in each direction and then applied the partitioning **Lock** before pressing the **Auto Dec** button.

Using explicit partition boundaries will also force explicit zonal boundaries between the six original zones. Whenever one or more of the **Implicit** settings is set to false, the value for **Implicit Zonal Boundaries** in the **Run Group** tab is overridden and treated as if it were set

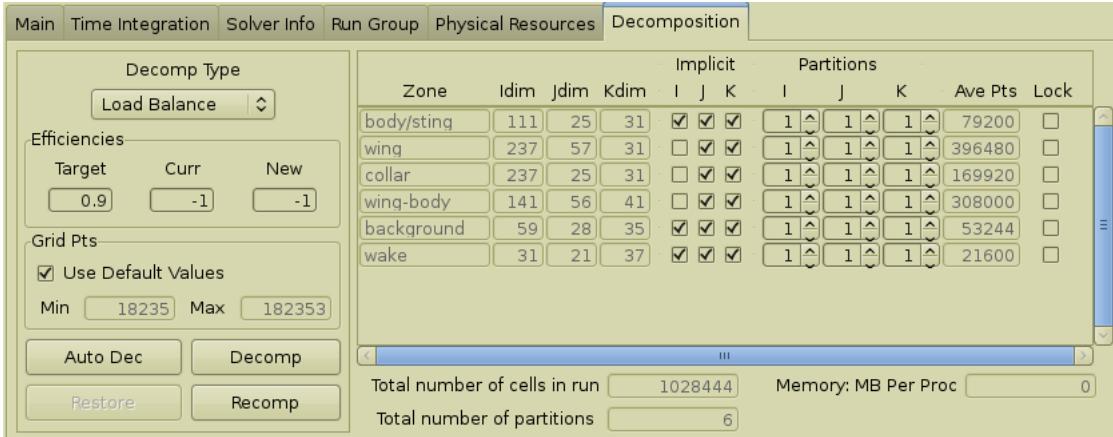


Figure 15.10: A decomposition example in which 6 processors with 150 MB each is used to run a problem with over 1 million grid cells. Without any partitioning, the problem cannot fit into the specified memory (indicated by the -1 efficiency value).

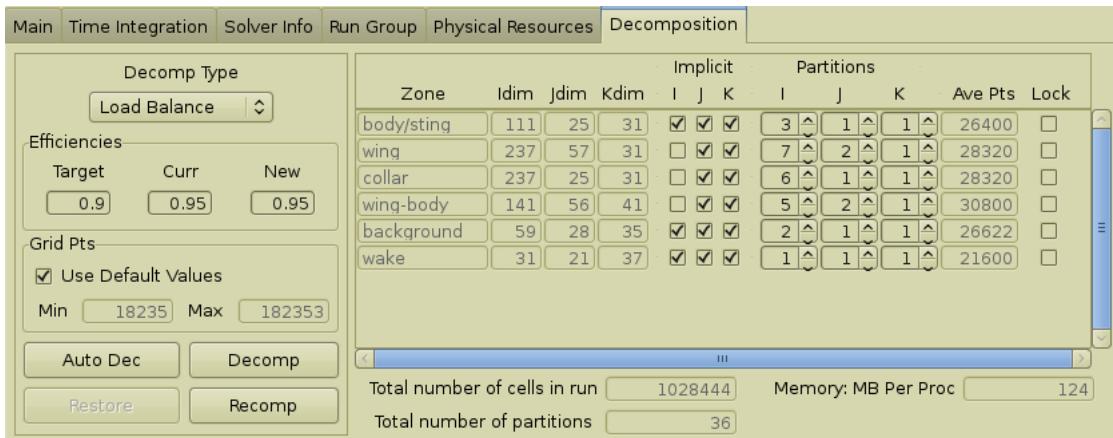


Figure 15.11: The final decomposition which uses explicit partition boundaries in the I direction.

to false. This now appears to be a good decomposition.

15.8.4 Number of Partition Boundaries

The **Partitions** input define the number of partitions that will exist in each coordinate direction (I , J , and K) for a structured zone. A value of 1 indicates that no decomposition will be done in the given direction, a value of 2 indicates that the zone will be decomposed into 2 partitions (*i.e.*, 1 partition boundary) in the given direction. The total number of partitions for a given zone will therefore be equal to the multiplication of the three partition values. For example, partition values of 2, 2, and 4 will result in a zone with 16 partitions. Since the zone is decomposed into partitions of approximately equal sizes, the **Ave Pts** (average

number of points) in each of the partitions will be approximately the total number of points in the zone divided by the total number of partitions. For unstructured grids, the user inputs the **Total Partitions** for a zone.

15.8.5 Setting Partition Values Manually

Typically, the **Partitions** input is determined through the use of the **Auto Dec** feature (see Sec. 15.8.1 on pg. 264). However, there are cases where the user will want to set the values manually or make adjustments to the **Auto Dec** results. This should ONLY be done when using the default **Load Balance** algorithm and NEVER for the **Zones-Per-Proc** algorithm. One example of this is when there are factors in the problem set-up that are not taken into account by the **Auto Dec** algorithm. Such factors include varying physical models (*i.e.*, differing number of equations to solve) and varying time integration techniques (*e.g.*, one zone uses more inner iterations than another).

Another case where the user may want to set the **Partitions** values manually is when the result of the **Auto Dec** puts one or more partition boundaries in a region with large gradients or important flow structures. This becomes important to consider when the user is also forced to use explicit partition boundaries either through the choice of time integration scheme (LU and 2-factor AF always use explicit partition boundaries) or because of memory limitations (see Sec. 15.8.3 on pg. 267).

Setting the partition values manually requires the use of the **Lock** setting. The purpose of the **Lock** setting is to allow the user to manually set the **Implicit** and **Partitions** values for one or more of the zones, and then be able to use the **Auto Dec** algorithm to determine the best decomposition for the remaining zones. This feature also ONLY works with the **Load Balance** algorithm and will be ignored by the **Zones-Per-Proc** algorithm.

Typically when considering using this feature, a user will first run the basic **Auto Dec** and see what kind of result are given. Then the user can tweak a few of the zones based on the desired result. The manually changed values will be displayed in red indicating that the changes are currently only temporary until the algorithm is re-run. In order to do this, the user would set the **Lock** value to true for the manually set zones, and then press the **Auto Dec** button. When one or more of the zones is locked the auto decomposition algorithm will use the partitioning in those zones as is, and will only alter the partition values in the remaining zones (*i.e.*, the zones where **Lock** is set to false).

15.8.6 Solver Decomposition

In some cases, the user may want the decomposition to be done by the solver. This can be specified by setting the **Check/auto Decomp in Solver** option on. When this is activated, the solver will perform an auto decomposition only if one of the following conditions are met.

- The current efficiency is lower than the target efficiency
- The current efficiency is -1 indicating that there is a memory issue

15.8.7 Reported Information

Located below the decomposition table are a number of useful data parameters. The “**Total number of cells in run**” reports the total cell count of all zones being used in the run definition. The zones used in the run definition are a result of the **Zone Family** setting in the run group.

The “**Total number of partitions**” is the summation of all zone partitions. This number will be updated after pressing the **Auto Dec** button in order to inform the user of the projected number of partitions. In a similar manner, the “**Memory: MB Per Proc**” reflects the projected amount of memory each processor will use. This is only a projection and may be lower or higher than the actual memory required.

Chapter 16

The Solution Visualization Section

16.1 Introduction

Post processing can be performed by visualizing the solution from within the GUI, or by exporting the solution to file for third party use. These tasks are found under the **Solution Visualization** and **File Output** sections respectively. A discussion of File Output can be found in Chap. 17 beginning on pg. 321. This chapter will discuss all of the aspects of solution visualization within the *GASpex* GUI. A visualization tutorial is also available for a hands-on learning experience (see Sec. 3.1.3 on pg. 26 for the location of the tutorials).

GASpex uses the Visualization ToolKit (VTK) [56] for all 3-D graphics rendered within the GUI. With the use of VTK comes a complete solution visualization package, including, but not limited to, cutting planes, iso-surfaces, stream traces, and animations. This chapter will discuss the process of creating solution visualizations from within the *GASpex* GUI. The user may find information regarding the three-dimensional graphics window, and the basic manipulation of objects in Chap. 6 beginning on pg. 73.

16.2 The VTK Pipeline

The concept of a graphics pipeline is utilized with the VTK graphics library. Within the pipeline, there are objects. These objects might be ordered data from a solution, iso-contours or iso-surfaces of a given piece of data, or particle traces. Each geometric object is placed within the pipeline, where there is a source, or upstream element for each object within the pipeline. Also, each object may, in turn, become a source for another object within the pipeline. For example, an iso-surface object requires, as input, a three-dimensional collection of data points consisting of physical coordinates as well as one or more variables to contour. The resulting iso-surface will be a surface of data points for the given iso-surface. This surface could then be used as input for displaying velocity vectors on that iso-surface. The user will create the VTK pipeline by connecting tree nodes within the “**Solution Visualization**” folder. Each object within the “**Solution Visualization**” folder may contain an “**Input**”

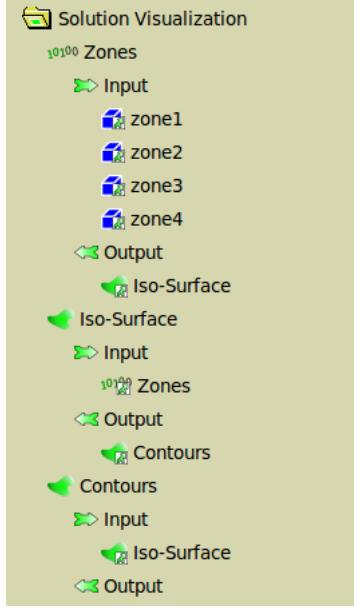


Figure 16.1: Examples of the VTK Pipeline within the tree view.

sub-folder and an “Output” sub-folder. The “Input” sub-folder will contain link nodes pointing to all of the objects within the pipeline which are to be used as input for the current element. The “Output” sub-folder will contain link nodes pointing to all of the objects for which the current object provides input. Fig. 16.1 on pg. 274 depicts a simple pipeline as described in the Tree View. Note that there are three objects within this figure; the “Zones” data node which provides access to the data at the nodes of all of the zones within this current problem, the “Iso-Surface” contour object, which will extract a single iso-surface from the Zones described in the data object, and the “Contours” contour object, which will define iso-contours of a different variable, extracted from the Iso-Surface. In this example, the “Zones” data object has an “Input” sub-folder containing link nodes pointing back to four zones within the current problem. Because the “Zones” object serves as input to the “Iso-Surface” object, you see a link node named “Iso-Surface” in the “Zones” object’s “Output” sub-folder. A “Zones” link object is also seen in the “Input” sub-folder of the “Iso-Surface” object.

There are two different ways to create object nodes within the “Solution Visualization” Folder. The first way to create nodes is through drag-n-drop of existing Zone or Surface nodes. The second way to create nodes is through the **Edit** Menu, (See Sec. 7.4.1 on pg. 85 associated with a solution visualization group folder, *e.g.*, the “Solution Visualization” Folder. Fig. 16.2 on pg. 275 shows the different VTK elements which can be created within the Solution Visualization section.

A single pipeline element, such as a data node, can serve as input to multiple pipeline elements. For example, a data node describing volume data for all the zones in a solution can be used as input to iso-surface nodes and particle tracing nodes simultaneously. Some

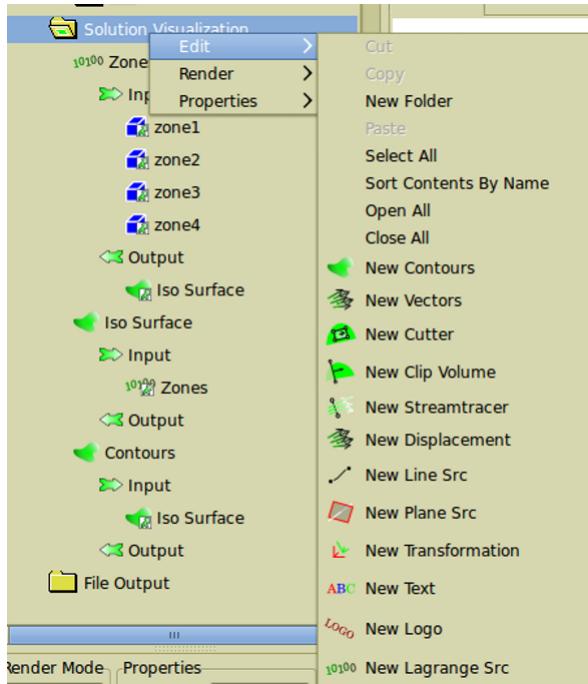


Figure 16.2: The **Edit** Menu for a VTK Group Folder.

VTK elements require more than one input. For example, the Cutter element which extracts cutting plane data requires as input the volume data to be cut, as well as an implicit function describing the geometry of the cut.

16.3 The Visualization Folder

Within the main “**Solution Visualization**” Folder, one can create an hierarchy of folders used to organize the visualization pipelines. As with any other folder, a Visualization Folder can be created by activating the **Edit** menu from any other Visualization Folder. All Visualization Nodes can be moved and reorganized within the main Solution Visualization hierarchy. Any Visualization Objects can be created as a child of any Visualization Folder.

16.4 The Data Node

In most situations, the first pipeline element to be created will be the Data Node. The Data Node represents a collection of data points, data point connectivity, and the data point variables. A Data Node will have one or more Zone Nodes associated with it as well. For each associated Zone Node, the user will control which grid points, control volumes or faces are to be included in the node definition. The user will create a Data Node by selecting one or more Zone Nodes, Zone Folders, Surface Nodes, or Surface Folders. Once

selected, these nodes are drag-n-dropped into a Solution Visualization Folder. Once the drag-n-drop operation is completed, a single Data Node is created within the parent folder. If possible, visualization properties and names will be copied from the dependent Zone and Surface nodes. You will recall that the first node in Fig. 16.1 on pg. 274 represents a Data Node created by dropping the “Zones” node into the “Solution Visualization” folder. The “Zones” folder contained four zones. The Data Node will likewise contain four Zone Link Nodes corresponding to the individual Zone Nodes found in the “Zones” folder. With the **Solution Visualization** section activated, and the “Zones” node selected, you will see all of the properties associated with the Data Node, as depicted in Fig. 16.3 on pg. 277. The notebook of property sheets found under the **Solution Visualization** section will vary depending upon which visualization node is selected. Only the properties associated with the selected nodes will be visible. There are four different properties associated with a Data Node, three of which are unique, and one of which is common to most visualization nodes. The **Variables** property and the **Sequence** property are unique to the Data Node. The **Grid Range** property is defined at the Zone Link Node level. Each Zone Link Node found in a Data Node can maintain different values stored at the **Grid Range** level. The **Mapper** property is stored at the Filter level, and most VTK Filter Nodes, including the Data Node, contain the Mapper Property.

Note: A quick overview of setting up a data node is as follows:

- Select either a zone node or a surface node and perform a drag-n-drop into the Solution Visualization folder.
- Add variables to the Active Scalars list.
- Select the variable to view using the mapper properties.
- Adjust the grid range and render the data node to Full for visualization.

16.4.1 Solution Variables

The solution variables used by this and dependent filters are specified by selecting the **Variables** tab in the **Solution Visualization** section. The default settings are shown in Fig. 16.4 on pg. 277. In addition to defining the visualization variables, the display unit system is also prescribed here.

These variables are associated only with Data Node filters and any downstream filters. The interface is enabled when a single visualization node is selected. The Variable Property displayed will be that of the most upstream Data Node in the selected pipeline.

Note that unlike the Solution Variables section of the Output Node entries, (see Sec. 16.4.1 on pg. 276) there are two sections in this property, the **Scalars**, and the **Vectors**. The **Scalars** list will be available to any filter function requiring scalar data, while the **Vectors** list will be available to any filter requiring a vector function, such as stream tracing or velocity vectors.

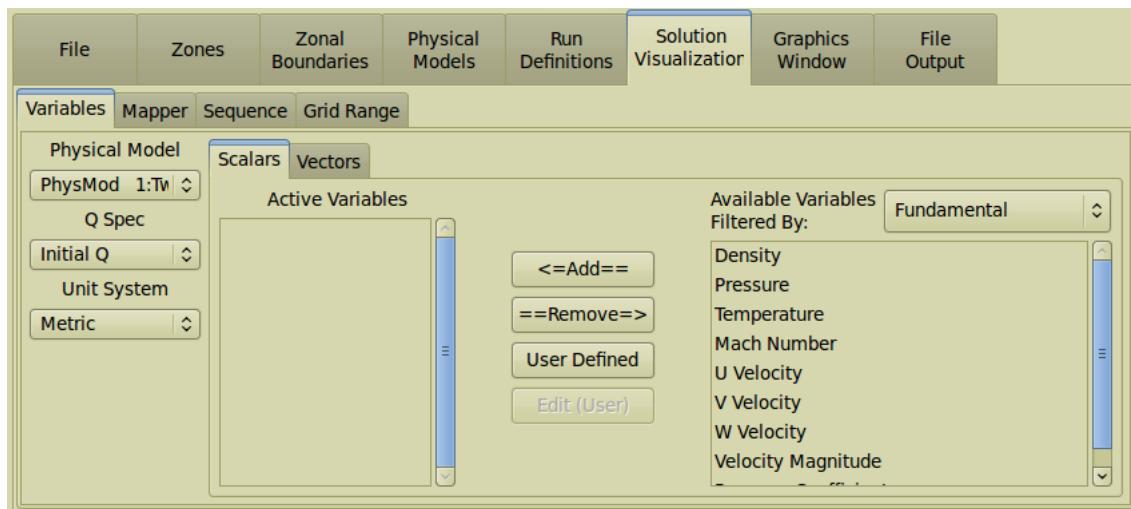


Figure 16.3: The Properties presented to the user when a Data Node is selected.

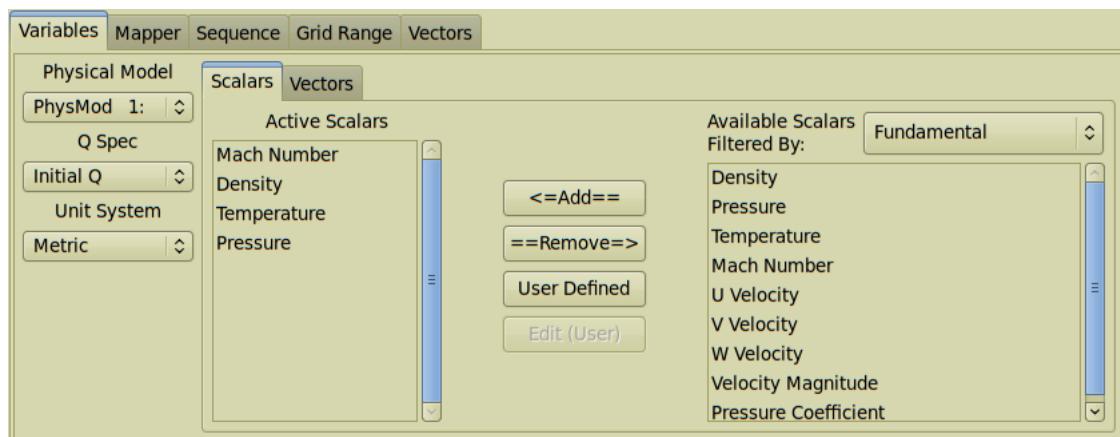


Figure 16.4: The Variables Tab

The Scalars Selection

Available Scalars Filtered By Option

*GAS*Pex supports the display of more than one hundred variables. These variables have been organized into categories for efficient presentation. The **Available Scalars Filtered By:** option menu is used to select the visualization variable category to be presented in the **Available Scalars** list. Some variables are independent of physical model and are available for any solution. Other variables may or may not be available, depending upon the currently selected physical model. Likewise, some variables are only appropriate when face type interpolation is active in the grid-range property.

A complete list of all variables available for solution visualization is given in Sec. 17.4 on pg. 339.

The Active Scalars List

In Fig. 16.4 on pg. 277, the **Active Scalars** list on the left displays the solution variables for the referenced Data Node. These variables are available to the Data Node as well as any downstream elements connected to the Data Node in the visualization pipeline. For example, if the user wishes to display Mach contours on a cutting surface, and create a pressure isosurface, then both Mach number and pressure would need to be added to the Active Variables List. The Active Scalars list is made visible to all downstream elements in the pipeline. Initially, the **Active Scalars** list is empty. Variables are added by selecting from the **Available Scalars** list and pressing the **<==Add==>** button located between the two respective lists. Variables can be inserted by selecting a variable in the **Active Scalars** list and clicking **<==Add==>**. If no variable is selected in the **Active Scalars** list, then added variables are appended at the bottom. Variables may be deleted from the **Active Scalars** list by selecting them and clicking on the **==Remove==>** button. User-defined variables can be constructed by clicking on the **User Defined** and defining the variable and units.

The Physical Model Option Menu

Some of the visualization variables depend on the modeling parameters found in a physical model. For example, the internal energy of a system depends on the thermodynamic model found in the **Thermo-Chem** section of the **Physical Models** tab. Likewise, the skin-friction calculation depends upon the viscous modeling parameters selected in the **Viscous** section.

Referring to the left-hand part of Fig. 16.4 on pg. 277, the physical model for output is selected by using the **Physical Model** option menu. Care should be taken to ensure that an appropriate physical model is selected for the zones being analyzed.

*Many of the visualization variables displayed in the **Available Scalars** list depend on the currently selected physical model. For example, turbulent quantities are not available if the current physical model describes inviscid or laminar flow.*

The Q Spec Option Menu

If any of the output variables depend on reference quantities, these values are computed

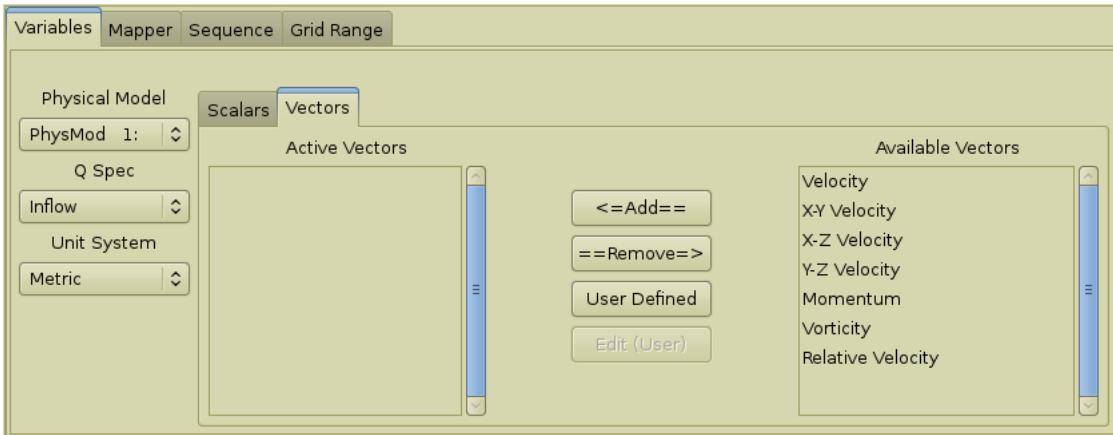


Figure 16.5: The Vectors section beneath the Variables Tab

from the current **Q Spec** option menu for the selected physical model. Example variables include the pressure coefficient or heat-transfer coefficient.

The Unit System Option Menu

Variables are displayed in the unit system selected under the **Unit System** option menu. The visualization unit system is independent of the unit system used for describing the input parameters for the current problem. The unit system for the current Data Node is also independent of other Data Nodes.

The Vectors Selection

Available Vectors

Selecting the **Vectors** tab under the **Solution Variables** property will display a notebook similar to that shown in Fig. 16.5 on pg. 279. The number of vector quantities available for use in visualization is limited. Because of the reduced number of variables, there is not a filter for the **Vectors** notebook.

Velocity The x , y and z components of the velocity, (u, v, w) , are inserted into the three components of this vector.

X-Y Velocity The x and y components of the velocity, (u, v) , are inserted into the first two components of this vector, the third component is set to 0.

X-Z Velocity The x and z components of the velocity, (u, w) , are inserted into the first and third components of this vector, the second component is set to 0.

Y-Z Velocity The y and z components of the velocity, (v, w) , are inserted into the second and third components of this vector, the first component is set to 0.

Momentum The x , y , and z components of the mass momentum per unit volume, $(\rho u, \rho v, \rho w)$, are inserted into the three components of this vector.

Vorticity The Vorticity vector represents the X, Y, and Z components of vorticity, $\nabla \times \vec{V}$.

Relative Velocity The relative x , y and z components of the velocity when a rotating frame is simulated.

The Active Vectors List

In Fig. 16.5 on pg. 279, the **Active Vectors** list on the left displays the visualization vectors for the selected Data Node. These vectors are available for use by any filter connected downstream of the current Data Node. Initially, the **Active Vectors** list is empty. Vectors are added by selecting from **Available Vectors** list and pressing the **<==Add==>** button located between the two respective lists. Variable can be inserted by selecting a variable in the **Active Vectors** list and clicking **<==Add==>**. If no variable is selected in the **Active Vectors** list, then added variables are appended at the bottom. Variables may be deleted from the **Active Vectors** list by selecting them and clicking on the **==Remove==>** button. User-defined variables can be constructed by clicking on the **User Defined** and defining the variable and units.

16.4.2 Sequence

The solution variables extracted from the zones in a Data Node need to be associated with a grid level sequence. The **Sequence** tab in the **Solution Visualization** section as shown in Fig. 16.6 on pg. 281 is used for this purpose. The user has two options for controlling the sequence to which a Data Node refers. First, the user may allow the Visualization to be performed on the currently selected sequence. Following the current sequence is the default behavior. Selecting the **Set To Current** check box will force the visualization of the current Data Node to follow the current sequence selection as described in Sec. 8.1 on pg. 87. If the **Set To Current** check box is not selected, the **Fixed To** sequence selection option menu is enabled. Selecting a given sequence from this option menu will force the visualization associated with this Data Node to be associated with the selected sequence.

16.4.3 The Grid Range Property

Parameters relevant to the computational coordinates are edited by selecting the **Grid Range** tab shown in Fig. 16.7 on pg. 281. In particular, the type of range (*e.g.*, volume, surface, line or point), interpolation type (*e.g.*, nodes, cell centers, faces), cell types (relevant to Chimera meshes) and the min/max coordinates in the I , J and K direction are specified. Fig. 16.7 on pg. 281 shows values relevant to a structured discretization, while Fig. 16.8 on pg. 281 shows values relevant to an unstructured discretization. For the current version, the values shown for unstructured zones have no impact on the display of the selected Data Link Node.



Figure 16.6: The Sequence Tab

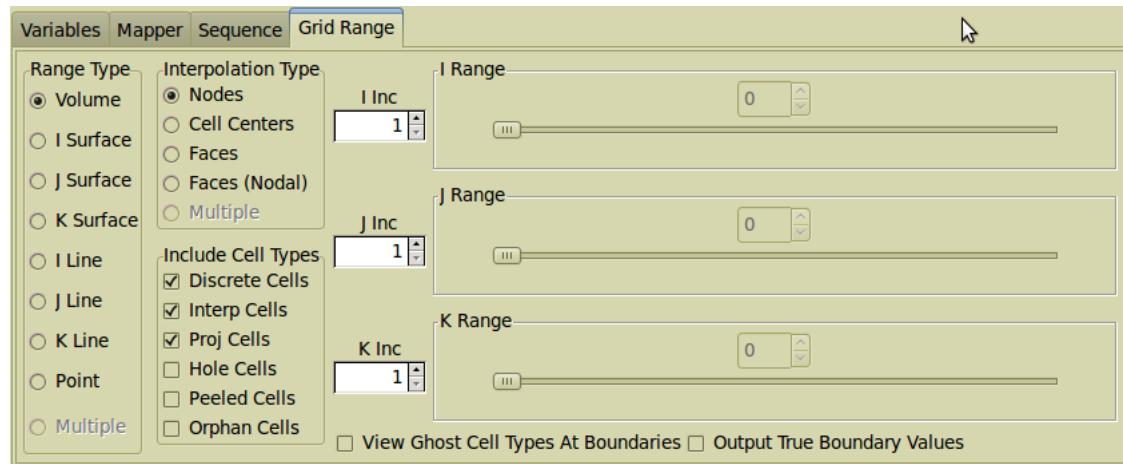


Figure 16.7: The Grid-Range Property Window for a structured grid

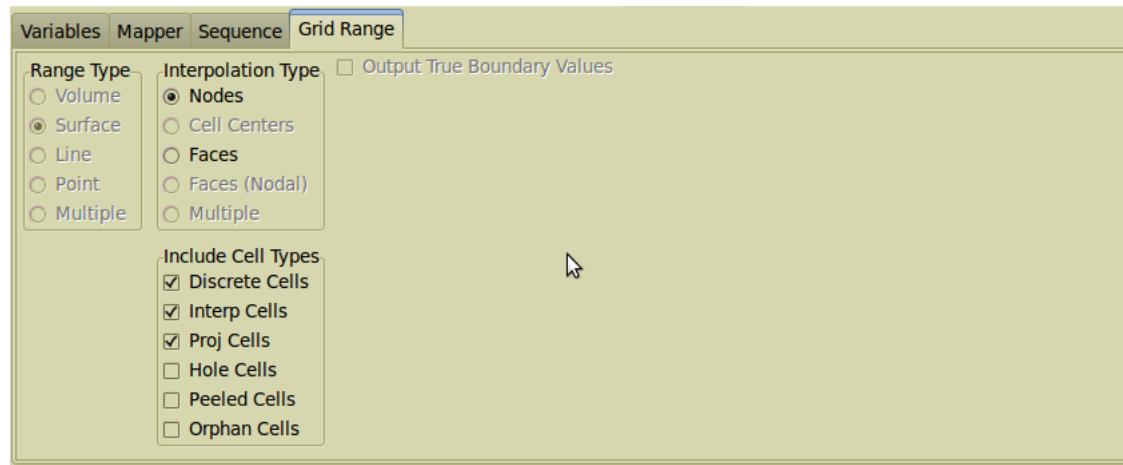


Figure 16.8: The Grid-Range Property Window for an unstructured grid

Each Data Link Node has a distinct grid-range definition. The user can set all of the parameters for a node independent of the other nodes. The GUI for the grid-range property is capable of simultaneously editing parameters for more than one Zone Link Node. Whenever a Data Node or a collection of Zone Link Nodes is selected, the grid-range property display represents the collective values of the selected Zone Link Nodes. Changing a parameter in the grid-range tab changes that parameter for all selected nodes. If changing a parameter for a multiple selection is not possible, then the appropriate widget will be disabled. In general, editing operations should be performed while the parent Data Node is selected, so that changes may be made to both the grid-range property and the solution property without changing node selections. Editing a grid-range property on a single node typically occurs only when merged editing is not possible.

The Range Type Selection Widget

Volumes, surfaces, lines and points are called “range types”. Radio buttons are selectable in the **Range Type** section shown on the left side of Fig. 16.7 on pg. 281. Each range type controls the sliders for selecting the logical coordinates shown on the right of the window. We describe each range type below.

Volume The **Volume** range type is primarily used to provide three-dimensional data to filters such as cutting planes, iso-surface generators and particle tracers. When the range type is volume, all of the **I Range**, **J Range** and **K Range** sliders will have a minimum and maximum value.

The **Volume** range type is also the most appropriate choice for visualizing overset grids. When used in conjunction with the the **Include Cell Types** toggle buttons, the user may easily identify hole cells and orphan cells created during the hole cutting and interpolation coefficient creation process.

The volume range type is the default value for Data Nodes created by interactively dropping zone nodes into a Visualization Folder.

I Surface, J Surface, K Surface The **I Surface**, **J Surface** and **K Surface** range types may be used to create a reduced data set for visualization of surface based data. When the range type is one of the surface types, two of the **I Range**, **J Range** and **K Range** sliders will have a minimum and maximum values. The third will have a single value indicating which surface is being viewed.

I Line, J Line, K Line The **I Line**, **J Line** and **K Line** range types are most often used to produce line plots or monitor flow profiles. With a line range type, only one of the **I Range**, **J Range** and **K Range** sliders has a minimum and maximum value. The remaining two have a single value indicating which line is being viewed for output. The line type is also a good choice as a set of seed locations for particle tracing.

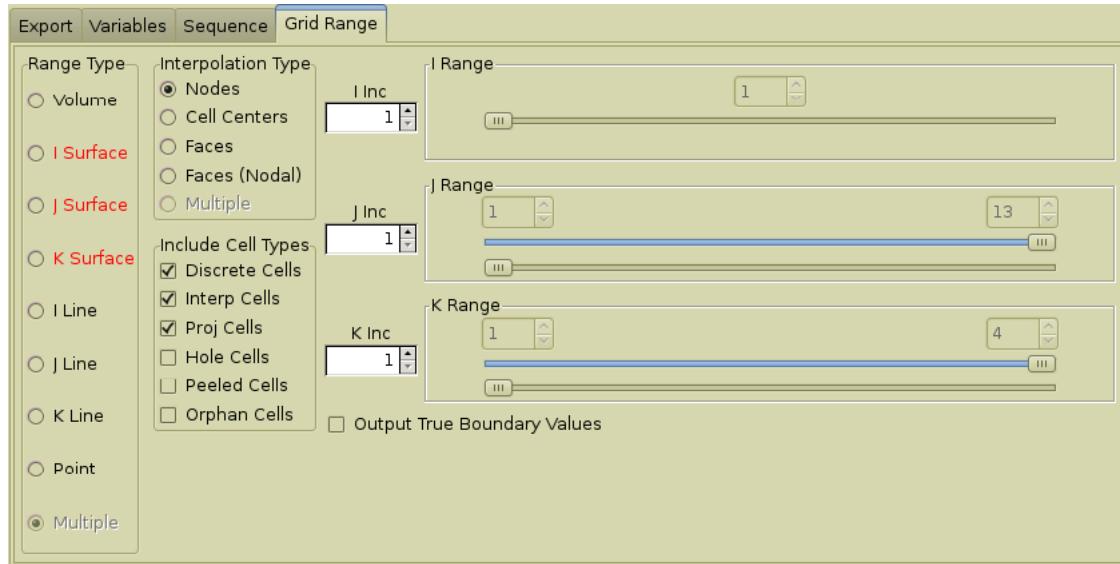


Figure 16.9: The Range Type Widget displaying multiple types.

Point The **Point** range type is most often used as a particle seed when used in visualization. When the **Point** range type is selected, all three of the **I Range**, **J Range** and **K Range** sliders have a single value indicating which point is being specified.

Multiple Because the **Grid Range** tab can represent a collection of Zone Link Nodes, the range type for the individual nodes may be different. When all selected nodes have the same range type, the widget will display that single value, and the button color will be normal. However, if the selected nodes have differing range types, the **Multiple** will be selected, and all range types from the selected nodes are colored red. For example, Fig. 16.9 on pg. 283 shows the appropriate display of a collection of nodes which include all three surfaces.

The Interpolation Type Selection Widget

Quantities selected for output are naturally defined at particular physical coordinates within the domain. For example, the solution variables (pressure, density, velocity, etc.) are computed and stored at the cell centers. However, grid coordinates are input and stored at the nodes. Likewise, mass flows, and viscous forces are computed on faces. If the user requires output at a location different from the natural location, interpolation must be performed. The **Interpolation Type** section identifies the interpolation location for the current output node.

Nodes The nodal coordinates are those provided directly from the grid generator to *GASpex*. Because solution variables are stored at cell centers, the cell-centered values must be averaged to the nodes. When the interpolation type is **Nodes**, the solution is averaged from the surrounding cell centers.

Interpolation to nodes is recommended when visualizing a chimera system. When visualizing nodal values with holes or orphans, the blanking information does not need to be averaged. Therefore, interpolation to nodes provides the most accurate representation of hole, peeling, and orphan cells.

Rendering real surfaces is best performed using interpolation to nodes. Because the grid coordinates naturally exist at the nodes, no interpolation of the geometry is necessary. This will yield the most accurate geometric representation.

Cell Centers When the interpolation type is **Cell Centers**, the grid points are interpolated to the cell centers. This point is computed as the centroid of the control volume, which is defined by the eight surrounding grid points. Selecting the **Line** and **Point** range types in combination with interpolation to cell centers allows the most accurate display of discrete flow-field quantities such as density, pressure, temperature and Mach number.

When a cell-center value is computed at a boundary (minimum or maximum cell-center range), the value is assumed to be on the boundary face. *GAS*Pex will not output values outside of the grid domain (*i.e.*, ghost cells). Solution data at a boundary is considered located in a “ghost cell” if the flux is split on that boundary. In this case, the first interior data and ghost data are averaged to the cell face. If the boundary is a full-flux boundary, then the actual boundary value is output.

When interpolating blanking information to cell centers, the actual hole cells or orphan cells may appear larger or smaller than in actuality.

Faces Interpolation to faces is most often used in conjunction with output quantities which are related to the evaluation of an inviscid or viscous flux (*i.e.*, mass flow, pressure or viscous forces, and heat transfer). Interpolation to faces is only appropriate when the **Range Type** is set to **Surface**.

Multiple Because the **Grid Range** tab may represent a collection of output nodes, the range type for the individual nodes may be different. When all selected nodes have the same range type, the widget will display that single value, and the button color will be normal. However, if the selected nodes have differing range types, the **Multiple** will be selected, and all range types from the selected nodes are colored red. For example, Fig. 16.10 on pg. 285 shows the appropriate display of a collection of nodes which include both interpolation to **Nodes** and **Cell Centers**.

The **Interp Cell Types** Toggle Buttons

The **Interp Cell Types** toggle buttons define the blanking information output to third-party, post-processing packages. Only those points with an active cell type will have their blanking value set to true. It is therefore possible to reverse the traditional definition of the blanking field for third-party packages (*e.g.*, the user can force the display of hole cells, and prevent

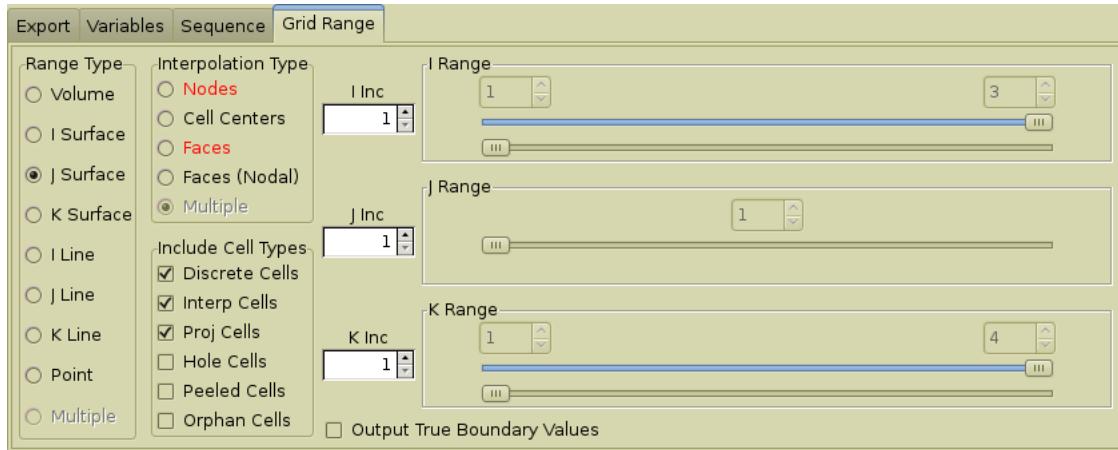


Figure 16.10: The Interpolation Type Widget displaying multiple types.

the display of discrete and interpolation cells.) The user can also decide whether or not to include interpolation cells in the display or to treat them as hole cells.

Discrete Cells “Discrete cells” are those cells that are updated by the flow solver. In the absence of overset grids, all points will have the discrete cell type. Toggling the **Discrete Cells** button will enable or disable the display of discrete cells. In general, discrete cells should always be enabled unless the user wishes to identify hole or orphan cells.

Interp Cells “Interp cells” exist next to the surfaces located in the **Chimera ZB** folder, as well as directly adjacent to hole cells assuming that an interpolation stencil can be found for adjacent cells. Toggling **Interp Cells** will enable or disable the display of the interpolation cells in an overset grid system. In general, it is best to include interp cells when visualizing overset grid solutions. Otherwise visual gaps may appear in the solution near the hole cells because the discrete cells may not completely overlap. Interp cells are updated through the passing of information from overlapping grids.

Proj Cells “Proj Cells” are created near surfaces where surface projection occurs – most often near collar grids about highly curved surfaces. Toggling **Proj Cells** will enable or disable the display of the projection cells in an overset grid system.

Hole Cells “Hole Cells” are created during the hole cutting process in the overset grid definition. (Note: hole cells are also created during the peeling process.) Toggling **Hole Cells** will enable or disable the display of the hole cells in an overset grid system. The solution variables associated with hole cells are invalid.

Peeling Cells “Peeled Cells” are created during the hole interpolation coefficient stencil selection process in the overset grid definition. Peeling cells are nothing more than

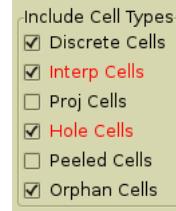


Figure 16.11: The **Include Cell Types** displaying multiple types.



Figure 16.12: The Range Widget in single slider mode

hole cells. Toggling **Peeled Cells** will enable or disable the display of the peeled cells in an overset-grid system. See Sec. 9.4.4 on pg. 137 for more information on peeling.

Orphan Cells “Orphan Cells” are created during the hole interpolation coefficient stencil selection process in the overset grid definition. Orphan cells represent candidate interpolation cells for which no interpolation stencil can be found. Toggling **Orphan Cells** will enable or disable the display of the orphan cells in an overset grid system.

Because the **Grid Range** tab may represent a collection of output nodes, the range type for the individual nodes may be different. When all selected nodes have the same cell display types, the widget will display the toggle buttons with default colors. If any of the different cell type display flags have differing values for the merged values, then the button will be colored red. For example, Fig. 16.11 on pg. 286 shows the appropriate display of a collection of nodes which have different values for the display of **Interp Cells** and **Hole Cells**, but identical values for **Discrete Cells** and **Orphan Cells**.

The I, J and K Range Widgets

The **I Range**, **J Range** and **K Range** sliders, allow the user to view and modify the limits of the logical coordinate directions for structured grids. The range sliders are multi-function, in that they either represent a single value (such as for describing a point), or a minimum and maximum value (such as for describing the limits of a coordinate line).

Single Slider Mode

The range widget will be in single-slider mode whenever there is a single value for the current coordinate direction. For example, the **I Range** widget will be in single-slider mode whenever the **Range Type** is set to **I Line**, **J Line**, **K Line**, or **Point** as shown in Fig. 16.12 on pg. 286.

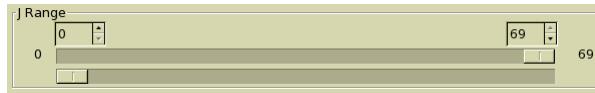


Figure 16.13: The Range Widget in multiple slider mode

The absolute minimum and maximum limits of the selection range are displayed to the left and right of the slider widget respectively. In the current example, the minimum value is 0, and the maximum value is 69. The thumb bar on the slider is positioned relative to the current value of the range. The current value of the selection is displayed in a type-in widget at the top center of the widget. For the current example, the current value is 0. The user may adjust the value either by positioning the slider, or by directly entering the desired value into the type-in.

Multiple Slider Mode

The range widget will be in multiple slider mode whenever there is a range of values being represented in the current coordinate direction. For example, the **I Range** widget will be in multiple slider mode whenever the Sec. 16.4.3 on pg. 282 is set to **Volume**, **J Surface**, **K Surface**, or **I Line**, as shown in Fig. 16.13 on pg. 287.

The absolute minimum and maximum limits of the selection range are displayed to the left and right of the slider widget respectively. In the current example, the minimum value is 0, and the maximum value is 69. The bottom slider will display and edit the minimum value for the current range, while the top slider will display and edit the maximum value. The thumb bar on the slider is positioned relative to the current value of the limit. The current minimum value of the range is displayed in a type-in widget located at the top left of the slider widget, while the current maximum value of the range is displayed in a type-in widget located at the top right of the slider widget. The user may adjust either value by positioning the slider, or by directly entering the desired value into the type-in. Note that if the minimum is set to a value larger than the maximum, then the maximum value will be adjusted so that it is greater than or equal to the minimum. Likewise the minimum value is adjusted such that it will remain less than or equal to the maximum value when the maximum value is adjusted.

Behavior of the Range Widgets When Viewing Multiple Nodes

Because the **Grid Range** tab may represent a collection of output nodes, the Range widgets for the individual nodes may be different. When all selected nodes have the same range type, the widget will display that single value, and the button color will be normal. However, if the selected nodes have differing range types, there are two different situations. There are three conditions which must be met before a range widget will allow simultaneous editing of multiple ranges. First, the slider mode must be equivalent for all ranges. In other words, it is not possible to simultaneously edit the ranges when one is displayed using Single Slider Mode, while the other is displayed using Multiple slider mode. The final two conditions are

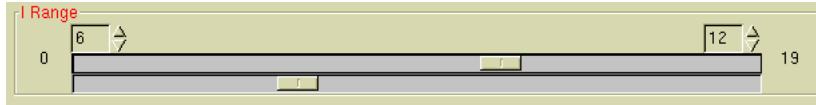


Figure 16.14: The Range Widget in multiple slider mode, displaying multiple ranges with different values.

that the absolute minimums and absolute maximums must be equivalent. Editing a range with absolute limits of 0-10 simultaneously with a range with absolute limits of 1-12 will not be permitted. If any of these three conditions are not met, then the range widget will be disabled, and the user will not be allowed to change the values. If the previous three conditions are met, but the actual current values of the ranges are different, then the range widget will allow the user to edit the values, and the label of the range will be colored red, to indicate that the values are different, as shown in Fig. 16.14 on pg. 288.

Note that the merging behavior for the I, J and K ranges will be treated independently. In other words, the I range may have identical values, the J Range may be different, and non-editable, while the K Range may be different and editable.

Absolute Range Limits Based Upon Interpolation Type

The absolute minimum and maximum values for a given range will depend upon the grid dimensions of the selected output node as well as the interpolation type.

Nodes When the interpolation type is **Nodes**, then the minimum values for all three logical directions will be equal to one. The maximum value will be set to the grid dimensions. Absolute minimum and maximum values will correspond to the boundary values of the current zone. For example, if viewing **I Surfaces** with **Nodes**, setting the **I Range** to 1 will correspond to viewing the I0 boundary values, interpolated to the nodes.

Cell Centers When the interpolation type is **Cell Centers**, then the minimum values for all three logical directions will be equal to zero. The maximum value will be set to the grid dimensions. Absolute minimum and maximum values will correspond to the boundary values of the current zone. For example, if viewing **I Surfaces**, with **Cell Centers**, setting the **I Range** equal to 0 will correspond to viewing the I0 boundary values.

Faces When the interpolation type is **Faces**, then the minimum values for all three logical directions will be equal to one. The maximum value will depend upon which Surface direction is selected in the Sec. 16.4.3 on pg. 282. If the **J Surface** is selected, then the maximum limits for the I and K directions will be one less than the grid dimension (which represents the total number of faces in that direction), while the J direction maximum will be equal to the grid dimension. The **I Surface** and **K Surface** will have similar limits.

View Ghost Cell Types At Boundaries

When ghost cells are included in a Chimera interpolation configuration, these cells can be difficult to visualize. Ghost Cells are not normally included in visualization. However, the cell type for a ghost cell can be copied to its immediate nearest interior neighbor. In this way the cell types for ghost cells can be visualized. Selecting the **View Ghost Cell Types At Boundaries** box will enable this behavior.

Output of True Boundary Values

The **Output True Boundary Values** option allows the user to override the handling of boundary values. In *GASPer*, a boundary value is stored at either the boundary face or the ghost cell. The ghost cell is simply a mirror reflection of the adjacent interior cell and is located outside the computational domain. Because of this, *GASPer* averages a boundary value located at a ghost cell with the first interior cell to attain the value at the boundary face. The **Output True Boundary Values** allows the user to override ghost cell averaging, such that no averaging is performed, regardless of the boundary value location. This option is only available for interpolation types of either cell centers or faces.

The location of a boundary value is determined from the boundary condition. For example, solid-wall boundary conditions store values at the boundary face, while inflow or outflow boundary conditions store values at the ghost cell. Under normal output conditions, the averaging of the ghost cell value with the interior value will represent the correct output value. But there are times when the user wishes to know the exact values at a boundary being enforced. An instance when this is true is when the user is verifying pointwise boundary values. To do so, the user should select the **Output True Boundary Values** option and output either face or cell center locations for the boundary surface. If output is at cell centers, the 0 index should be selected. In this way the output should be identical to the pointwise data supplied in the physical model.

16.4.4 The Solution Mapper

The mapping of solution values to a color map, and the application of this color map to a visual surface is controlled through the Mapper Property, as seen when selecting the **Mapper** tab in the **Solution Visualization** section. The interface for controlling values within this property is shown in Fig. 16.15 on pg. 290. Note that while the Mapper Property is documented in the Data Node section of this chapter, this property is associated with any VTK Filter Node which is connected through the pipeline with a Data Node. Each Filter Node containing a Mapper Property maintains values for this property independently from all other Filter Nodes. In addition to controlling the mapping of solutions to a color map, the Mapper Property also controls basic mirroring of geometry produced by the current Filter.

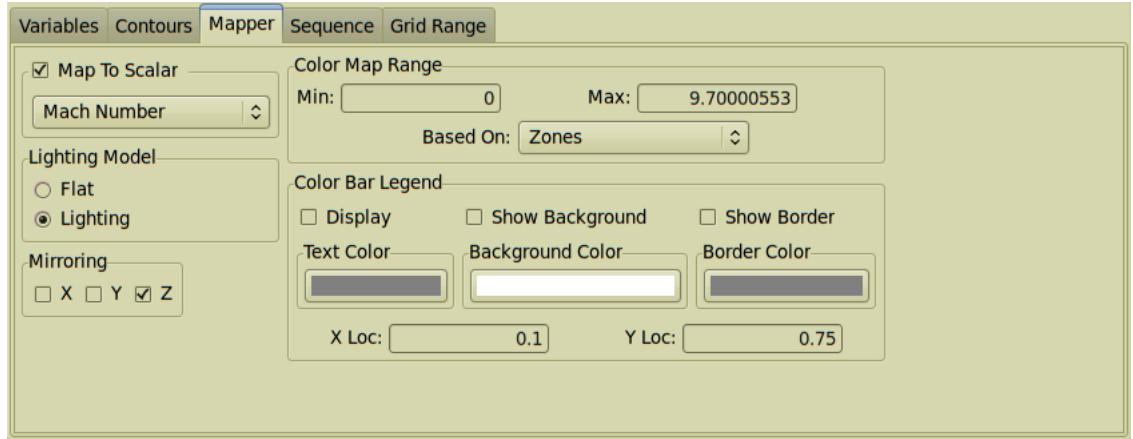


Figure 16.15: The Solution Mapper Tab

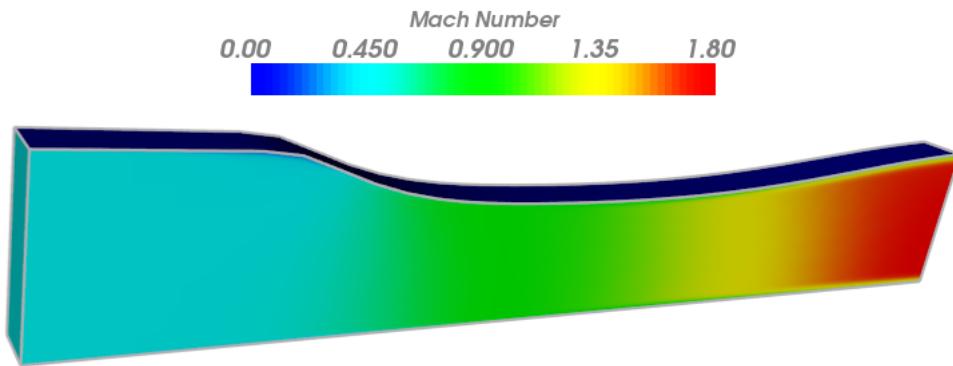


Figure 16.16: Example of a Volume Data Node rendered with a color map scaled to the local Mach number.

The Map To Scalar Frame

As discussed in Sec. 16.4.1 on pg. 276, the variables required for visualization must be selected in the **Variables** section. Once selected, the Data Node will compute a value for each variable at each point specified in the **Grid Range** section. A color map can be associated with the range of scalar values of a given variable. Then, each point on the displayed geometry can be drawn with the color associated with the scalar at that location. Fig. 16.16 on pg. 290 provides an example of the outer surfaces of the Duct tutorial rendered with a color map scaled to the local Mach number. To achieve the visualization rendered in this figure, the Mapper section must be set to map a scalar variable to a color ramp. Enable the **Map To Scalar** check box in order to visualize a scalar value mapped to the current Filter's geometry. Once the check box is enabled, and there are scalar variables selected, then the **Current Scalar** option menu will allow the user to select among all of the Scalar Variables in the **Active Variables** list, (see Sec. 16.4.1 on pg. 278).

The Color Map Range Frame

Once a **Current Scalar** variable is active, the values in the **Color Map Range** section are enabled. There must be a mapping of minimum and maximum values of the current scalar to the color map values. The color map provides a spectrum of colors from blue at the low end, varying through cyan, green, yellow, orange and red on the high end. There are several ways to accomplish the setting of these **Min:** and **Max:** scalar values. By default, the values for the color map range are set to the absolute minimum and maximum of the current Filter Node. The **Based On:** option menu will contain a list of Filter Node's which can be used to set the Min and Max values for the Color Map Range. The option menu list will be populated with the current Filter Node, as well as any Filter Nodes upstream of the pipeline, all the way to the the originating Data Node. Setting the **Based On:** option menu to a different value will change the **Min:** and **Max:** values to the minimum and maximum values of the selected Filter Node. The **Min:** and **Max:** values will always reflect the current values of the selected Filter Node. If any change in the input deck causes a change in the minimum and maximum values of the selected scalar on the selected Filter Node, then the Color Map Range will be updated to reflect this change. The user can directly enter values for **Min:** and **Max:** at any time. When this occurs, the **Based On:** option menu will automatically be set to USER DEFINED. When set to USER DEFINED, the **Min:** and **Max:** values will remain unchanged when minimum and maximum scalar values change.

The Color Bar Legend Frame

A legend showing the relationship of the colors to the scalar values can be displayed in the render window as an overlay to the current scene by selecting the **Display** button within the **Color Bar Legend** frame. Once the legend is visible in the viewing window the user can control its size and position interactively by pressing the left mouse button over the legend. Pressing and holding the left button over the middle of legend allows the position to be changed. Pressing and holding the left button at the legend edge allows the legend to be re-sized. Moving the legend to the left or right edge of the visualization window will transform a horizontally oriented legend to a vertically oriented legend. Likewise, moving the legend to the top or bottom edge of the window will transform a vertically oriented legend to a horizontally oriented legend.

The color bar is rendered as a two-dimensional overlay in the scene. This means that the color bar will cover all three-dimensional objects in the scene. However, by default, this object will only render the color bar gradient itself, and the text associated with the legend title, and the values of the scalar itself. The color of the text associated with the legend may be changed by pressing the color button labeled **Text Color**. Pressing this button will display a standard color dialog which may be used to change the text color. An opaque background can be rendered for the color bar by selecting the **Show Background** check box. The background color can be altered by pressing the color button labeled **Background Color**. An outline of the bounding box defining the legend can be drawn by selecting the **Show Border** check box. The color of the outline can be altered by pressing the color button

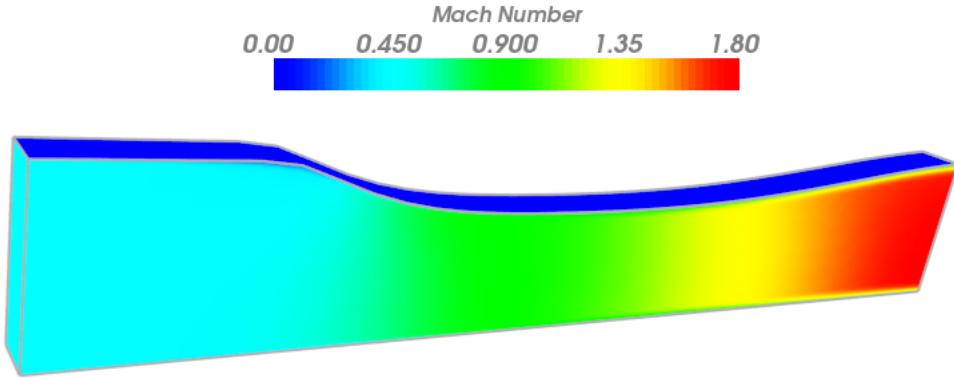


Figure 16.17: Example of a Volume Data Node rendered with a color map scaled to the local Mach number with Flat lighting.

labeled **Border Color**.

The position of the legend can also be controlled using the **XLoc:** and **YLoc:** type-ins. These values represent the relative position of the lower left corner of the legend in the visual window. Both values range from 0 to 1, where (0,0) represents the lower left corner, and (1,1) represents the upper right corner.

The Lighting Model

The **Lighting Model** radio buttons control the technique for modeling light reflecting off the surface of the selected object. By default, the value is set to **Lighting**. When **Lighting** is selected, the surfaces rendered will reflect light in a diffuse manner, such that the user will see the orientation of the surface. When the surface is directly normal to the light source, then the colors will be brightest, while the colors will be darker when the light source is at a large angle to the the surface. The three-dimensional composition will be enhanced. Fig. 16.16 on pg. 290 shows the duct geometry rendered using the **Lighting** option. Selecting **Flat** lighting model will render all colors with the absolute color from the color map. **Flat** light model for the Duct example is shown in Fig. 16.17 on pg. 292.

The Mirroring Frame

Mirroring of the current Filter Node is also controlled within the Mapper Property. Pressing any of the **X**, **Y**, or **Z** check boxes will mirror the current object about the **X**, **Y**, or **Z** axis respectively. The user can mirror about axes singly, or collectively.

16.5 The Contour Node

The Contour Node is a VTK object capable of extracting iso-lines or iso-surfaces from two or three-dimensional data. While the most frequent source of input for the Contour Node

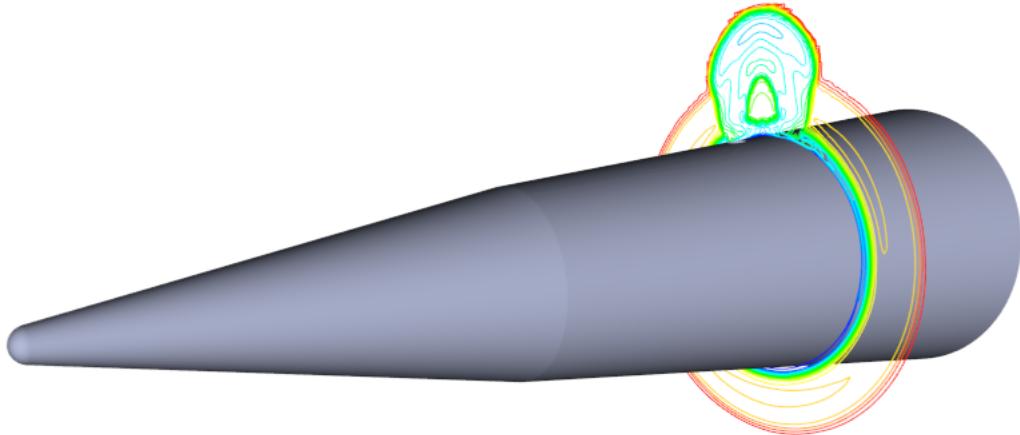


Figure 16.18: Line contours of Mach number extracted from a cutting plane of data about a biconic missile.

is the Data Node (see Sec. 16.4 on pg. 275), any VTK object which produces as output a two or three-dimensional connected data set can be used as input. For example, one can create a cutting plane which extracts a single arbitrary plane of data from a three-dimensional data set. Once created, the cutting plane can then be used as input to the Contour Node. Fig. 16.18 on pg. 293 demonstrates line contours of Mach Number extracted from a cutting plane of the biconic missile visualization tutorial. The Contour Node is created by activating the **Edit** menu from the Visualization Folder, and selecting **New Contour**. A single, unconnected Contour Node Object will be added to the selected Visualization Folder. Once you have created a Contour Node, you must connect it within the pipeline. Select, for example, a previously created Data Node, and drop it onto the new Contour Node. If there are any solution variables associated with the Data Node (see Sec. 16.4.1 on pg. 276), then you should see iso-surfaces or iso-lines of the first variable in the **Active Scalar** list. The construction of three-dimensional iso-surfaces may be a computationally intensive operation. The user may therefore wish to disable the calculation of iso-surfaces while making intermediate changes to the Contour Node by setting the **Render Mode** to “None”. Once all editing changes for the selected Contour Node have been made, the user should return the **Render Mode** to the original setting.

When a Contour Node that has been connected within the pipeline is selected, there are four property tabs shown under the **Solution Visualization** section; **Variables**, **Contours**, **Mapper**, and **Sequence**. The only property associated directly with the Contour Node is the **Contours** property. The other properties are actually associated with the Data Node which is upstream in the pipeline. The upstream properties are included in the property list when the Contour Node is selected in order to facilitate making changes which will impact the Contour Node. However, the user must be aware that making changes to the upstream Data Node may also impact other Filters which use the Data Node.



Figure 16.19: The Contours Property Tab

Note: A quick overview of setting up a contour node is as follows:

- Select the Solution Visualization folder and access the edit menu to select New Contours.
- Select a visualization node that represents the surface or volume to have contours or iso-surfaces displayed. Drag-n-drop the node (most often this is a data node consisting of surfaces or volumes) into the new contours node.
- Adjust the contour settings and render the node to Full for visualization.

16.5.1 The Contours Property

All values associated with the specification of the contour levels in a Contour Node are found under the **Contours** property as seen in Fig. 16.19 on pg. 294.

The Current Scalar Frame

As discussed in Sec. 16.4.1 on pg. 276, the variables required for visualization must be selected in the **Variables** section. Once specified, the Data Node will compute a value for each variable at each point specified in the **Grid Range** section. The **Current Scalar** option menu will be populated with the **Active Scalars** list, (see Sec. 16.4.1 on pg. 278). Selecting a **Current Scalar** will associate the iso-contours with that variable. Once a **Current Scalar** is selected, the **Min** and **Max** entries will be updated to reflect the global bounds of the selected data. These values are provided to assist the user in manual selection of contour levels.

If the value of **Current Scalar** is identical to the **Current Scalar** value found in Sec. 16.4.4 on pg. 290, then changing the value in the contours property will also change the value in the mapper property. If the user wishes the values to be different, they must change the value from the mapper property.

The Contour Levels Frame

There are currently two methods used to specify the contour levels included in a given Contour Node. The method is selected through the top option menu in the frame. When “Uniform Contours” is active, (see Fig. 16.19 on pg. 294) the user will specify the total number of contour levels to include, as well as a minimum and maximum values for the contours. The algorithm will then uniformly distribute contours between the minimum and maximum values. The **Min** and **Max** values may be automatically or manually set in a fashion similar to that of the color map limits in the mapper property. The **Based On:** option menu is populated with all Filter Nodes upstream of the Contour Node. Selecting a Filter Node will associate the minimum and maximum limits of the **Current Scalar** with current Contour Node. For example, if the user wishes to extract contours for a cutting surface with limits set to the local bounds of the cutting surface, and not the global limits of the original Data Node, then the **Based On:** menu should be set to the Cutter Node. By default, the limits will be based upon the root Data Node. If the user directly changes either the **Min** or **Max** value, then the **Based On:** setting will be changed to “User Defined”. The **Min** and **Max** values will not be updated when the actual minimum or maximum values change within the solution when “User Defined” is selected. Instead, it will be up to the user to ensure that the minimum and maximum values are appropriate.

Adjusting the **Number Of Levels** entry will control the total number of contour levels included in the current object. The list to the right of the **Contour Levels** frame will be filled with the values of the levels included in the Contour Node. If the mapper property **Selected Scalar** is the same as the **Current Scalar**, then the list will also show the Color level associated with each contour level. If the mapper property **Selected Scalar** is different from the **Current Scalar**, then the list color mapping will be irrelevant. It is not possible to edit individual contour level values when “Uniform Contours” is selected.

If the user selects “Custom Contours” from the top option menu, then they will be presented with an interface which looks similar to Fig. 16.20 on pg. 296. When “Custom Contours” is selected, the user controls not only the number of contour levels, but also the exact value of each level. Because “Uniform Contours” is the default mode, the “Custom Contours” will be initialized with values from the “Uniform Contours” settings. Therefore the number of levels present, as well as their values will be initialized based upon the “Uniform Contours” settings. Once in “Custom Contours” mode, the user can add individual contour levels by entering a value in the **Custom Level** entry, and then pressing the **Add Custom Level** button. The user may also adjust an existing contour level by selecting the level, and entering a new value in the type-in. Selected entries will be highlighted in the list. Pressing the **Delete Selected Levels** button will delete any selected contour levels.

16.6 The Vector Node

The Vector Node is a VTK object capable of placing a glyph at each discrete data point of the parent VTK Filter. The glyph is a vector drawn from the origin of each data point with

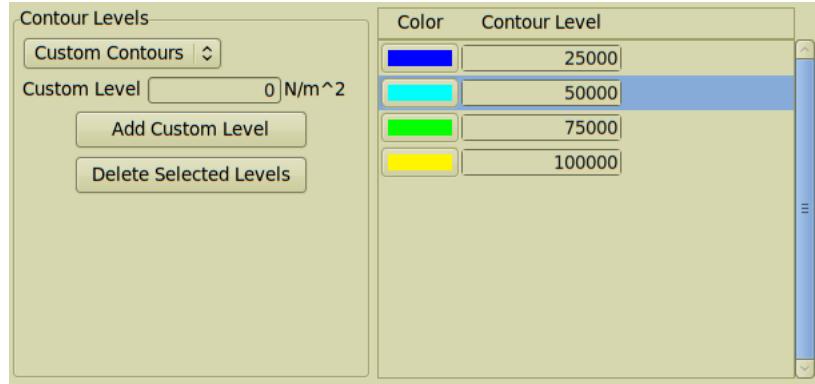


Figure 16.20: The Contours Property Tab when custom contours are selected.

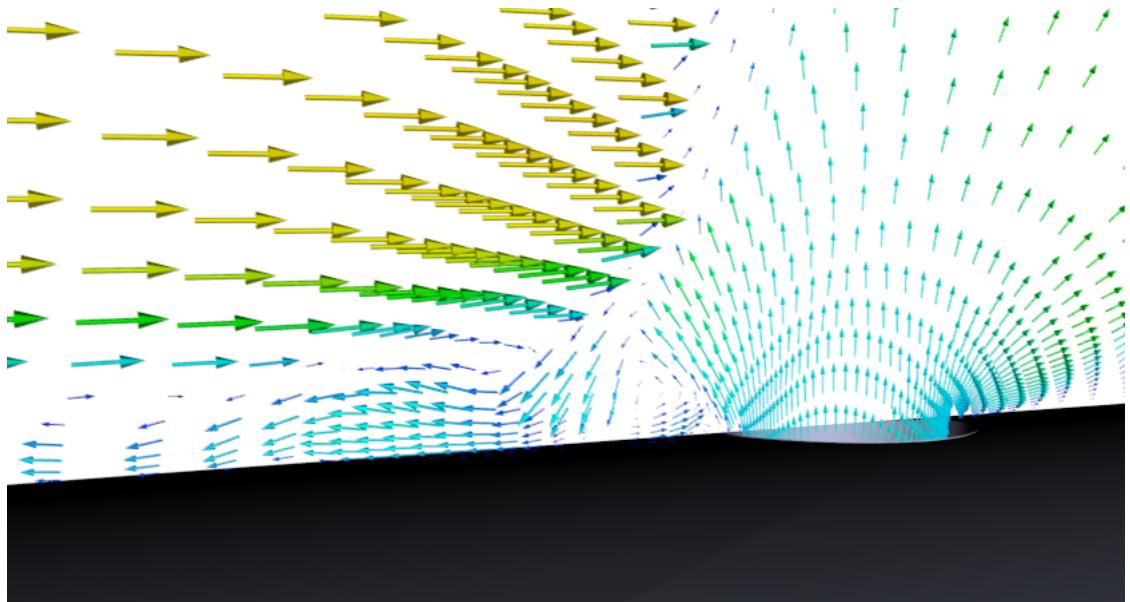


Figure 16.21: Velocity Vectors in the symmetry plane of the visualization example problem.

magnitude and direction taken from a selected vector data function. An arrow tip is placed at the end of the glyph as can be seen in Fig. 16.21 on pg. 296. While the most frequent source of input for the Vector Node will be a Data Node, any Filter which produces discrete points may be used as the parent to a Vector Node.

The Vector Node is created by activating the **Edit** menu from any Visualization Folder, and selecting **New Vectors**. A single, unconnected Vector Node Object will be added to the selected Visualization Folder. Once you have created a Vector Node, you must connect it within the pipeline. Select, for example, a previously created Data Node, and drop it onto the new Vector Node. If there is vector data associated with the parent Data Node (see Sec. 16.4.1 on pg. 279), then you should see vector glyphs of the first vector in the **Active Vectors** list. Using a three-dimensional data set as input to a Vector Node is not

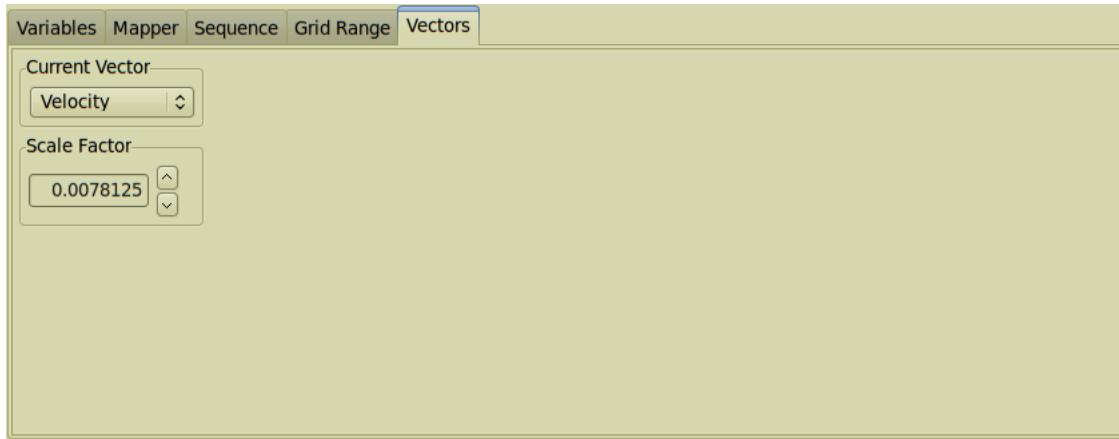


Figure 16.22: The Vectors Property Tab

recommended because glyphs are created at each physical data point.

16.6.1 The Vectors Property

All values associated with the specification of the vector glyphs Vector Node are found under the **Vectors** property as seen in Fig. 16.22 on pg. 297.

Note: A quick overview of setting up a vector node is as follows:

- Select the Solution Visualization folder and access the edit menu to select New Vectors.
- Select a visualization node that represents the surface or volume to have vectors displayed. Drag-n-drop the node into the new vectors node.
- Adjust the vector settings and render the node to Full for visualization.

The Current Vector Frame

As discussed in Sec. 16.4.1 on pg. 276, the variables required for visualization must be selected in the **Variables** section. Once specified, the Data Node will compute a value for each variable at each point specified in the **Grid Range** section. The **Current Vector** option menu will be populated with the **Active Vectors** list, (see Sec. 16.4.1 on pg. 279). Selecting a value in **Current Vector** will associate the vector glyphs with that vector.

The Scale Factor Frame

The vector glyph at each point is drawn with the magnitude scaled based upon the magnitude of the source vector. The magnitude of the vector may have no relation to the relative location of neighboring data points. However, for clarity of viewing vector glyphs, the length of the

glyph should be scaled such that there is little or no overlap between neighboring glyphs. The **Scale Factor** can be used to alter the lengths of the glyphs for the the current Vector Node. Pressing the down arrow will reduce the length of all the glyphs by a factor of two, while pressing the up arrow will increase the length of the glyphs by a factor of two.

16.7 The Cutter Node

The Cutter Node is a VTK object capable extracting an $N - 1$ dimensional surface from an N dimensional surface according to an implicit function supplied to the cutter. That is, a polygonal surface is created corresponding to the implicit function $F(x, y, z) = \text{value}$, where you can specify one or more values used to cut with. For example, the Cutter Node can be used to extract a cutting plane of data from volume data. The Volume Data might be a three-dimensional Data Node. The cutting surface is described by an additional input, an implicit plane definition (see Sec. 16.12 on pg. 311). While the Cutter Node can take as an input any appropriate implicit function, the implicit plane is the only currently implemented function. Fig. 16.23 on pg. 299 shows Mach number flooded contours about a cutting plane extracting data from the Visualization example. The Cutter Node requires two inputs in the pipeline. The first is the data which is to be cut, and the second is the implicit function used to extract the Cut Surface, (see Fig. 16.24 on pg. 299). Note that that there are no control parameters for the Cutter Node. The extracted surface is defined completely by the data source and the implicit function. The Input Folder of the Cutter Node will accept any basic Filter Node. However, the Implicit Function Folder will only accept as input a VTK Node based upon an Implicit Function. Currently the only VTK Node based upon an Implicit Function is the Plane Node, Sec. 16.12 on pg. 311.

Note: A quick overview of setting up a cutter node is as follows:

- Select the Solution Visualization folder and access the edit menu to select New Cutter.
- Select the Solution Visualization folder and access the edit menu to select New Plane Src.
- Create a data node if one does not already exist. Select the data node and drag-n-drop it into the cutter node. It will automatically be added to the Input folder.
- Drag-n-drag the plane src into the cutter node. It will automatically be added to the Implicit Function folder.
- Adjust the mapper settings and render the node to Full for visualization.

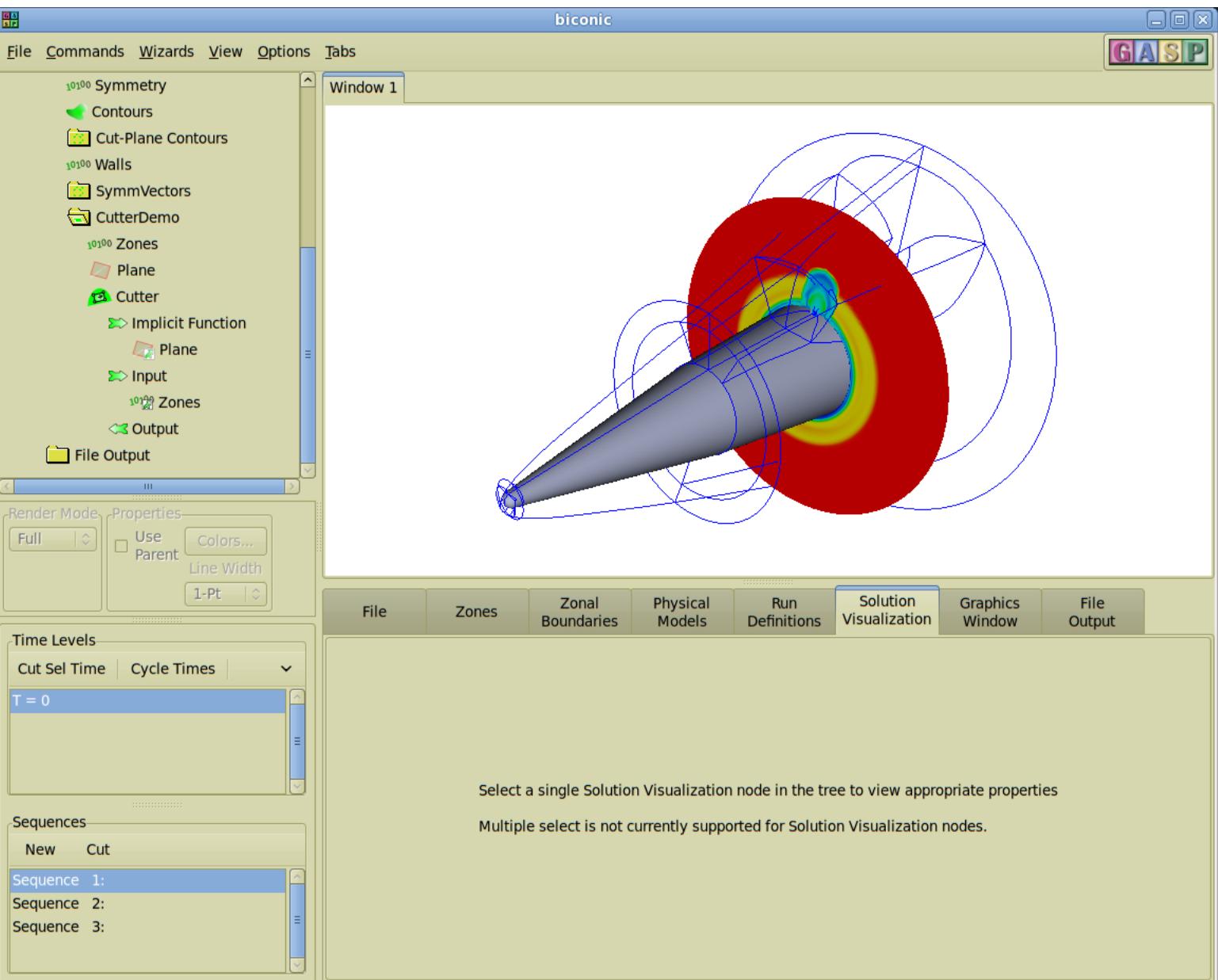


Figure 16.23: Example of a cutting plane extracted from the visualization example problem.

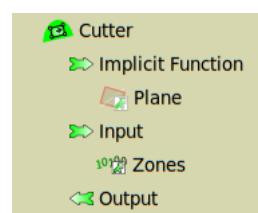


Figure 16.24: The different inputs to the Cutter Node.

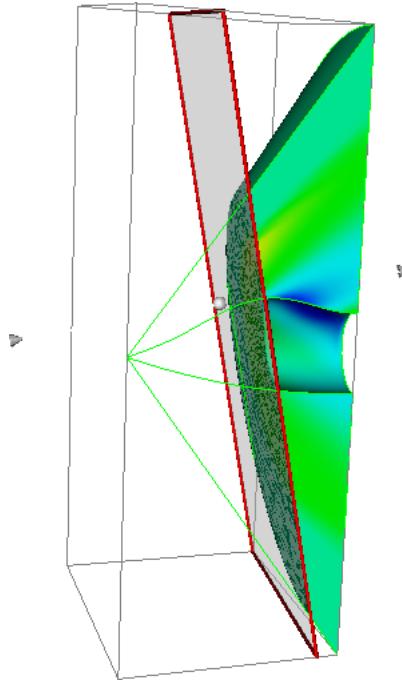


Figure 16.25: Volume dataset of an analytic forebody solution clipped by an arbitrary plane.

16.8 The Clip Volume Node

The Clip Volume Node is a VTK object that clips any type of dataset using either an implicit function, or the input scalar data associated with this dataset. Clipping means that the algorithm actually "cuts" through the cells of the dataset, returning everything inside of the specified implicit function (or greater than the scalar value) including portions of a cell. Fig. 16.25 on pg. 300 demonstrates the extraction of a portion of a volume dataset about an analytic forebody, which was created by clipping based upon an arbitrary implicit plane.

To use this filter, you must decide if you will be clipping with an implicit function, or whether you will be using the input scalar data. If you want to clip with an implicit function, you must define an implicit function and associate it with the the current Node. If you want to clip based upon one of the scalar data components, then you must specify which variable to will be used as a clipping function, and a threshold value for clipping.



Figure 16.26: The Clip Volume Property Tab

Note: A quick overview of setting up a clip volume node is as follows:

- Select the Solution Visualization folder and access the edit menu to select New Clip Volume.
- Select the Solution Visualization folder and access the edit menu to select New Plane Src.
- Create a data node if one does not already exist. Select the data node and drag-n-drop it into the clip volume node. It will automatically be added to the Input folder.
- Drag-n-drag the plane src into the clip volume node. It will automatically be added to the Implicit Function folder.
- Adjust the clip volume settings and render the node to Full for visualization.

16.8.1 The Clip Volume Property

All settings associated with the specification of a clipping function in a Clip Volume Node are found under the **Clip Volume** property as seen in Fig. 16.26 on pg. 301.

The Orientation of Clipping Frame

The clipping function keeps data satisfying a boolean operator. If an implicit function is used, then only data which is less than that implicit function (*e.g.*, to one side of an implicit plane) will be included in the output. Selecting the **Invert Clipping** check box will cause the function to return the opposite data for the clipping function (*e.g.*, the opposite side of the implicit plane).

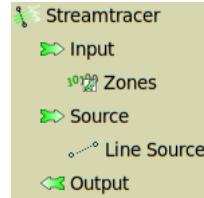


Figure 16.27: The StreamTracer Node and its children.

The Clip To Scalar Frame

If the **Clip To Scalar** check box is selected, then the Clip Volume will use a scalar function for clipping instead of an implicit function. As discussed in Sec. 16.4.1 on pg. 276, the variables required for visualization must be selected in the **Variables** section. Once specified, the Data Node will compute a value for each variable at each point specified in the **Grid Range** section. The **Current Scalar** option menu will be populated with the **Active Scalars** list, (see Sec. 16.4.1 on pg. 278). Selecting a **Current Scalar** will cause the clip volume to operate based upon that scalar variable. The **Threshold** value is used to define the boolean clipping function. For example, if Mach Number is the current variable, and the threshold value is 1.3, then only volumes whose Mach number is less than 1.3 will be included in the output. If the **Invert Clipping** box is selected, then only volumes whose Mach number is greater than 1.3 will be included in the output.

16.9 The Streamtracer Node

The Streamtracer Node is a VTK object that integrates a vector field to generate streamlines. The integration is performed using the provided integrator. The default is second order Runge-Kutta. The Streamtracer node generates polylines as output and requires two inputs in the pipeline. The first provides the data through which to integrate, and the second provides a collection of data points with which to seed each stream line (see Fig. 16.27 on pg. 302). Identifying the intended destination of a VTK Filter node being dropped onto the main node may be impossible. For example, a Data Node may be intended to be dropped as either the domain through which to integrate, or as a collection of seed points. Because of this possible confusion, the user will need to drop the upstream element of the pipeline directly onto either the **Input** folder, or the **Source** folder. The Line Source Node is intended to be solely used as a generator of seed locations for the Streamtracer Node. However, any VTK Filter which creates data points can be used as a streamtracer source. In addition, the **Source** folder will accept more than one input. If more than one input is provided to the **Source** folder, then the resulting data points of the individual filters will be concatenated.

The Streamtracer node can integrate both forward and backward. The length of the streamline is controlled by specifying a maximum integration time (the time each particle would have traveled if flow were steady). Otherwise, the integration terminates after exiting the dataset or if the particle speed is reduced to a value less than the terminal speed or when

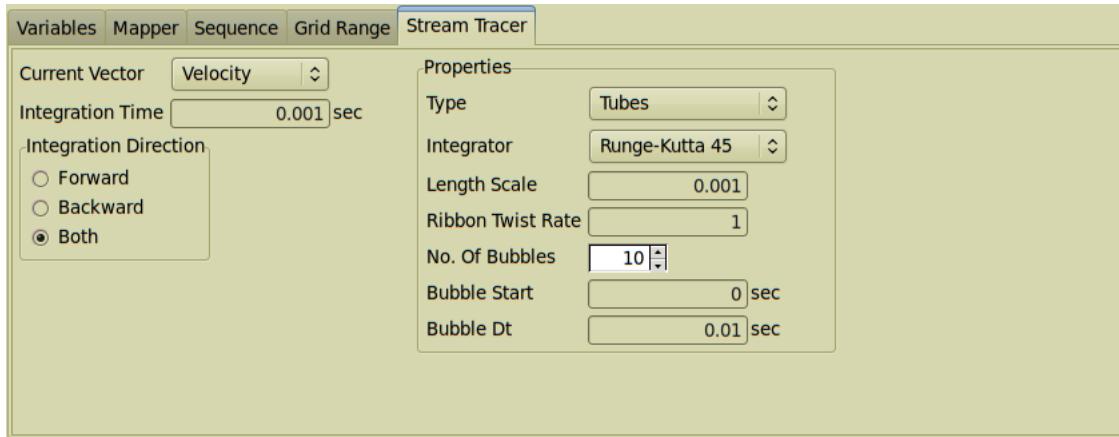


Figure 16.28: The Stream Tracer Property Tab

a maximum number of steps is reached.

Note: A quick overview of setting up the stream tracer is as follows:

- Select the Solution Visualization folder and access the edit menu to select New Streamtracer.
- Drag-n-drop a data node into the stream tracer Input folder. The data node should represent a volume.
- Drag-n-drop a node that will be used to locate the starting points into the stream tracer Source folder. This can be a line src or a data node.
- Adjust the stream tracer settings and render the node to Full for visualization.

16.9.1 The Streamtracer Property

All values associated with the specification of the stream tracer in a Streamtracer Node are found under the **Stream Tracer** property as seen in Fig. 16.28 on pg. 303.

The Current Vector Frame

As discussed in Sec. 16.4.1 on pg. 279, the variables required for visualization must be selected in the **Variables** section. In particular, the Streamtracer Node requires a vector field representing a velocity. Once specified, the Data Node will compute a value for each variable at each point specified in the **Grid Range** section. The **Current Vector** option menu will be populated with the **Active Vectors** list, (see Sec. 16.4.1 on pg. 278). Selecting a **Current Vector** will associate the streamlines with that vector field.

The Integration Time Frame

A streamtracer assumes that the associated Vector field represents a time rate of change of position or velocity. The streamtracer integrates the velocity vector through space as a function of time. The **Integration Time** represents the total amount of time in the integration. The limits of integration are from 0 to **Integration Time** where the points are located at the seed locations at $t = 0$. The integration time should be set to a reasonable value, based upon the local dimensions and velocities. For a typical CFD simulation, an appropriate integration time would be measured in milliseconds, instead of seconds or minutes.

The Integration Direction

The streamtracer algorithm includes the ability to integrate both forward and backward in time from a given seed location. The **Integration Direction** radio box allows the user to select integration in the “Forward”, or the “Backward” direction, as well as in “Both” directions. Note that in all cases, the starting integration locations are defined by the seed points provided as an input to the VTK Pipeline.

Streamtracer Type

Once the coordinates of the streamlines have been determined, by integrating through the data set, the Stream Tracer Node must then render the streamlines to the screen. There are a number of ways to represent the streamline data, controlled through the **Type** option menu. If **Type** is set to “Lines”, then the streamlines are simply rendered as lines, as shown in Fig. 16.29 on pg. 305.

If **Type** is set to “Ribbons”, then the streamlines are rendered as an oriented ribbon along the streamlines as shown in Fig. 16.30 on pg. 305. The width of the ribbon is a constant, but can be made smaller or larger by adjusting the value of **Length Scale**. The ribbon will also twist according to the vorticity in the flow. The **Ribbon Twist Rate** can be used to control the mapping of vorticity to the twist angle. Currently, there is no means to automatically adjust the relative values of **Length Scale** and **Ribbon Twist Rate** in order to achieve a pleasing visualization. The user is responsible for selecting reasonable values for these parameters.

If **Type** is set to “Tubes”, then the streamlines are rendered as tubes with a round cross section as shown in Fig. 16.31 on pg. 306. The radius of the tube will be constant along the length of the streamline. The magnitude of the radius can be scaled by changing the value of **Length Scale**.

If **Type** is set to “Bubbles”, then discrete points along each streamline are rendered as bubbles or spheres as shown in Fig. 16.32 on pg. 307. When the streamline is created, the integration time is recorded along each streamline. Points at uniform increments in time can be extracted from the streamline. The **Bubble Dt** value represents the uniform increment in time at which to display a bubble. The **No. Of Bubbles** represents the number bubbles to generate along each streamline. The **Bubble Start** represents the starting location of the

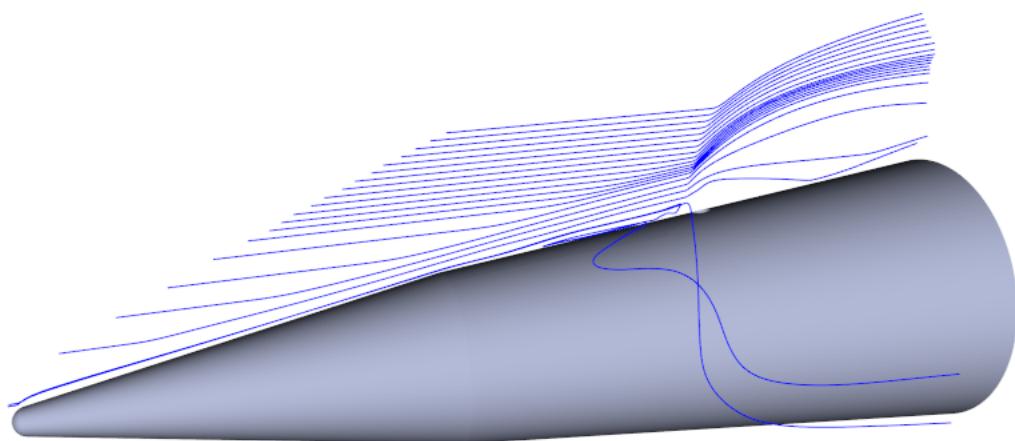


Figure 16.29: The Stream Tracer stream lines with **Type** set to “Lines”

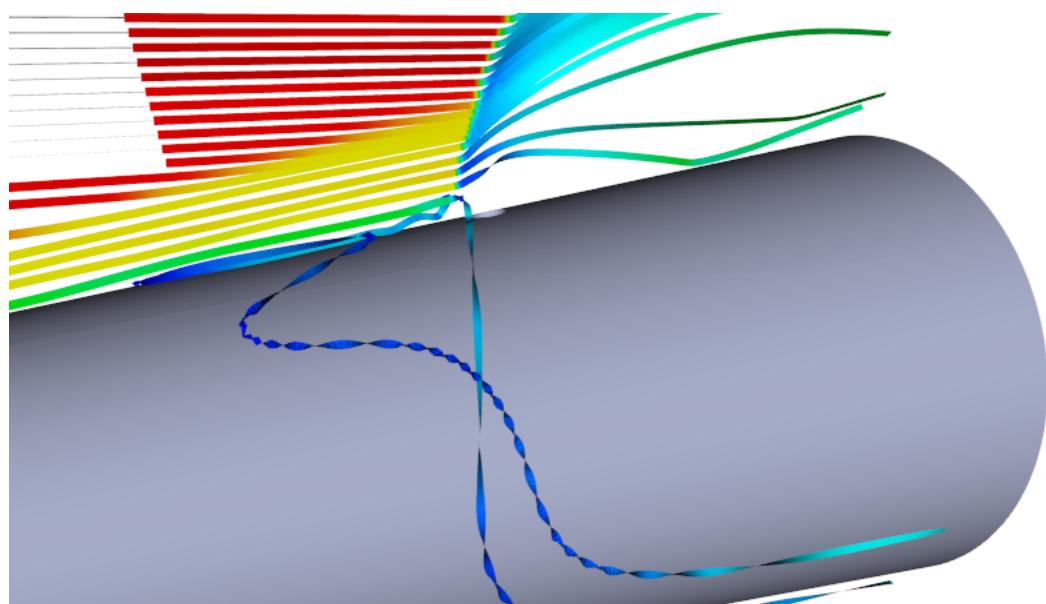


Figure 16.30: The Stream Tracer stream lines with **Type** set to “Ribbons”

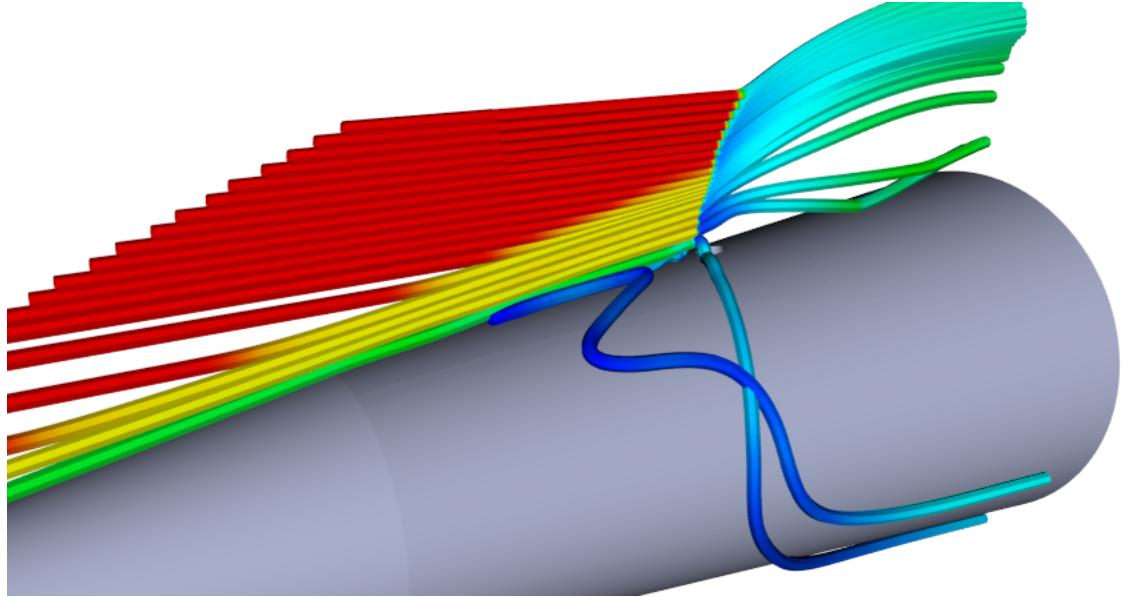


Figure 16.31: The Stream Tracer stream lines with **Type** set to “Tubes”

first bubble. The **Bubble Start** can be used in conjunction with Movies to animate bubbles along a streamline.

16.10 The Displacement Node

The Displacement Node is a VTK object that displays the displaced grid of the parent VTK Filter. The displaced grid is computed from the parent VTK filter grid with each point in the grid moved by the magnitude and direction of the displacement data function. The Displacement Node is used in conjunction with the material stress solver to show the material deformation. An example showing the original grid in blue and the deformed grid in red can be seen in Fig. 16.33 on pg. 307. Note that the actual displacements are typically very small, so the displacements have been scaled for this case by the user to appear more visible. While the most frequent source of input for the Displacement Node will be a Data Node, any Filter which produces discrete points may be used as the parent to a Displacement Node.

The Displacement Node is created by activating the **Edit** menu from any Visualization Folder, and selecting **New Displacement**. A single, unconnected Displacement Node object will be added to the selected Visualization Folder. Once you have created a Displacement Node, you must connect it within the pipeline. Select, for example, a previously created Data Node, and drop it onto the new Displacement Node. If there is displacement data associated with the parent Data Node (see Sec. 16.4.1 on pg. 279), then you should see displacement in the **Active Vectors** list.

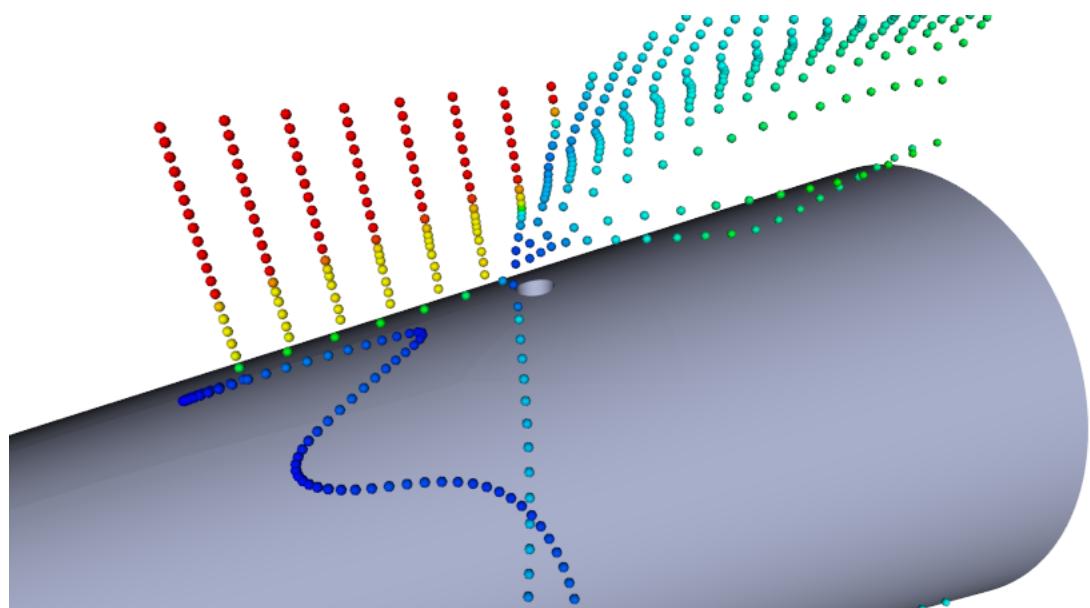


Figure 16.32: The Stream Tracer stream lines with **Type** set to “Bubbles”

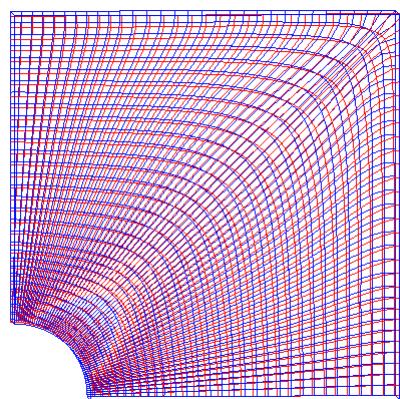


Figure 16.33: Displacements in the symmetry plane of the visualization example problem.

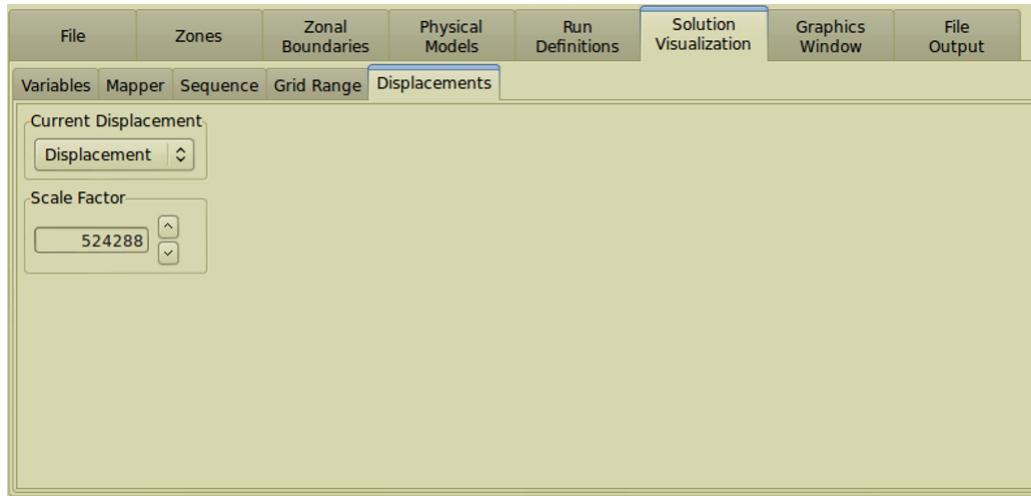


Figure 16.34: The Displacements Property Tab

Note: A quick overview of setting up a displacement node is as follows:

- Select the Solution Visualization folder and access the edit menu to select New Displacement.
- Create a material stress data node if one does not already exist. Select the data node and drag-n-drop it into the displacement node. Recall that only solutions from the material stress physical model can be used with this visualization operator.
- Adjust the displacement settings and render the node to Full for visualization.

16.10.1 The Displacements Property

All values associated with the specification of the Displacement Node are found under the **Displacements** property as seen in Fig. 16.34 on pg. 308.

The Current Displacement Frame

As discussed in Sec. 16.4.1 on pg. 276, the variables required for visualization must be selected in the **Variables** section. Once specified, the Data Node will compute a value for each variable at each point specified in the **Grid Range** section. The **Current Displacement** option menu will be populated with the **Active Vectors** list, (see Sec. 16.4.1 on pg. 279). Only the **Displacement** variable should be selected for the **Current Displacement**.

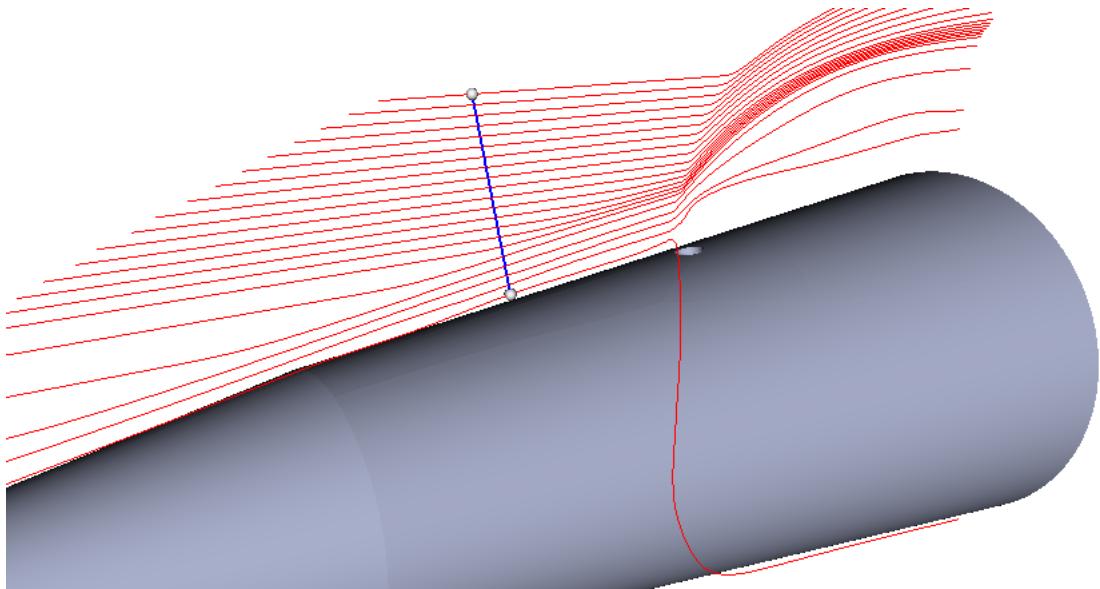


Figure 16.35: A Line Source Node being used as a seed location for a Streamtracer Node.

The **Scale Factor** Frame

The displacement at each point is drawn with the magnitude scaled based upon the magnitude of the source displacement. The magnitude of the displacement are typically very small and not visible without scaling. The **Scale Factor** can be used to alter the magnitude of the displacements for the current Displacement Node. Pressing the down arrow will decrease the scaling of the displacements by a factor of two, while pressing the up arrow will increase the scaling of the displacements by a factor of two.

16.11 The Line Source Node

The Line Source Node is a support filter which may be used in conjunction with the Stream Tracer Node. The Line Source Node is a source of data, and does not need to be connected upstream in the pipeline. There is no child input folder. The Line Source Node is used mainly as a source of seed points to the Stream Tracer Node and is defined by its two end points. The seed points will be uniformly distributed along the line from the start point to the end point. The Line Source Node is an interactive widget, and is only visible when the node is selected. When selected the user may interactively edit the source. If the user picks an end point, then the end point can be translated. If the user picks the line itself, then the entire line will be translated. To avoid accidentally moving the line graphically, the user may lock the line in place using the **Lock Points** option. Fig. 16.35 on pg. 309 shows the Line Source when it is selected.

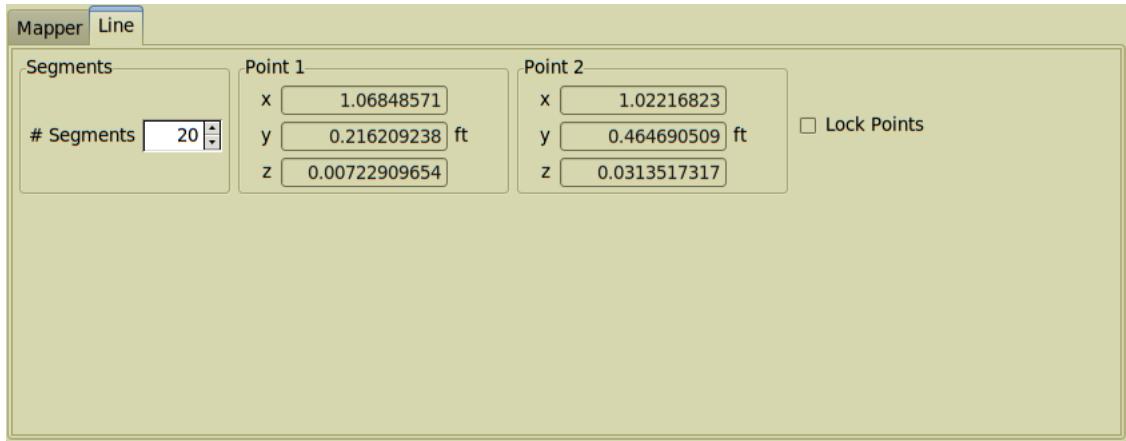


Figure 16.36: The Line Source Property Tab

16.11.1 The Line Source Property

All values associated with the specification of the Line Source Node are found under the **Line Source** property as seen in Fig. 16.36 on pg. 310.

The # Segments Entry

The purpose of the Line Source Property is to provide seed locations for the Streamtracer Node. The **# Segments** entry specifies the number of seed points which will be created between the starting and ending points of the current Line Source.

The Point 1 Coordinates

The **Point 1** coordinates display the exact coordinates of the first end point of the current Line Source. The values will be updated if the user edits the end point graphically. Otherwise, the exact desired coordinates may be input directly into this entry.

The Point 2 Coordinates

The **Point 2** coordinates display the exact coordinates of the second end point of the current Line Source. The values will be updated if the user edits the end point graphically. Otherwise, the exact desired coordinates may be input directly into this entry.

The Lock Points Option

If the **Lock Points** check box is enabled, then the user will be unable to graphically change the values of the end points of the current Line Source. Enabling this feature can prevent inadvertently changing the Line Source location graphically.

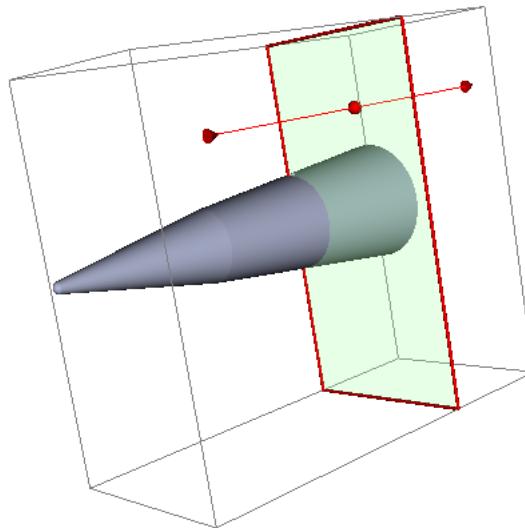


Figure 16.37: A Plane Source Node as displayed when selected.

16.12 The Plane Source Node

The Plane Source Node is a support filter which may be used in conjunction with the Cutter Node. The Plane Node is a source of data, and does not need to be connected upstream in the pipeline. There is no child Input folder. The Plane Node is used exclusively as a source for the implicit function required by the Cutter Node. The Plane defines an infinite plane in physical space. The plane is defined by a single point lying on the surface of the plane, and a vector normal to the surface of the plane. While the visual representation of the plane describes finite limits to the plane, the actual plane as is used by the Cutter Node has infinite bounds.

The Plane Source Node is an interactive widget, and is only visible when the node is rendered and selected. When selected the user may interactively edit the source. If the user picks on the red outline of the plane, then the plane will be translated in the direction of the unit normal. If the user selects on the unit normal vector, then the direction of the normal can be changed. If the bounding box is selected, then it will be translated, along with the location of plane itself. Fig. 16.37 on pg. 311 shows the Plane Source Node when it is selected.

16.12.1 The Plane Property

All values associated with the specification of the Plane Source Node are found under the **Plane** property as seen in Fig. 16.38 on pg. 312.

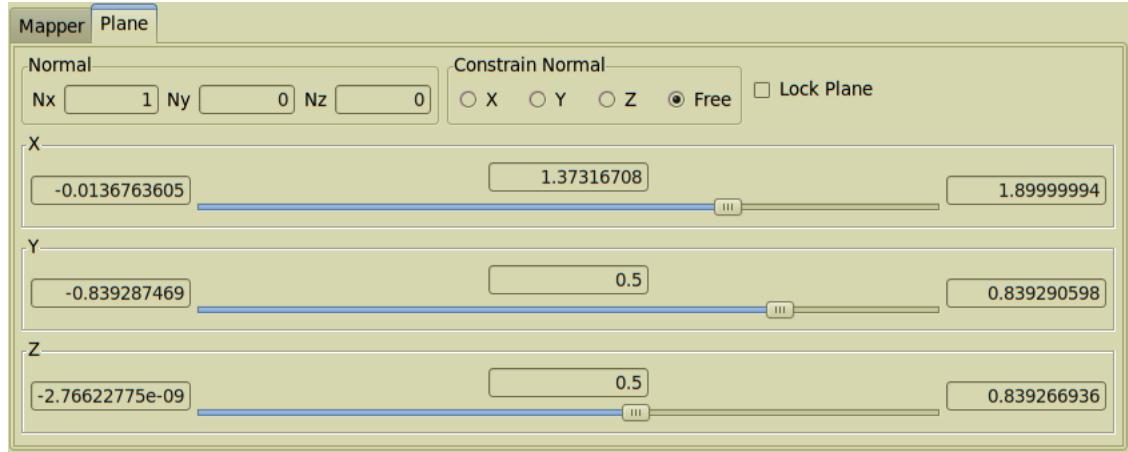


Figure 16.38: The Plane Property Tab

The Normal Frame

The unit normal for the Plane Node is displayed or modified through the **Normal** Frame. The **Nx**, **, and **Nz** entries represent the *X*, *Y*, and *Z* components of the plane unit normal respectively. The magnitude of the unit normal need not be normalized.**

The Constrain Normal Frame

The user may constrain the unit normal to lie in either the *X*, *Y* or *Z* global axis if they so choose. When the constraint direction is not “Free”, the user will be unable to change the direction of the unit normal either interactively, or through the **Normal** Frame.

The Lock Plane Check Box

The **Lock Plane** check box may be activated in order to prevent inadvertent modification of the plane geometry. Because the plane can be modified from the viewing window, disabling the interactive editing of the geometry can be advantageous.

The X, Y, and Z, Range Widgets

The **X**, **Y**, and **Z**, Range Widgets allow quantitative manipulation of the plane location. Interactive manipulation of the plane can be imprecise. Directly entering the values of the coordinates allows precise control of the plane location.

16.13 The Transformation Node

The Transformation Node is a support filter which takes any number geometric filters as input, and applies a geometric transformation matrix to the output of those geometric filters.

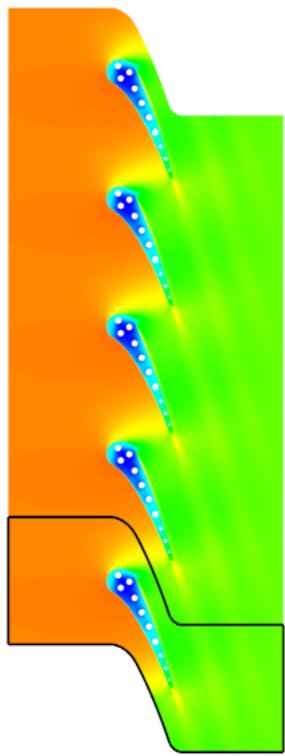


Figure 16.39: A Cascade Vane translated in the vertical direction and reproduced four times through the Transformation Node.

The Transformation Node is used primarily for reproducing periodic solutions, whether they be from a problem such as a cascade of blades, or from a section of a problem which takes advantage of symmetry about an axis. For example, a turbo-machinery problem which models a single turbine blade. The Transformation Node can also apply a repeat factor so that the transformation can be applied recursively any number of times. Fig. 16.39 on pg. 313 demonstrates the use of the Transformation Node to reproduce a Cascade Vane. Note that the transformation node will accept any number of inputs. All Filters connected to the Transformation Node will have that transformation applied.

Note: A quick overview of setting up a transform node is as follows:

- Select the Solution Visualization folder and access the edit menu to select New Transformation.
- Select an existing visualization node and drag-n-drop it into the transformation node. It will automatically be added to the Input folder.
- Adjust the transformation settings and render the node to Full for visualization.



Figure 16.40: The Transformation Property Tab

16.13.1 The Transformation Property

All values associated with the specification of the Transformation Node are found under the **Transformation** property as seen in Fig. 16.40 on pg. 314.

The Repeat Number Frame

The transformation matrix can be applied recursively any number of times. The **# Segments** input controls the number of times the transformation matrix is applied.

The Reset Transformation Button

Pressing the **Reset Transformation** will set the transformation matrix back to the identity matrix. This operation will remove all transforms.

The Scale Transformation... Button

Pressing **Scale Transformation...** button will present an input dialog allowing the user to specify scale factors to be applied to the current transformation as seen in Fig. 16.41 on pg. 315. **X0**, **Y0**, and **Z0** specify the point about which to apply the scaling parameters. **Scale X**, **Scale Y**, and **Scale Z** allow the specification of scale factors independently in the three component directions. The **Scale Units** unit system specifies the unit system for the specification of the scaling parameters. Pressing the **Scale** button will apply the current scaling transform to the existing transformation matrix and close the dialog. Pressing the **Cancel** button will cancel the operation and close the dialog.

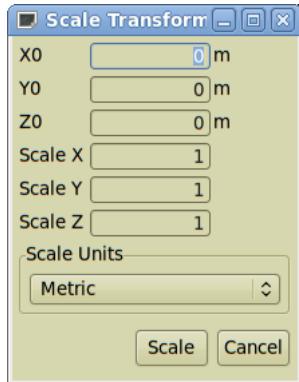


Figure 16.41: The Scale Transformation Dialog

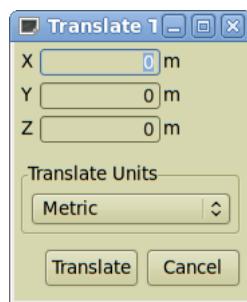


Figure 16.42: The Translate Transformation Dialog

The **Translate Transformation...** Button

Pressing **Translate Transformation...** button will present an input dialog allowing the user to specify translation factors to be applied to the current transformation as seen in Fig. 16.42 on pg. 315. X, Y, and Z values specify the translation vector. The **Translate Units** unit system specifies the unit system for the specification of the translation parameters. Pressing the **Translate** button will apply the current translation transform to the existing transformation matrix and close the dialog. Pressing the **Cancel** button will cancel the operation and close the dialog.

The **Rotate Transformation...** Button

Pressing **Rotate Transformation...** button will present an input dialog allowing the user to specify an arbitrary rotation matrix about a specified point to be applied to the current transformation as seen in Fig. 16.43 on pg. 316. X0, Y0, and Z0 values specify the point about which to apply the rotation. The **Axis X**, **Axis Y**, and **Axis Z** values specify a normal vector about which to apply the rotation. The **Angle** specifies the angle of the rotation. **Rotation Units** unit system specifies the unit system for the specification of the rotation parameters. Pressing the **Rotate** button will apply the current rotation transform to the

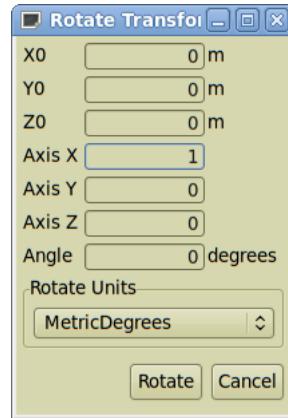


Figure 16.43: The Rotation Transformation Dialog

existing transformation matrix and close the dialog. Pressing the **Cancel** button will cancel the operation and close the dialog.

16.14 The Text Node

The Text Node is a support filter which may be used to place Text in the display area. When a Text Node is selected, the user may interactively move and resize the text within the viewing area. The user may also change the font characteristics, and choose whether or not to display a bounding box. The Text Node is rendered as a two-dimensional overlay in the viewing window. The text will always be displayed on top of any three-dimensional object.

16.14.1 The Text Property

All values associated with the specification of the Text Node are found under the **Text** property as seen in Fig. 16.44 on pg. 317.

The **Text To Display** Entry

The actual text to be displayed for the current Text Node is entered in the **Text To Display** entry.

The **Font** Option Menu

The user can currently select from one of three type fonts; “Arial”, “Courier”, and “Times”.

The **Italic** Check Box

Selecting the **Italic** check box will toggle the italic version of the selected typeface.

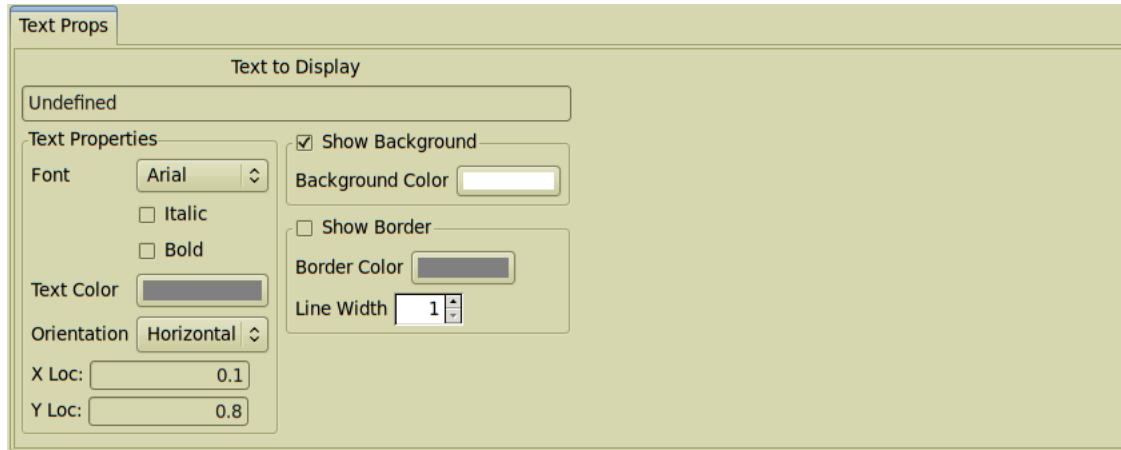


Figure 16.44: The Text Property Tab

The **Bold** Check Box

Selecting the **Bold** check box will toggle the bold version of the selected typeface.

The **Font Color**

Pressing the **Color** button will display a color selection dialog. From this dialog, the user can change the displayed font color of the current Text.

The **X Loc:** and **Y Loc:** typeins

The position of the text can also be controlled using the **X Loc:** and **Y Loc:** type-ins. These values represent the relative position of the lower left corner of the legend in the visual window. Both values range from 0 to 1, where (0,0) represents the lower left corner, and (1,1) represents the upper right corner.

The **Show Background** Frame

An opaque background can be rendered for the text node by selecting the **Show Background** check box. The background color can be altered by pressing the color button labeled **Background Color**.

The **Show Border** Frame

An outline of the bounding box defining the text node can be drawn by selecting the **Show Border** check box. The color of the outline can be altered by pressing the color button labeled **Border Color**. The line width of the border can also be adjusted.



Figure 16.45: The Logo Property Tab

16.15 The Logo Node

The Logo Node is a support filter which may be used to place an arbitrary image, such as a logo, in the display area. When a Logo Node is selected, the user may interactively move and resize the image within the viewing area. The Logo Node is rendered as a two-dimensional overlay in the viewing window and will always be displayed on top of any three-dimensional scene. Transparency can be added to the Logo Node through the alpha value found in the Color Property for the Logo Node.

16.15.1 The Logo Property

All values associated with the specification of the Logo Node are found under the **Logo** property as seen in Fig. 16.45 on pg. 318.

The Logo File (png)

The file containing the PNG image to be displayed for the current Logo Node is entered in the **Logo file (png)** entry. Note that the image file is not stored in the GASP input files. The user must ensure that the image file remains available to the GASP executable. The file name can be a local path or a complete path.

The X Loc: and Y Loc: typeins

The position of the logo can also be controlled using the **X Loc:** and **Y Loc:** type-ins. These values represent the relative position of the lower left corner of the legend in the visual window. Both values range from 0 to 1, where (0,0) represents the lower left corner, and (1,1) represents the upper right corner.

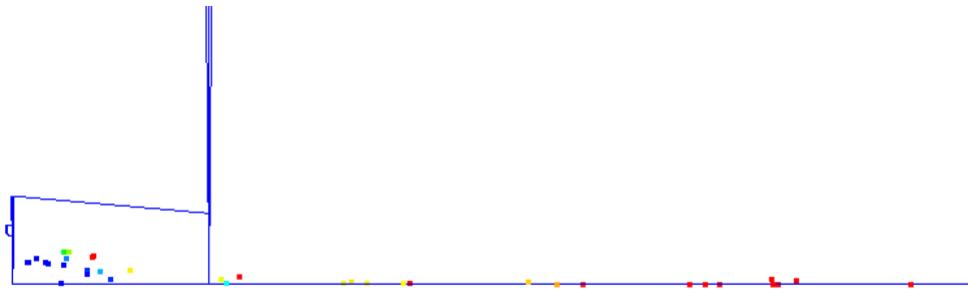


Figure 16.46: Lagrange particles colored by particle temperature in the symmetry plane of the visualization example problem.

16.16 The Lagrange Src Node

The Lagrange Src Node is a VTK object that displays the lagrange particle data of a given set of zones. Unlike other VTK nodes, the Lagrange Src Node only takes a Zone Node or Zone Group Node as input. It currently does not accept other VTK nodes as input. The Lagrange Src Node is used in conjunction with the lagrange spray solver to show particle data. An example showing lagrange particles that have been colored by particle temperature can be seen in Fig. 16.46 on pg. 319.

The Lagrange Src Node is created by activating the **Edit** menu from any Visualization Folder, and selecting **New Lagrange Src**. A single, unconnected Lagrange Src Node Object will be added to the selected Visualization Folder. Once you have created a Lagrange Src Node, you must connect it within the pipeline. This can only be done by selecting a Zone Node or Zone Group Node and dropping it onto the new Lagrange Src Node. If the Zone Node dropped into the Lagrange Src Node has a Lagrange solution, then the particles for that solution will be displayed if the **Render Mode** is set to either **Wire** or **Full**.

The particles can be colored to scalar values in the same manner as **Contours**. In order to color the particles to particle data, a particle variable such as particle temperature, must be selected in the **Variables** section. In order to gain access to the particle variables, the Physical model selected in the **Variables** tab must be a Lagrange Physical model as shown in Fig. 16.47 on pg. 320.

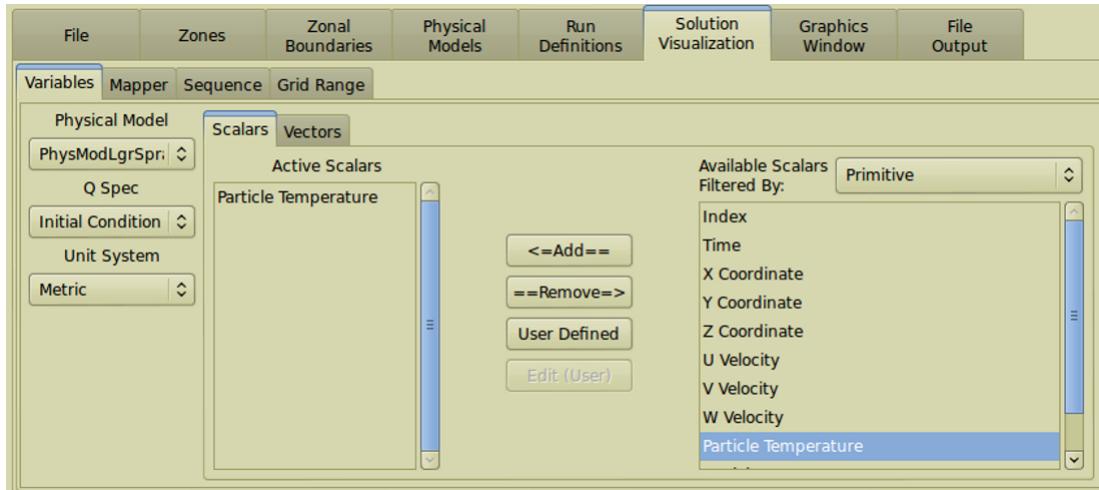


Figure 16.47: Variable tab with Lagrange Physical Model and variables.

Note: A quick overview of setting up a Lagrange source node is as follows:

- Select the Solution Visualization folder and access the edit menu to select New Lagrange Src.
- Select a zone node or zone folder directly and drag-n-drop it into the Lagrange Src node. Note that this is a different behavior than other VTK operators.
- Select the Lagrange Src node and go to the Variables section. Add a scalar variable that will be used to color the particle values. Set the correct scalar in the Mapper section.
- Render the node to Full for visualization.

Chapter 17

The File-Output Section

17.1 Introduction

The *GASPer* software distinguishes between post processing solutions for analysis within the GUI and analysis external to the *GASPer* software. The **File Output** section of the software is intended for manipulating solutions into meaningful output intended for use in external programs, while the **Solution Visualization** section is intended for analysis within the GUI. The process of creating the file output can take place either within the GUI or during flow solver execution. However, all File Output *parameters* must be edited from within the GUI. Within the context of *GASPer*, **File Output** includes:

- Output of volumetric data suitable for input to a third-party post processor (*e.g.*, Plot3D, Fieldview, Ensight, Tecplot, VisIt, and ParaView).
- Creation of tabulated line output.
- Output of integrated quantities (*e.g.*, forces, moments, mass flow).
- Output of minimum or maximum parameters and location

As part of the set up of post processing entities, the user must determine four attributes of a post processing entity.

17.1.1 What variables to output?

When operating on a Eulerian physical model, such as the Navier Stokes model, *GASPer* solves a discrete system of governing equations for a set of primitive variables at each cell center of a grid. For Navier Stokes, these primitive variables will include specie densities, velocity components, pressure, and possibly turbulence quantities. From the primitive variables, a wide range of different quantities may be computed such as Mach number, temperature, skin friction, and mass flow. Each post-processing entity may be a function of multiple primitive

variables and the grid. One step in defining output entities is selecting the desired quantities for output. Depending upon the output type or format (*e.g.*, Tecplot or Integrated Quantity), different output quantities may or may not be available. For example, the Mach number is available for tabular output but is not appropriate for integrated output. Details relating to the specification of which variables to export can be found in Sec. 17.4 on pg. 339

17.1.2 Where in the domain to output?

Because *GAS*Pex is fundamentally a finite-volume code, the solution file contains volume-averaged data for the primitive variables. In a discrete sense, data is available at the centroid of each control volume. However, grid information is specified at the corners, or nodes of each control volume. A second step of the output-specification process is to identify where in the domain a solution is output. *GAS*Pex can either interpolate the grid coordinates to the cell centers and print information at cell centers, or it can interpolate the cell-averaged data to the nodes of the grid. A third interpolation location is to cell faces. Face interpolation will be selected when integrated quantities are desired. Finally, the user can control which range of grid points should be included in the output.

17.1.3 When to output the data?

*GAS*Pex can produce output at different times during the solution process. The user specifies when output is to occur. Post processing can occur immediately from within the GUI, or during scheduled times in the solution process. For example, output can be created after a user-specified number of cycles during solver execution. This output schedule is helpful for monitoring solution convergence. The output can also occur periodically during a time-dependent problem. Finally, output can be scheduled to occur at the end of a flow-solver run. The output at the end of a run helps to eliminate the need to open the GUI just to extract output. As an added convenient, the user can also set up output to be performed from the command line (batch mode).

17.1.4 In what format should the data be output?

The last attribute for output is the format of the post-processed data. The format can be for straight quantitative analysis, or for further post processing by a third-party package. If a third-party package is selected (such as Tecplot), then a suitable format may be specified.

17.2 Output Nodes

Post-processing information is organized hierarchically within the tree view, under the folder **File Output**, (see Fig. 17.1 on pg. 324). There are three types of nodes which reside under the **File Output** folder:

1. Output folders,
2. Output nodes,
3. Group folders

Each Output folder identifies a single export file. For example, a folder can organize the surface integration for tracking lift and drag on an vehicle. Each output folder maintains unique settings for output variables and output format. Each output folder will contain one or more output nodes. Each output node is associated with a specific zone. Once an output node is created, changing the associated zone is not possible. A new node must be created. Each output node maintains grid range properties which define how and where to collocate geometry and solution data for the given associated zone. Finally, there may be group folders in the file output hierarchy. Group folders are available simply for the purpose of organizing output folders and nodes. Creating group folders is done by selecting the **New** button from within the tree-view edit pop-up, (see Sec. 7.4.1 on pg. 85).

17.2.1 Creating Output Nodes

The primary method for creating file output folders and nodes is to “drag and drop” a selected zone node or a selected surface node into a file output group folder. When a zone node is dropped into a group folder (either the “File Output” folder, or a group folder within the “File Output” hierarchy), a new output folder is created containing a single output node linked to the given zone node. Many properties associated with the zone node are copied to the output node. In addition, the grid range limits of the output node are set to the maximum dimensions of the selected zone. Likewise, if a surface node is dropped into a group folder, a new output folder is created with the properties of the contained output node set to those of the surface node. The zone for the new output node is set to that of the surface node, and the grid limits are set to correspond to the limits of the surface.

If a zone folder (or a surface folder) is dropped into an output group folder, then a new output folder is created with a folder and node hierarchy corresponding to the associated zone or surface folder. For example, an output folder representing the symmetry plane of a problem can easily be created by drag dropping the surface folder which has been set up to describe the symmetry boundary conditions into the “File Output” group folder.

New output nodes and folders can also be created through the copy/paste mechanism found in the edit menu, (see Sec. 7.4.1 on pg. 85). If the user selects a range of output nodes and presses the **Copy** button, the selected output nodes are copied to the clip board. Selecting an output folder and pressing the **Paste** button will copy the output nodes stored in the clip board to the selected output folder.

Output nodes can be added to existing Output folders by drag dropping zone or surface nodes directly to that folder.

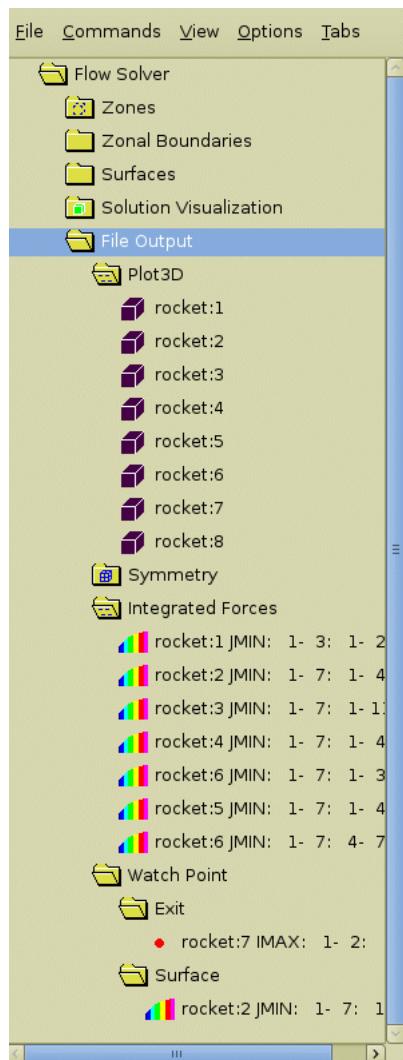


Figure 17.1: The Tree View with example Output Nodes.

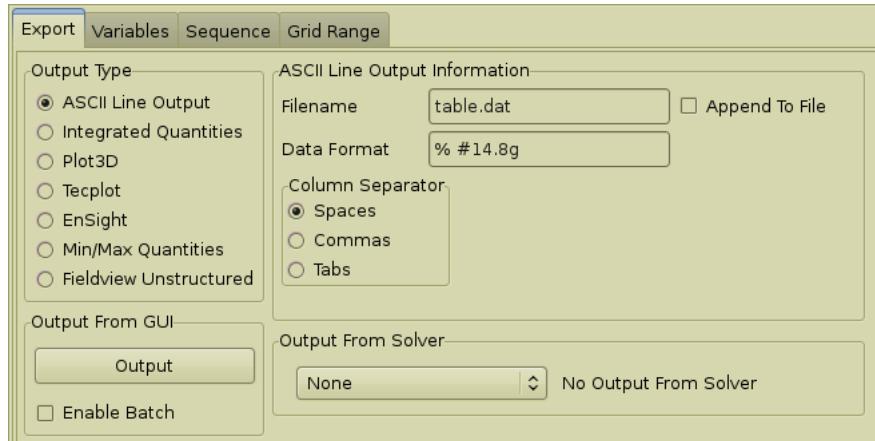


Figure 17.2: The File Output View

17.2.2 Folder/Node Properties

As with all tree elements, each output node and folder has properties which are specific to post processing. These properties are accessible when a node or folder is selected. Selecting **File Output** under the graphics window displays a tab sheet with all of the appropriate controls for setting the output properties. Within the **File Output** interface are four tabs: **Export**, **Variables**, **Sequence**, and **Grid Range** (see Fig. 17.2 on pg. 325). Each of these tabs represents a given set of properties associated with output folders and nodes.

17.3 The Export Property

Parameters relevant to exporting data from *GASPex* are modified by selecting the **Export** tab. These parameters are only sensitive in the GUI when a single *output folder* is selected. For all other folders and nodes (including the children of an output folder), these export parameters are not editable. The values set for a given output folder apply to that folder and all child nodes.

17.3.1 Output Type

The **Output Type** radio box shown in Fig. 17.3 on pg. 326 controls the output format for the selected output folder. Once a format is chosen, the information on the right side of the window changes accordingly. Each output type is described next.

ASCII Line Output

Selecting the **ASCII Line Output** displays information specific to tabular line output as shown in Fig. 17.4 on pg. 326. This format represents a formatted output suitable for viewing with a text editor or for input to a generic data analysis package. When information is printed

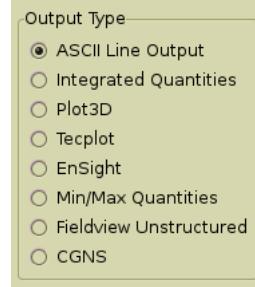


Figure 17.3: The Output Type

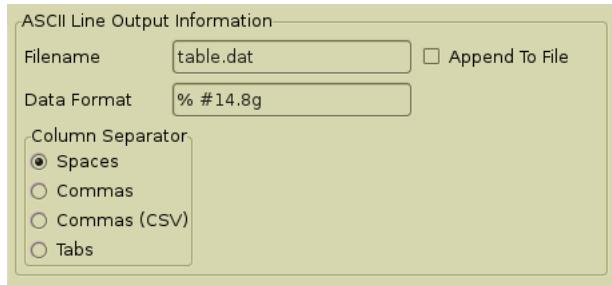


Figure 17.4: ASCII Output Section

to the file, each line will represent data from a single point within the domain. See Sec. 17.5 on pg. 354 for an explanation of setting the range of data points. Upon creation, data from each output node which is a child of the selected output folder will be included in this file. Each line of data will include all output quantities specified in the solution property for the given surface folder, as described in Sec. 17.4 on pg. 339.

A two-line header appears at the top of the output ASCII file. The first line contains the names of the variables listed in the file. The second line provide the units of each quantity.

The **Filename** type-in box displays the name of the file to be generated when output is performed. The file name can represent either a relative or full pathname. If a relative pathname is selected, then the file will be created relative to the directory containing the main input deck.

Several special token strings exist to insert solution information into the file name. These are

\$n – Insert the number of cycles in the file name. The string “\$n” is replaced with the current cycle number at the time of output.

\$t – Insert the time level in the file name. The string “\$t” is replaced with the current physical time step (in seconds) at the time of output.

\$s – Insert the sequence level in the file name. The string “\$s” is replaced with the sequence level at the time of output.

These string substitution parameters are useful for creating multiple files during a single execution of *GASPer*. For example, if the user wishes to monitor the pressure coefficient along the surface of an airfoil, as it varies with iteration, they can set the filename to `./cp-wing.$n`. If the current ASCII file is set to output every 100 cycles, then the following series of files will be created.

```
./cp-wing.100
./cp-wing.200
./cp-wing.300
```

The substitution can be formatted by appending a “C style” format in parenthesis after the token. For example, `$n("%3d")` or `$t("%7.3lf")`.

The **Data Format** type-in box controls the format for each output variable. This format string is passed to a C-style printf command for formatting all values printed to the file. The default format string of `% #14.8g` prints a number using 14 columns with 8 decimal places. The decimal point will be aligned, and all of the trailing zeros will be included.

The **Append To File** button should be selected if the user wishes to append information to the named file upon each execution of the output. Otherwise, the file will be re-created each execution. The motivation for appending to an existing file is to save a history of a given set of parameters. For example, if the user wishes to monitor the surface temperature and pressure at a given point, then they may create an ASCII output node and select the **Append To File** button. If the user specifies output of this node every 10 cycles, then the file will contain a single line of temperature and pressure data every 10 cycles. If the button is not selected, the file will only contain the most recent data. Since files can become quite large, care should be taken not to have a large number of data points selected when the append mode is selected. Note that the **Append To File** feature should not, in general, be used in conjunction with the filename substitution characters, `$n` and `$t`.

The **Column Separator** radio button selection allows the user to select the character used as a delimiter between columns of the output. Selecting “Spaces” will insert a number of single spaces in an effort to align columns for visualization in a text editor with a fixed width font. Selecting “Commas” or “Tabs” as the column delimiter is suitable for importing into spreadsheets or plotting packages. The “Commas (CSV)” option follows the common separated value format which is commonly used with spreadsheets.

Integrated Quantities

The integrated quantities selection is intended to provide a summation of the selected output variables over a range of faces. This is useful when the user is computing forces, moments, heat transfer, mass flow or total area on a surface. Note that the **Integrated Quantities** option is applicable only when interpolating to cell *faces*. The **Interpolation Type** is specified in the **Grid Range** tab, which is discussed in Sec. 17.5 on pg. 354.

Selecting the **Integrated Quantities** will display information specific to creating a file where quantities are summed over user specified surfaces as shown in Fig. 17.5 on pg. 328.

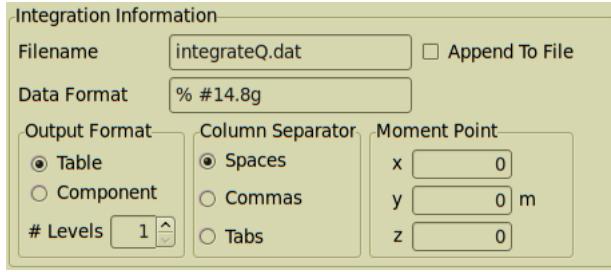


Figure 17.5: Integrated Quantity Section

The **Filename** type-in string displays the name of the file to be generated when output is performed. The file name can represent either a relative or full pathname. If a relative pathname is selected, then the file will be created relative to the directory containing the main input deck. As with ASCII output, the same special substitution strings (*i.e.*, \$n, \$t and \$s) change the behavior of the file name.

Two unique formatting styles are supported for organizing integrated quantity information. The style to be used for the current collection of integrated quantities is selected under the **Output Format**.

The **Table** format creates entries in a format similar to the ASCII Line Output. At the top of the file, a two-line header is created. The top line contains the names of the variables listed in the file. The second line contains a string representation of the units of each quantity. Beneath the header, a single line of integrated quantities is created for each execution of the output function for this output node. Note that the table format is most suitable for viewing integration histories, in conjunction with the **Append To File** feature. For example, if the user specifies output of this node every 10 cycles, then the file will contain a single line of integrated quantities every 10 cycles. Below is an example of lift and drag convergence history using table format.

Cycle	Fx_Inv	Fy_Inv
	N	N
10	71.0696	1987.95
20	86.0776	1991
30	76.2308	1987.9
40	62.6828	1973.74
50	57.8415	1966.45
60	51.5353	1962.68
70	41.6227	1959.94

The **Component** format creates entries in an indented style broken down by components of the output folders. For example, the user might select a group of surfaces which represent the entire boundary of an aircraft. Within the surface group, the surfaces are further organized into components (*e.g.*, fuselage, wing, horizontal, and vertical tail) If the user selects **Component** format, the file will not only contain values integrated over the entire aircraft,

but also contributions from each of the individual components. The values are listed one quantity per line. Each component level is indented, relative to its parent, and component names are also included in the file, as in the example below. Note that the parent values are listed below the component values.

```
grid:1 JMIN: 1- 16: 1- 36
  Fx_Vis      11.3795 N
  Fy_Vis     -0.253218 N
  Fx_Inv      103.89 N
  Fy_Inv      14768 N
```

```
grid:2 JMIN: 1- 16: 1- 36
  Fx_Vis      10.8854 N
  Fy_Vis      0.852614 N
  Fx_Inv     -116.083 N
  Fy_Inv     -12748.5 N
```

Integrated Qtys on cycle #1800

Fx_Vis	22.2649 N
Fy_Vis	0.599396 N
Fx_Inv	-12.1932 N
Fy_Inv	2019.54 N

The **# Levels** type-in is only relevant when the **Component** format is active. This integer value represents the number of hierarchical levels to display in the file. For example, the number of levels in the example above is two (the top level, and then the level immediately below the parent which contains two children).

Integrated output is formatted according to the C style format provided in the **Data Format** type-in. The **Append To File** button is selected if the user wishes to append integrated data to the named file upon each execution of the output. Otherwise, the file will be re-created upon each execution. These features are described in more detail above in the section discussing ASCII output.

The **Column Separator** radio button selection allows the user to select the character used as a delimiter between columns of the tabular formatted output. Selecting “Spaces” will insert a number of single spaces in an effort to align columns for visualization in a text editor with a fixed width font. Selecting “Commas” or “Tabs” as the column delimiter is suitable for importing into spreadsheets or plotting packages. The “Commas (CSV)” option follows the common separated value format which is commonly used with spreadsheets.

The **Moment Point** type-ins are relevant only to moment calculations. The moment point specifies the physical location of the point about which to take force moments for the current integration.

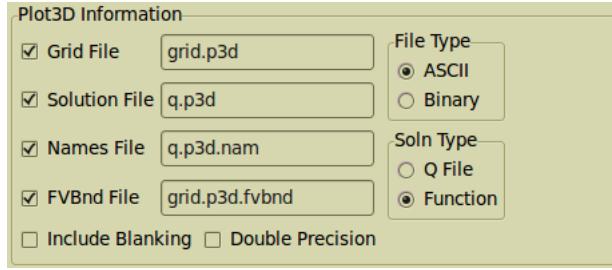


Figure 17.6: Plot3D Section

Plot3D

Selecting the **Plot3D** will display the information specific to creating grid and solution files in Plot3D format. Plot3D format is often used as input to external solution visualization programs such as Fieldview. The Plot3D format may also be used to transfer solutions to other analysis packages. With the Plot3D format, the user can create a grid file and solution files for any number of quantities. Parameters specific to the Plot3D output format are shown in Fig. 17.6 on pg. 330.

GASPer will always create Plot3D files with the Multi-Zone format. Each output node which is a child of the current output folder creates a separate zone within the Plot3D files. Zone ordering follows the ordering of the output nodes within the folder, which may or may not be consistent with zone ordering within the *GASPer* solution.

The **Grid File** type-in string displays the name of the grid file to be generated when output is performed. The file name can represent either a relative or full pathname. If a relative pathname is selected, then the file will be created relative to the directory containing the main input deck. The toggle button itself controls whether or not a grid file will be created.

The **Solution File** type-in string displays the name of the solution file to be generated when output is performed. The file name can represent either a relative or full pathname. If a relative pathname is selected, then the file will be created relative to the directory containing the main input deck. The toggle button itself controls whether or not a solution file will be created.

The **Names File** type-in string displays the name of a file which will contain identification of the parameters saved within the solution file. This file is used by some commercial post-processing packages to identify the data within the Plot3D solution file. The file name can represent either a relative or full pathname. If a relative pathname is selected, then the file will be created relative to the directory containing the main input deck. The toggle button itself controls whether or not a names file will be created.

The **FVBnd File** type-in string displays the name of a file which will contain boundary information for the selected zones in a format suitable for use by the Fieldview post-processing package. The boundary surfaces are identified by the most immediate surface group which is a parent for the particular boundary. The file name can represent either a relative or full pathname. If a relative pathname is selected, then the file will be created relative to the

directory containing the main input deck. The toggle button itself controls whether or not a names file will be created.

All filenames can utilize the special token strings $\$n$, $\$t$ and $\$s$. Hold the mouse over any type-in box to obtain tool tips. The use of tokens is explained in more detail in Sec. 17.3.1 on pg. 325.

The **File Type** parameter identifies the file format for the grid and solution file. **ASCII** format is more portable than the **Binary** format, but also results in much larger file sizes. I/O for ASCII format is typically slower than binary I/O. The **Binary** format provides single-precision values in the native format of the machine which creates the output. There is currently no support for creating non-native binary files in Plot3D format.

The **Soln Type** parameter controls the format for the Plot3D solution file. The **Q File** is a standard Plot3D solution file. The **Q File** format expects five variables specified in the **Variables** tab (see Sec. 17.4 on pg. 339). These five variables usually represent conservative variables for a perfect gas calculation, but this is not a requirement. The **Function** format represents a modified form of the standard Plot3D function file. In particular, the regular Plot3D function file only supports the specification of one or three variables. The modified format created in *GASPerx* supports an unlimited number of variables.

The **Include Blanking** toggle button is used to determine whether or not blanking information will be included in the Plot3D grid file. With overset grid schemes, the blanking information within the *GASPerx* solution file can be output to the grid file. In non-overset grid schemes, no blanking information exists. In this situation, including the blanking information is unnecessary.

The **Double Precision** toggle button is used to determine whether or not real values output to the grid and solution files will be represented as single or double precision values. Most commercial post processing packages recognize only single-precision Plot3D format. However, when transferring grids and solutions to a different solver, the use of double precision is recommended when it is supported.

Tecplot

Selecting the **Tecplot** option will display the information specific to creating an output file in the Tecplot format as shown in Fig. 17.7 on pg. 332. The file created by *GASPerx* will be either an ASCII formatted file or a binary file. The Tecplot format does not have separate grid and solution files. The user should include grid coordinates as solution variables when outputting in the Tecplot format.

The **File Name** type-in string displays the name of the Tecplot file to be generated when output is performed. The file name can represent either a relative or full pathname. If a relative pathname is selected, then the file will be created relative to the directory containing the main input deck. As with all other filename type-ins, the special token strings $\$n$, $\$t$ and $\$s$ may be used (representing cycle, time, and sequence level). Hold the mouse over any type-in box to obtain tool tips.

The **File Type** parameter identifies the file format for the Tecplot files. **ASCII** format is

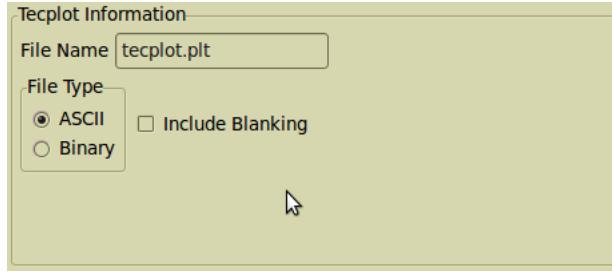


Figure 17.7: Tecplot Section

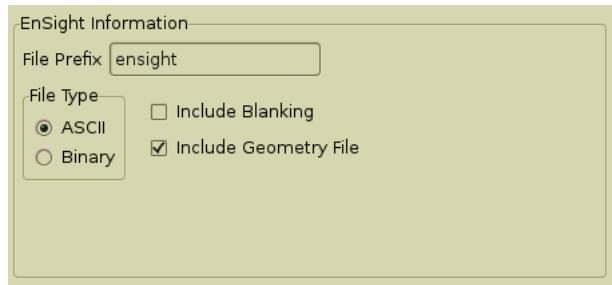


Figure 17.8: EnSight Section

more portable than the **Binary** format, but also results in larger file sizes. I/O for ASCII format is typically slower than binary I/O. The **Binary** format will be double precision values in the native format of the machine which creates the output.

The **Include Blanking** toggle button is used to determine whether or not blanking information will be included in the Tecplot file. With overset grid schemes, the blanking information within the *GASPE* solution file can be output to the Tecplot file. In non-overset grid schemes, no blanking information exists. In this situation, including the blanking information is unnecessary.

EnSight

Selecting the **EnSight** option will display the flags specific to creating an output file in the EnSight case file format as shown in Fig. 17.8 on pg. 332. The files created by *GASPE* will be either ASCII or binary formatted file. The EnSight format does include a separate file for the grid coordinates. The user need not include grid coordinates as solution variables when outputting in the EnSight format.

The **File Prefix** type-in string displays the name of file prefix to use when constructing filenames for the Ensight file format. The file prefix can represent either a relative or full pathname. If a relative pathname is selected, then the files will be created relative to the directory containing the main input deck. Upon output, *GASPE* will create a case file, a geometry file, and a solution file for each variable selected for output. The case file will correctly identify the contents of the solution files. The special token strings \$n, \$t and \$s

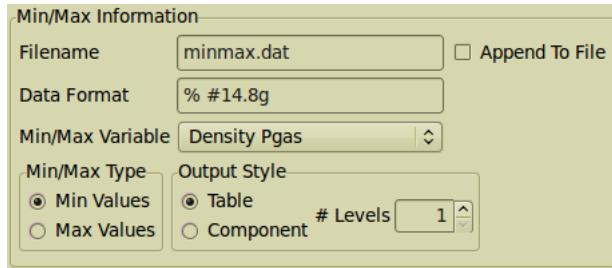


Figure 17.9: Min/Max Quantities Output Section

may be used. Hold the mouse over any type-in box to obtain tool tips.

The **File Type** parameter identifies the file format for the grid and solution files. **ASCII** format is more portable than the **Binary** format, but also results in larger file sizes. I/O for ASCII format is typically slower than binary I/O. The **Binary** format will be single precision values in the native format of the machine which creates the output. There is currently no support for creating non-native binary files in EnSight.

The **Include Blanking** toggle button is used to determine whether or not blanking information will be included in the EnSight geometry file. With overset grid schemes, the blanking information within the *GASPE* solution file can be output to the EnSight file. In non-overset grid schemes, no blanking information exists. In this situation, including blanking information is unnecessary.

The **Include Geometry File** option allows the user to print out the grid file or not. This may be of use if the user is printing out multiple solution files during a steady state run in which the grid file is not changing.

Min/Max Quantities

Selecting the **Min/Max Quantities** option will display the flags specific to creating a text file containing information relevant to the minimum or maximum value of one of the output variables as shown in Fig. 17.9 on pg. 333. When performing the output function for the given output node, all solution variables are computed at each interpolation point within the grid node. Once computed, the algorithm locates the interpolation point which contains the minimum or maximum value of the specified variable. At this point all variables are output to the specified file at the given interpolation point. This output option may be used to track the location of a minimum or maximum value. For example, one could monitor the location of the maximum temperature in a flow-field, and print the other flow quantities at that location.

The **Filename** type-in string displays the name of the file to be generated when output is performed. The file name can represent either a relative or full pathname. If a relative pathname is selected, then the file will be created relative to the directory containing the main input deck. As with ASCII output, the same special substitution strings (*i.e.*, \$n, \$t and \$s) change the behavior of the file name.

The **Table** format creates entries in a format similar to the ASCII Line Output. At the top of the file, a two-line header is created. The top line contains the names of the variables listed in the file. The second line contains a string representation of the units of each quantity. Beneath the header, a single line of integrated quantities is created for each execution of the output function for this output node. Note that the table format is most suitable for viewing solution histories, in conjunction with the **Append To File** feature. For example, if the user specifies output of this node every 5 cycles, then the file will contain a single line of data every 5 cycles. Below is an example of tracking the maximum Mach number, along with the thermodynamic state. Note that the location of minimum or maximum value is printed at the beginning of each line.

Zone	Location			Cycle	Density kg/m ³	Temperature K	Mach Number(max)
	i	j	k				
7	54	3	8	2005	0.0031375896	227.79411	8.6816908
7	54	3	8	2010	0.0031384691	227.84992	8.6801948
7	54	3	8	2015	0.0031398768	227.93807	8.6778550
7	54	3	5	2020	0.0031422246	227.98858	8.6762336
7	54	3	5	2025	0.0031430031	227.91403	8.6772897
7	54	3	8	2030	0.0031382602	227.82301	8.6790947
7	54	3	8	2035	0.0031325957	227.74152	8.6809522
7	54	3	5	2040	0.0031320430	227.64335	8.6835412
7	54	3	5	2045	0.0031291709	227.53424	8.6866444
7	54	3	8	2050	0.0031276685	227.41684	8.6902907

The **Component** format creates entries in an indented style broken down by components of the output folders. For example, the user might select a group of surfaces which represent the entire boundary of an aircraft. Within the surface group, the surfaces are further organized into components (*e.g.*, fuselage, wing, horizontal, and vertical tail) If the user selects **Component** format, the file will not only contain values representing the minimum or maximum over the entire aircraft, but also which represent the minimum or maximum of each of the individual components. The values are listed one quantity per line. Each component level is indented, relative to its parent, and component names are also included in the file, as in the example below. Note that the parent values are listed below the component values.

```

rocket:1  (zone = rocket:1,    index = 1)
(i,j,k) at max = (1, 49, 0)
Cycle          2050
Density        0.1000000 kg/m3
Pressure       6311.8008 N/m2
Temperature    220.00000 K
Mach Number(max) 2.0000000

```

```
rocket:2  (zone = rocket:2,    index = 2)
```

```
(i,j,k) at max = (0, 49, 1)
Cycle           2050
Density        0.10000000 kg/m^3
Pressure       6311.8008 N/m^2
Temperature    220.00000 K
Mach Number(max) 2.0000000
```

```
.
.
.
```

```
rocket:7 (zone = rocket:7, index = 7)
(i,j,k) at max = (54, 3, 8)
Cycle           2050
Density        0.0031276685 kg/m^3
Pressure       204.06755 N/m^2
Temperature    227.41684 K
Mach Number(max) 8.6902907
```

```
Max-Mach (zone of max variable = rocket:7, index = 7)
(i,j,k) at max = (54, 3, 8)
Cycle           2050
Density        0.0031276685 kg/m^3
Pressure       204.06755 N/m^2
Temperature    227.41684 K
Mach Number(max) 8.6902907
```

The **# Levels** type-in is only relevant when the **Component** format is active. This integer value represents the number of hierarchical levels to display in the file. For example, the number of levels in the example above is two (the top level, and then the level immediately below the parent which contains seven children).

Minimum/Maximum output is formatted according to the C style format provided in the **Data Format** type-in. The **Append To File** button is selected if the user wishes to append data to the named file upon each execution of the output. Otherwise, the file will be re-created upon each execution. These features are described in more detail above in the section discussing ASCII output.

FieldView Unstructured

Selecting the **FieldView Unstructured** option will display the flags specific to creating an output file in the FV-UNS file format as shown in Fig. 17.10 on pg. 336. The file created by



Figure 17.10: Fieldview File Output Section

*GAS*Pex will be either a ASCII or binary formatted file. The Fieldview file format written from *GAS*Pex can be either version 2.4 or version 3.0. There are three output options: grid file only, solution file only, and combined grid/solution file format. The user can select one or more of these options.

The file names can represent either a relative or full pathname. If a relative pathname is selected, then the files will be created relative to the directory containing the main input deck. The special token strings \$n, \$t and \$s may be used. Hold the mouse over any type-in box to obtain tool tips.

The **File Type** parameter identifies the file format for the grid and solution files. **ASCII** format is more portable than the **Binary** format, but also results in larger file sizes. I/O for ASCII format is typically slower than binary I/O. The **Binary** format will be single precision values in the native format of the machine which creates the output.

CGNS

The **CGNS** option is used to export the grid, the solution, or both to a file in CGNS format. The name of the output file is set using the **File Name** input. Special tokens can be used in the file name to insert cycle number, time, and the grid sequence level. The use of tokens is explained in more detail in Sec. 17.3.1 on pg. 325.

If grid data is to be included in the CGNS file, then ensure that the **Include Grid** option is selected. A similar option is available for including the solution.

The current CGNS version used in *GAS*Pex is 3.2.

17.3.2 The **Output** Button

The **Output** button is found in the **Output From GUI** section of the **Export** tab. Pressing the **Output** button results in an automatic variable calculation and output of the current output node. Note that execution of the post-processing functions may require significant amounts of memory, since solutions must be read from disk and intermediate values must be calculated. Remember that the GUI is a single-processor job and may require significant local resources. Computing post-processing functions from within the solver in a distributed-parallel environment may be possible even though performing the same output function is

not possible from the GUI due to memory requirements.

*Note that the information printed when the **Output** button is pressed corresponds to the physical time step selected in the **Time Levels** section of the GUI.*

17.3.3 The Enable Batch Option

The **Enable Batch** option allows the user to perform output in batch mode without running the GUI or the flow solver. This is designed to support output after a job has completed, but with parallel processing. To execute output for a selected output node, begin by selecting the enable batch option. Then from the command line, run

```
<mpirun syntax> gasp --batchoutput -i problem.gsp --run #
```

Batch output will perform output on all selected batch outputs which correspond to the sequence level selected by “`--run`”. Command line help can be found using

```
gasp --help --batchoutput
```

For time accurate problems, the batch output feature will only perform output on the current (last) solution. If the user would like to perform output on all the saved time levels, then the **Output All** option in the **Time Levels** menu should be used.

17.3.4 The Output From Solver Section

The parameters found in the **Output From Solver** section represent output options that can be performed while the flow solver is running. The *GASpex* flow solver can perform post-processing in parallel during a run.

There are five options that are available to the user. These options are selected from an option menu and determine the frequency of the output while a run definition is being performed. The first option is **None**, which indicates that no output will be performed for the output node during the run. This is the default setting for a new output node.

The next option is output **At End Of Run**. This results in the automatic calculation of the current output node at the completion of each run definition. The use of this option eliminates the need to run the GUI to create post-processing files for third-party analysis packages. This feature is of significant importance when running on a remote machine which does not support the GUI. Without this feature, the user would have to transfer the solution file back to a local machine simply to run the GUI and create post-processing files. The **Output From Solver** section is shown in Fig. 17.11 on pg. 338.

Selecting the output **During Run (cycles)** option results in the automatic post-processing of the current output node at a user-specified increment in the cycle number as specified in the cycle number type-in. A type-in box next to this option appears only when this option is selected. This option is normally used for steady state runs. The **Output From Solver** section is shown in Fig. 17.12 on pg. 338.

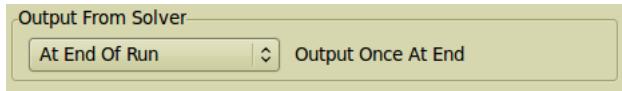


Figure 17.11: Output From Solver (At End of Run)

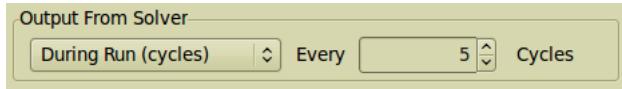


Figure 17.12: Output From Solver (During Run (cycles))

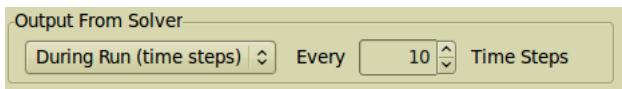


Figure 17.13: Output From Solver (During Run (time steps))

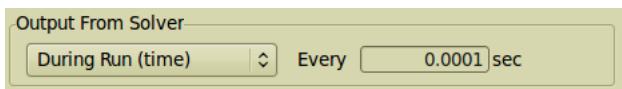


Figure 17.14: Output From Solver (During Run (time))

There are several uses for this output option. If the user wants to periodically check the solution without having to stop the solver execution, they can include this option. Note that if file substitution variables are not included in the output file names and the **Append To File** toggle button is not activated, then the files for the output will be overwritten upon each execution of the output. No history of the output will be saved in this case.

Selecting the output **During Run (time steps)** option results in the automatic calculation of the current output node at a user-specified increment in the number of time steps taken during a time-dependent run as specified in the **Time Steps** type-in. The type-in box appears only when this option is selected. The **Output From Solver** section is shown in Fig. 17.13 on pg. 338.

If the user wants to periodically check a time-dependent solution without having to stop the solver execution, they can include this option. Note that if file substitution variables are not included in the output file names and the **Append To File** toggle button is not activated, then the files for the output will be overwritten upon each execution of the output. No history of the output will be saved in this case.

Selecting the output **During Run (time)** option results in the automatic calculation of the current output node at a user-specified increment in physical time taken during a time-dependent run. The type-in box for specifying the time increment appears only when this option is selected. The **Output From Solver** section is shown in Fig. 17.14 on pg. 338.

Use of this option is appropriate for time-dependent data analysis. Note that if file substitution variables are not included in the output file names and the **Append To File** toggle button is not activated, then the files for the output will be overwritten upon each

execution of the output. Note the output may not occur at discrete times as calculated with respect to the output time step due to incompatible integration time steps. However, output will occur as close as possible to the specified increment in time.

If the **Append To File** toggle button is selected, then a solution history can be saved. For example, the user can monitor solution convergence through either an integrated quantity, a viscous coefficient (such as skin friction), or a set of solution variables at a select number of points.

If the user wishes to maintain a history of a more complex post-processing data set (such as a Plot3D file), then they may use the output **During Run (cycles)** option in combination with file-name substitution strings to create separate files for each successive output execution.

If the **Temporal Accuracy** is set to **Steady State** in the current run specification, then options **During Run (time steps)** and **During Run (time)** are ignored.

With these options, if the **Append To File** toggle button is selected then a time-dependent history can be saved in a single file. For example, the user can monitor the history of a discrete number of parameters such as integrated quantities, viscous coefficients or a set of solution variables.

If the user wishes to maintain a history of a more complex post-processing data set, such as a Plot3D file, then they may use the output **During Run (time steps)** option (as well as the **During Run (time)** option) in conjunction with file-name substitution strings. This allows the user to create separate files for each physical time step as specified in the time-step increment.

17.4 The Variables Property

The output variables are specified by selecting the **Variables** tab in the **File Output** section. The default settings are shown in Fig. 17.15 on pg. 340. In addition to defining the output variables, the output unit system is also prescribed here.

These variables are associated only with File Output folders. The interface is enabled when a single output folder is selected. The values set for a given output folder will apply to that folder and all child nodes.

17.4.1 Available Variables Filtered By Option

*GAS*Pex supports the output of more than one hundred variables. These variables have been organized into categories for efficient presentation. The **Available Variables Filtered By:** option menu is used to select the output variable category to be presented in the variables list. Some variables are independent of physical model and are available for any solution. Other variables may or may not be available, depending upon the currently selected physical model. If a filter is not applicable to the selected **Physical Model**, then it will not be selectable. Likewise, some variables are only appropriate when face type interpolation is active in the grid-range property (*e.g.*, integrated quantities). In this part of the reference

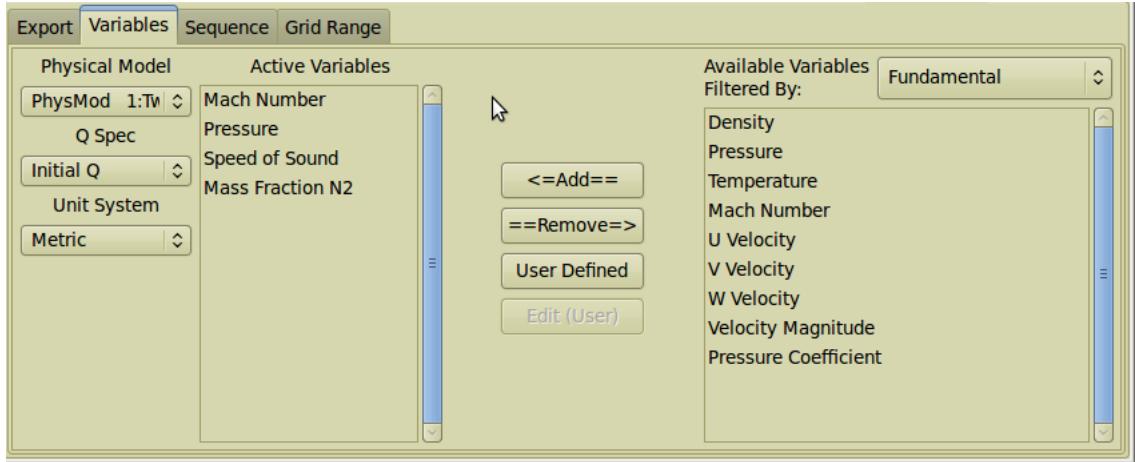


Figure 17.15: The Variables Window

manual, we list each variable, sorted by filter type, accompanied by a brief description of that variable.

- **Variables filtered by Fundamental:** The fundamental variables are the most commonly requested Navier-Stokes variables for display by third-party post processing packages.

Density The mixture density is computed by summing all the specie densities for the current physical model.

$$\rho = \sum_{i=1}^N \rho_i.$$

In the above expression, N is the total number of specie densities. The specie densities (ρ_i) are part of the Navier-Stokes primitive state vector.

Pressure This variable corresponds to the static pressure (p).

Temperature The translational temperature is computed according to the perfect-gas law

$$T = \frac{p}{\sum_{i=1}^N (\rho_i R_i)}.$$

Mach Number The Mach number is the ratio of the flow speed to the speed of sound

$$M \equiv \frac{\sqrt{u^2 + v^2 + w^2}}{a},$$

where a represents the speed of sound.

U Velocity, V Velocity, W Velocity The x , y and z components of the velocity, (u, v, w) , are elements of the primitive state vector.

Velocity Magnitude The magnitude of velocity is computed as

$$\|\vec{\mathbf{V}}\| = \sqrt{u^2 + v^2 + w^2}.$$

Pressure Coefficient The pressure coefficient is defined as

$$C_p = \frac{p - p_\infty}{1/2 \rho_\infty V_\infty^2}$$

The freestream values ($p_\infty, \rho_\infty, V_\infty$) are taken from the selected **Q Spec**.

Rel. U Velocity, Rel. V Velocity, Rel. W Velocity The relative x, y and z components of the velocity vector when a rotating frame of reference is used.

Rel. Mach Number The relative Mach number when a rotating frame of reference is used.

- **Variables filtered by Miscellaneous:** The variables in the **Miscellaneous** section are typically independent of the solution or the geometry.

Cycle This variable corresponds to the current cycle number. This variable might be used when monitoring solution convergence through a “watch point”.

Time This variable corresponds to the current physical time. This variable is appropriate for time-dependent problems.

Time Step For time-accurate flows, the time step is equal to the physical time step. For steady-state flows, the time step is the “pseudo” time step computed from either the CFL number or constant Dt. When using the CFL option in the time-integration settings, a time step is calculated for each cell based on cell geometry and either the reference velocity or local velocity.

The time step requires information from a run definition, which is determined based on when the output is produced. If output is produced while running the flow solver, then the active run definition is used. If output is produced from the GUI, then the selection is a little more complicated. First, a run definition selected for execution which contains the zone and sequence of the output node is sought. If one cannot be found, then the first run definition that matches the sequence is used. If both of these searches fail, then the first run definition is used for output.

Time Step Limit Factor When using the automated time step limiting option (see Sec. 15.3.5 on pg. 252), the user may wish to know the limit factor that is applied to the time step on a cell by cell basis. Note that the limit factor is computed and used during the solver, so GUI output of this variable is not meaningful.

CFL The variable corresponds to the Courant number. The output may be constant (if the CFL number is used) or vary with each cell in the case of a constant Dt. If the time step is based on a constant Dt, then CFL is computed using the time step, local cell size, and a reference velocity.

See the note on time step regarding which run definition is used to compute the CFL number.

- **Variables filtered by Geometric:** Geometric variables are those which are strictly a function of the geometry or grid. No solution is required to output geometric variables.

I Coordinate, J Coordinate, K Coordinate These variables correspond to the "I", "J", and "K" location in generalized coordinates for the current zone. These variables are only applicable for a structured mesh.

X Coordinate, Y Coordinate, Z Coordinate These variables correspond to the "X", "Y", and "Z" position in physical coordinates for the current zone.

If the Output Type selected in the Export tab, (see Sec. 17.3 on pg. 325), is either Plot3D or EnSight, the x, y and z coordinates are included in the output file by default, and including these variables in the Active Variables list is not necessary. However, these values are not included by default for any other output type.

Volume This variable corresponds to the volume of a given grid cell.

Area This variable corresponds to the cell-face area for a given face. The area is only available for output when the Interpolation Type is set to Faces in the Grid Range tab.

unit normal: nx, unit normal: ny, unit normal: nz These variables correspond to the x , y , and z components of the outward-facing normal of a given cell face. These variables should only be used for output when the Interpolation Type is set to Faces in the Grid Range tab.

Partition # When zones are decomped for load balancing, a unique index is assigned to each partition that results. The user may determine the partition index of each cell using this variable. The use of this variable is most applicable when the user would like to see how zones are decomped within the solver.

Cell Type For Chimera or overlapping zones, this gives a numeric value for the cell type. The values are given in order that the cell type appears in the grid range. For example, discrete cells equal 1, interpolation cells equal 2, projection cells equal 3, hole cells equal 4, peeled cells equal 5, and orphan cells equal 6.

Wall Distance This variable corresponds to the distance between the cell center of the current control volume and the closest wall point in the domain. This value is dependent upon the Wall Distance Method in the Viscous section of the Navier-Stokes physical model (see Sec. 11.4 on pg. 181).

- **Variables filtered by Residuals:** Most physical models in *GASpex* are represented by a system of governing equations that are solved for during the simulation. The residual value represents how well the solution satisfies the equation solved for. This filter gives the L2-norm of the residual for each governing equation of the physical model.

- **Variables filtered by Plot3D Conservative:** The Plot3D Conservative variables represent the five variables found in a Plot3D solution file. This filter is only available for the Navier Stokes Physical Model.

Third party post-processing packages which expect Plot3D conservative Q file will only be able to correctly interpret these variables when the physical model uses a single specie, perfect-gas equation of state.

Density The mixture density is computed by summing all the specie densities for the current physical model.

$$\rho = \sum_{i=1}^N \rho_i.$$

The specie densities are part of the primitive state vector.

X Momentum, Y Momentum, Z Momentum These variables correspond to the "X", "Y", and "Z" momentum per unit volume in the X, Y, and Z coordinate directions, respectively.

Total Energy The variable corresponds to the total internal energy per unit volume defined as

$$e_0 = e + \frac{\vec{V} \cdot \vec{V}}{2}$$

where

$$e = \sum_{i=1}^N \frac{\rho_i}{\rho} \tilde{e}_i + \sum_{j=1}^M \frac{\rho_j}{\rho} \tilde{e}_{n_j},$$

\tilde{e}_{n_j} is the non-equilibrium energy for specie j and the equilibrium energy for specie i is

$$\tilde{e}_i(T) = \int_{T_{\text{ref}}}^T \tilde{c}_{v_i}(\tau) d\tau + h_{f_i}^0.$$

- **Variables filtered by Primitive:** The primitive variables represent the variables for which *GASPer* solves the governing equations. The available primitive variables will differ depending upon the selected **Physical Model**.

Navier-Stokes (NS) Model: Primitive variables include the specie densities, the velocity components, and the pressure. Additional variables are also possible depending on the model settings. These include non-equilibrium components of internal energy, turbulence quantities, and temperature.

Solid Thermodynamics (ST) Model: Primitive variable includes the solid material temperature.

Mechanical Stress (MS) Model: Primitive variables include the X, Y, and Z material displacements.

Lagrange Spray Model: Primitive variables are the solution parameters associated with each particle. These include the particle index value, time, (X,Y,Z) coordinates, (U,V,W) velocity components, temperature, mass and particle diameter.

- **Variables filtered by Thermodynamics:** Variables related to thermodynamics are listed in this filter. This filter is only available for the Navier Stokes Physical Model.

Temperature This variable corresponds to the translational temperature.

Pressure This variable corresponds to the static pressure (p).

Density The mixture density is computed by summing all the specie densities for the current physical model.

$$\rho = \sum_{i=1}^N \rho_i.$$

The specie densities are part of the primitive state vector.

Speed of Sound The frozen speed of sound, a , is defined by $a^2 = \gamma p / \rho$.

Gamma This variable corresponds to the ratio of specific heats, $\tilde{\gamma} = \tilde{c}_p / \tilde{c}_v$ where \tilde{c}_p and \tilde{c}_v are defined as

$$\tilde{c}_p = \sum_{i=1}^N \frac{\rho_i}{\rho} \tilde{c}_{p_i} \quad \tilde{c}_v = \sum_{i=1}^N \frac{\rho_i}{\rho} \tilde{c}_{v_i}.$$

Cp The specific heat at constant pressure,

$$\tilde{c}_p = \sum_{i=1}^N \frac{\rho_i}{\rho} \tilde{c}_{p_i}$$

Cv The specific heat at constant volume,

$$\tilde{c}_v = \sum_{i=1}^N \frac{\rho_i}{\rho} \tilde{c}_{v_i}.$$

Mixture Gas Constant The mixture gas constant,

$$R = \sum_{i=1}^N \frac{\rho_i}{\rho} R_i.$$

Mixture Molecular Wt. The mixture molecular weight,

$$M_w = \sum_{i=1}^N \frac{\rho_i R_i}{\rho R} M_{wi}.$$

Entropy This variable corresponds to a measure of the entropy, s , defined as $s = p/\rho^\gamma$.

- **Variables filtered by Laminar Transport:** Transport properties are listed in this filter. This filter is only available for the Navier Stokes Physical Model.

Viscosity The mixture viscosity, μ , is computed using the transport property models found in the selected physical model and Wilkes' mixing rule.

Thermal Conductivity The mixture thermal conductivity, k , computed using the transport property models found in the selected physical model and Wilkes' mixing rule.

Diffusion Coefficient *SpeciesName* The laminar diffusion coefficient value for a given gas species.

- **Variables filtered by Turbulent:** Variables related to turbulent flow are listed in this filter. This filter is only available for the Navier Stokes Physical Model.

y+ This variable corresponds to the non-dimensional turbulence distance, $y^+ = \rho u^* y / \mu$.

u+ This variable corresponds to the non-dimensional turbulence velocity, $u^+ = u / u^*$.

Eddy Viscosity This variable corresponds to the turbulence model's eddy viscosity, μ_t .

Eddy Viscosity/Laminar Viscosity This variable corresponds to the ratio of eddy viscosity to laminar viscosity, μ_t/μ .

Deformation This variable corresponds to the flow deformation defined as

$$Def \equiv \frac{S_{ij}S_{ij} - \Omega_{ij}\Omega_{ij}}{S_{ij}S_{ij} + \Omega_{ij}\Omega_{ij}}.$$

$Def = -1$ corresponds to solid-body rotation. $Def = 0$ corresponds to pure shear. $Def = +1$ corresponds to pure strain.

Spalart Variable This variable corresponds to $\tilde{\nu}$ in the Spalart Allmaras one-equation turbulence model.

K+ This variable corresponds to the non-dimensional TKE, k^+ .

Turbulent Kinetic Energy This variable corresponds to the turbulent kinetic energy, k .

Epsilon The TKE dissipation rate per unit mass, ϵ .

Omega The inverse of the turbulence turn-over time, ω . Also, the dissipation per unit TKE, $\omega = \epsilon/k$.

Rho-K The turbulent kinetic energy per unit volume, ρk .

Rho-Epsilon The TKE dissipation rate per unit volume, $\rho\epsilon$.

Rho-Omega The TKE dissipation rate per unit volume, $\rho\epsilon$.

Turbulence Length The turbulence length scale, $l_t = k^{3/2}/\epsilon$.

Turbulence Intensity The turbulence intensity, $T.E. = \sqrt{(2/3 k)/(\vec{V} \cdot \vec{V})}$.

Relative Strain, Sk/e This variable corresponds to the strain rate S normalized by a time scale for the turbulence (ϵ/k).

Production/Dissipation The ratio of the turbulence production to dissipation, $\mathcal{P}/(\rho\epsilon)$, indicates whether the locally generated turbulence dominates the local structure. For the equilibrium log layer, $\mathcal{P}/(\rho\epsilon) = 1$.

Menter's F1 This variable (F_1) controls the blending of models in Menter's SST turbulence model.

Reynolds Stress; Txx, Reynolds Stress; Tyy, Reynolds Stress; Tzz These variables correspond to the Reynolds normal stresses $\tau_{xx}, \tau_{yy}, \tau_{zz}$ in the X, Y, and Z directions respectively.

Reynolds Stress; Txy, Reynolds Stress; Txz, Reynolds Stress; Tyz These variables correspond to the Reynolds shear stress $\tau_{xy}, \tau_{xz}, \tau_{yz}$ in the XY, XZ, and YZ planes respectively.

LES Blend Fcn The blending function for the BSL/LES hybrid model.

LES * Avg The average value for the BSL/LES hybrid model where * is one of the following: Density (Rho), TKE, Omega, Rho*U, Rho*V, Rho*W, or (Rho*U*U + Rho*V*V + Rho*W*W).

Intermittency The turbulent intermittency value computed by solving Mentor's transition equation [49, 50].

Transition Reynolds No. The local transition onset momentum thickness Reynolds number computed by solving Mentor's transition equation [49, 50].

Reynolds No. Correlation The connection between the critical Reynolds number where the intermittency first starts to increase in the boundary layer and the transition Reynolds number.

- **Variables filtered by Stagnation:** Stagnation properties are listed in this filter. This filter is only available for the Navier Stokes Physical Model.

Stagnation Density The stagnation density represents a mixture stagnation or total density, defined assuming isentropic flow. $\rho_t = \rho(T_t/T)^{1/\gamma}$.

Stagnation Pressure The Stagnation Pressure represents a stagnation or total pressure, defined assuming isentropic flow. $p_t = p(T_t/T)^{\gamma(\gamma-1)}$.

Stagnation Temperature The Stagnation Temperature represents the stagnation or total temperature defined where $h(T_0) = h(T) + 1/2 \rho (\vec{V} \cdot \vec{V})$.

Stagnation Enthalpy The Stagnation Enthalpy represents the stagnation or total enthalpy per unit mass defined as $h_0 = h(T) + 1/2 \rho(\vec{V} \cdot \vec{V})$.

Stagnation Energy The Stagnation Energy represents the stagnation or total energy per unit mass defined as $e_0 = h_0 - p/\rho$.

- **Variables filtered by Gradient Based:** Variables which require flow gradients are listed in this filter. The gradient calculation require more CPU time during output than other variable categories.

Vorticity Magnitude The Vorticity Magnitude represents the magnitude of the X, Y, and Z components of vorticity, $\nabla \times \vec{V}$. If the vorticity tensor is represented by

$$\Omega_{ij} = \frac{1}{2} \left(\frac{\partial u_i}{\partial x_j} - \frac{\partial u_j}{\partial x_i} \right)$$

then the magnitude of the tensor is computed from $(2\Omega_{ij}\Omega_{ij})^{1/2}$.

X Vorticity, Y Vorticity, Z Vorticity The X, Y, and Z Vorticities are the X, Y, and Z components of the vorticity defined by $\nabla \times \vec{V}$.

Vortex Core This is intended to give the strength of the vortex core. The vortex core correspond to a value near ± 1 . [57],[58]

Mach Normal The Mach normal is defined as,

$$M_n = \frac{u \frac{dP}{dx} + v \frac{dP}{dy} + w \frac{dP}{dz}}{a \sqrt{\left(\frac{dP}{dx}\right)^2 + \left(\frac{dP}{dy}\right)^2 + \left(\frac{dP}{dz}\right)^2}}$$

where a represents the speed of sound.

Helicity The helicity is the dot product of the normalized vorticity and normalized velocity vectors. Helicity occurs when flow is rotating about the flow direction.

Vortex Lambda 2 This is the second eigenvalue of the $S_{ik}S_{kj} + \Omega_{ik}\Omega_{kj}$ matrix. A negative value for the eigenvalue should indicate a vortex core region. [59]

Strain-Rate Magnitude The fluid dynamic strain-rate tensor is defined as

$$S_{ij} = \frac{1}{2} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)$$

The magnitude of the tensor is computed from $(2S_{ij}S_{ij})^{1/2}$.

dQ/dx, dQ/dy, dQ/dz Gradients of Q with respect to the x, y, and z coordinate directions where Q represents u-velocity, v-velocity, w-velocity, pressure or species densities.

- **Variables filtered by Skin Fric/Heat Trans:** The variables in this section are associated with the viscous shearing forces and thermal conduction of heat. They are calculated from the viscous contributions to the Navier-Stokes governing equation, using the modeling parameters of the current physical model.

These variables are only applicable while the interpolation type is faces,

Cf The skin friction coefficient for a face, C_f , is a non-dimensional representation of the viscous force on a face. The viscous force used in this calculation is that force which is tangent to the face, and in the direction of the free-stream velocity component.

This variable is only applicable while the interpolation type is faces, and is not suitable for integration.

|**Cf|** The skin friction coefficient magnitude for a face, $\|C_f\|$, is a non-dimensional representation of the viscous force on a face. The viscous force used in this calculation is the magnitude of the force which is tangent to the face.

This variable is only applicable while the interpolation type is faces, and is not suitable for integration.

Ch The heat transfer coefficient, C_h on a face.

$$C_h = \frac{q}{\rho_\infty V_\infty [(h_0)_\infty - (h_0)_w]}$$

The freestream values $((h_0)_\infty, \rho_\infty, V_\infty)$ are taken from the selected **Q Spec**.

Heat Flux (q_dot) The heat transfer rate, \dot{q} , on a face. The heat transfer is defined as $-k(dT/dn)$. In order to have a consistent sign convention on surfaces, the sign is flipped on idim, jdim, and kdim surfaces. The heat transfer on a surface should be positive for heat transfer from the wall into the flow and negative for heat transfer from the flow into the solid.

Heat Power (q_dot*da) The heat transfer through a face due to conduction. This is the heat flux multiplied by the face area to yield power.

Viscous Energy Flux*Area The heat transfer through a face due to all viscous contributions, including conduction, diffusion, and convection of non-equilibrium internal energy. This is the flux computed from the energy equation multiplied by the face area to give a power.

Film Coefficient The film coefficient is also called the heat transfer coefficient and comes from Newton's law of cooling. This equation expresses the heat flux as,

$$\dot{q} = -k(dT/dn) = h(T - T_{ref}).$$

where T_{ref} is a reference temperature taken from the **Q Spec** selected for output and h is the film coefficient. The heat transfer coefficient is useful since it can be

an input for the convection boundary condition in solid thermodynamic problems. The film coefficient can also be used to compute the Nusselt number.

- **Variables filtered by Forces:** The variables in this section are associated with the forces acting on a face, as calculated in the momentum equations of the Navier-Stokes governing equations. The calculations are performed using the modeling parameters of the current physical model. These variables are intended for file output using the integrated quantities option and not visualization from within the GUI.

These variables are only applicable while the interpolation type is faces.

For surfaces which represent physical boundaries, the forces output using these variables represent forces acting on the surface.

Fx_Total, Fy_Total, Fz_Total The force components, $F_{x_{\text{tot}}}$, $F_{y_{\text{tot}}}$ and $F_{z_{\text{tot}}}$ represent the components of force acting on a face in the X, Y, and Z component directions respectively due to pressure and viscous forces.

Fx_Inv, Fy_Inv, Fz_Inv The force components, $F_{x_{\text{inv}}}$, $F_{y_{\text{inv}}}$ and $F_{z_{\text{inv}}}$ represent the components of force acting on a face in the X, Y, and Z component directions respectively due solely to the pressure.

Fx_Vis, Fy_Vis, Fz_Vis The force components, $F_{x_{\text{vis}}}$, $F_{y_{\text{vis}}}$ and $F_{z_{\text{vis}}}$ represent the components of force acting on a face in the X, Y, and Z component directions respectively due solely to the viscous contributions to the momentum equations.

Momentum_Force_x, Momentum_Force_y, Momentum_Force_z These force components, represent the components of force acting on a face in the X, Y, and Z component directions respectively due to pressure forces, viscous forces, and momentum flux. These force components are suitable for integrating the thrust component of a flow-through boundary condition such as a nozzle exit. These forces are computed by directly evaluating the inviscid and viscous flux models of the current physical model.

- **Variables filtered by Moment:** The variables in this section are associated with the force moments acting on a face, as calculated in the momentum equations of the Navier-Stokes governing equations. The calculations are performed using the modeling parameters of the current physical model. The point about which to take the moment is found in the **Integrated Quantities** section of the output property, Sec. 17.3.1 on pg. 327.

These variables are only applicable while the interpolation type is faces.

For surfaces which represent physical boundaries, the moments output using these variables represent moments of forces acting on the surface.

Mx_Total, My_Total, Mz_Total The moment components, $M_{x_{\text{tot}}}$, $M_{y_{\text{tot}}}$ and $M_{z_{\text{tot}}}$ represent the components of moments acting on a face in the X, Y, and Z component directions respectively due to pressure and viscous forces.

Mx_Inv, My_Inv, Mz_Inv The moment components, $M_{x_{\text{inv}}}$, $M_{y_{\text{inv}}}$ and $M_{z_{\text{inv}}}$ represent the components of moments acting on a face in the X, Y, and Z component directions respectively due solely to the pressure force.

Mx_Vis, My_Vis, Mz_Vis The moment components, $M_{x_{\text{vis}}}$, $M_{y_{\text{vis}}}$ and $M_{z_{\text{vis}}}$ represent the components of moments acting on a face in the X, Y, and Z component directions respectively due solely to the viscous force.

Mx_Momentum, My_Momentum, Mz_Momentum These variables represent the components of the moment acting on a face in the X, Y, and Z component directions respectively due to pressure forces, viscous forces, and momentum flux. The force used in the moment is computed by directly evaluating the inviscid and viscous flux models of the current physical model.

- **Variables filtered by Mass Flow:** The variables in this section are associated with the mass flow through a face, as calculated in the continuity equations. The calculations are performed using the modeling parameters of the current physical model. The direction of the outward pointing normal is constructed using a right handed rule for the current grid.

These variables are only applicable while the interpolation type is faces.

Area This variable corresponds to the cell-face area for a given face. The area is only available for output when the **Interpolation Type** is set to **Faces** in the **Grid Range** tab.

Mass Flow The Mass Flow variable represents the total mass flow through a face. It represents the summation of mass flows of all the individual species. The value can be either positive or negative, depending on both the grid and flow orientation. The quantity represents the actual value computed by the flux.

Mass Flow Bnd The Mass Flow Bnd variable represents the total mass flow through a face. Its value is the summation of mass flows of all the individual species. The sign of the mass flow is adjusted such that positive mass flow represents mass flowing into the domain, while negative mass flow represents flow out of the domain.

Mass Flow SpeciesName The *SpeciesName* will be one of the species names for the current physical model. This variable represents the component mass flow for that particular specie.

Mass Flow Bnd SpeciesName The *SpeciesName* will be one of the species names for the current physical model. This variable represents the component mass

flow for that particular specie. The sign of the mass flow is adjusted such that positive mass flow represents mass flowing into the domain, while negative mass flow represents flow out of the domain.

- **Variables filtered by Fluxes:** The variables in this section are associated with the computed flux through a face, as calculated in the flow solver. The calculations are performed using the modeling parameters of the current Navier-Stokes physical model. The direction of the outward pointing normal is constructed using a right handed rule for the current grid.

These variables are only applicable while the interpolation type is faces.

Continuity Flux SpeciesName The *SpeciesName* will be one of the species names for the current physical model. This variable represents the component mass flux for that particular specie.

U Momentum Flux, V Momentum Flux, W Momentum Flux The Momentum Flux represents the sum of momentum transport, as well as forces applied to the face in the *X*, *Y*, and *Z* component directions respectively.

Energy Flux The Energy Flux represents the sum of energy transport per unit mass, as well as any heat applied to the face.

- **Variables filtered by Chemistry:** The variables found in the Chemistry section represent local state variables which will vary with selected chemistry models.

Damkohler Number The Damkohler number represents the ratio of the fluid dynamic time scale to the chemical reaction time scale. Large Damkohler numbers indicate that the chemical reaction will tend to equilibrium much more quickly than the particles will traverse a control volume.

Heat Release Rate The Heat Release Rate measures the the rate of energy release into a flow due to chemical reactions.

Density SpeciesName The *SpeciesName* will be one of the species names for the current physical model, and represents the component density for that particular specie.

Mass Fraction SpeciesName The *SpeciesName* will be one of the species names for the current physical model, and represents the component mass fraction for that particular specie.

Mole Fraction SpeciesName The *SpeciesName* will be one of the species names for the current physical model, and represents the component mole fraction for that particular specie.

Internal Energy SpeciesName The *SpeciesName* will be one of the species names for the current physical model, and represents the component internal energy for that particular specie.

Production Rate *SpeciesName* The *SpeciesName* will be one of the species names for the current physical model, and represents the chemical production rate for that particular specie.

Number Density *SpeciesName* The *SpeciesName* will be one of the species names for the current physical model, and represents the component number density (particles per volume) for that particular specie.

- **Variables filtered by Condensation:** The variables in this section are specific to the water condensation model located in the Navier-Stokes physical model.

Log(N): *SpeciesName* The *SpeciesName* will be one of the species names for the current physical model, and represents the specie logarithmic droplet number density.

N: *SpeciesName* The *SpeciesName* will be one of the species names for the current physical model, and represents the specie droplet number density.

dr/dt: *SpeciesName* The *SpeciesName* will be one of the species names for the current physical model, and represents the specie droplet radius rate of change.

Psr: *SpeciesName* The *SpeciesName* will be one of the species names for the current physical model, and represents the specie droplet saturation pressure of change.

Surface Tension The surface tension based on the condensation mixture.

Saturation Pressure The saturation pressure based on the condensation mixture.

Vapor Pressure The vapor pressure based on the condensation mixture.

Critical Radius

Water Density (rhoL)

Total Water Density

Total Water Mass Fraction

Saturation Pressure Ratio

Log Saturation Pressure Ratio

Nucleation Rate The rate of nucleation of liquid droplets to the gas phase.

- **Variables filtered by Mechanical Stress:** The variables in this section are specific to the Mechanical Stress physical model.

X, Y, Z Displacement The material displacement in the physical coordinate system.

Tau_xx, Tau_xy, Tau_xz, Tau_yy, Tau_yz, Tau_zz The material stress tensor components.

Temperature The material temperature used in the stress equations. The temperature is either a fixed value or taken from the solid thermodynamics through the use of a coupler.

- **Variables filtered by Solid Thermodynamics:** The variables in this section are associated with any zones which are run with a Solid Material physical model.

Solid Temperature The temperature of the solid.

Solid Density The density of the solid as computed with the current physical model.

Solid Thermal Conductivity The Thermal Conductivity of the solid as computed with the current physical model.

Solid Specific Heat The latent specific heat of the solid as computed with the current physical model.

Solid Heat Power The heat flux across the given face as computed with the current physical model multiplied by the face area.

This variable is only applicable while the interpolation type is faces.

Solid Heat Flux The heat flux (power per unit area) across the given face as computed with the current physical model.

This variable is only applicable while the interpolation type is faces.

Solid Heat Transfer Coef The heat transfer coefficient across the given face, as computed with the current physical model.

This variable is only applicable while the interpolation type is faces.

17.4.2 The Active Variables List

In Fig. 17.15 on pg. 340, the **Active Variables** list on the left displays the output variables for the selected output folder. These variables are output when the **Output** button is pushed as described in Sec. 17.3 on pg. 325. Initially, the **Active Variables** list is empty. Variables are added by selecting from **Available Variables** list and pressing the **<=Add=>** button located between the two respective lists. If no variable is selected in the **Active Variables** list, then added variables are appended at the bottom. Variables may be deleted from the **Active Variables** list by selecting them and clicking on the **==Remove==>** button.

User Defined Variables

User-defined variables can be constructed by clicking on the **User Defined** button. A pop-up window will appear in which the user can provide the variable name, the defining equation and the units. The variable name and units that the user provides are for reference purposes only and will not change the output. The defining equation must consist of variables and symbols as shown in the token key. Standard operators like addition(+), subtraction(-),

division(/) and multiplication(*) are supported. Once an equation is defined, press the **Add Variable** button. Once all the user defined variables are created, the window can be closed using **Done**.

If the defining equation cannot be parsed due to unsupported operators or characters, the output will be zero. A list of all the output variables with their corresponding token key name is given in a scrollable list. The token name represents the expression to be used in the defining equation.

The Physical Model Option Menu

Some of the output variables depend on the modeling parameters found in a physical model. For example, the internal energy of a system depends on the thermodynamic model found in the **Thermo-Chem** section of the **Physical Models** tab. Likewise, the skin-friction calculation depends upon the viscous modeling parameters selected in the **Viscous** section.

Referring to the left-hand part of Fig. 17.15 on pg. 340, the physical model for output is selected by using the **Physical Model** option menu. Care should be taken to ensure that an appropriate physical model is selected for the zones being analyzed.

Note that many of the output variables displayed in the Available Variables list depend on the currently selected physical model. For example, turbulent quantities are not available if the current physical model describes inviscid or laminar flow.

The Q Spec Option Menu

If any of the output variables depend on reference quantities, these values are computed from the current **Q Spec** option menu for the selected physical model. Example variables include the pressure coefficient or heat-transfer coefficient.

The Unit System Option Menu

Variables are output in the unit system selected under the **Unit System** option menu. The output unit system is independent of the unit system used for describing the input parameters for the current problem. The unit system for the current Output Folder is also independent of other Output Folders.

17.5 The Grid Range Property

Parameters relevant to the computational coordinates are edited by selecting the **Grid Range** tab shown in Fig. 17.16 on pg. 355. In particular, the type of range (*e.g.*, volume, surface, line or point), interpolation type (*e.g.*, nodes, cell centers, faces), cell types (relevant to Chimera meshes) and the min/max coordinates in the *I*, *J* and *K* direction are specified here.

Each output node has a distinct grid-range definition. The user can set all of the parameters for a node independent of the other nodes. The GUI for the grid-range property

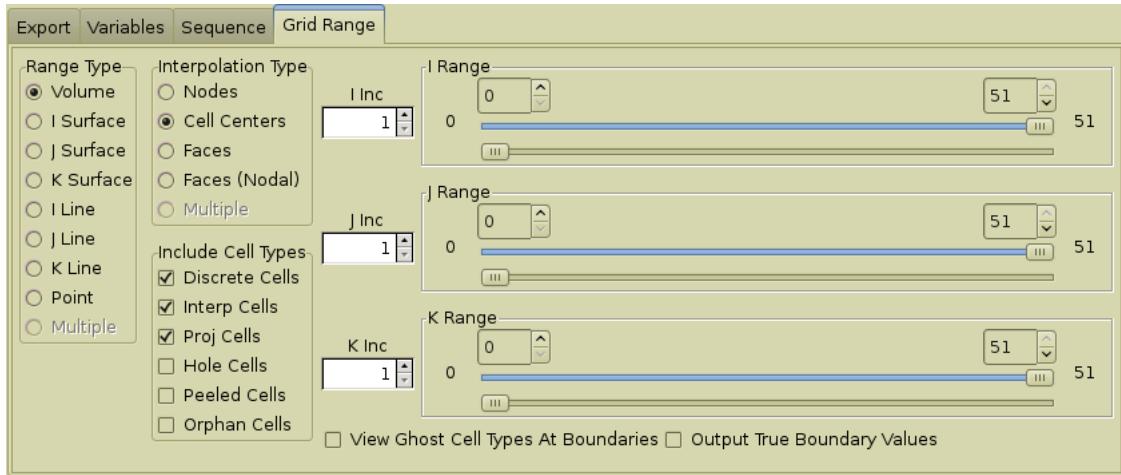


Figure 17.16: The Grid-Range Property Window

is capable of simultaneously editing parameters for more than one output node. Whenever an output folder or a collection of output nodes is selected, the grid-range property display represents the collective values of the selected output nodes. Changing a parameter in the grid-range tab changes that parameter for all selected nodes. If changing a parameter for a multiple selection is not possible, then the appropriate widget will be disabled. In general, editing operations should be performed while the parent output folder is selected, so that changes may be made to both the grid-range property and the solution property without changing node selections. Editing a grid-range property on a single node typically occurs only when merged editing is not possible.

17.5.1 The Range Type Selection Widget

Volumes, surfaces, lines and points are called “range types”. Radio buttons are selectable in the **Range Type** section shown on the left side of Fig. 17.16 on pg. 355. Each range type controls the sliders for selecting the logical coordinates shown on the right of the window. We describe each range type below.

Volume The **Volume** range type is primarily used to export files to third-party three-dimensional visualization packages such as Tecplot, Fieldview or Ensight. When the range type is volume, all of the **I Range**, **J Range** and **K Range** sliders will have a minimum and maximum value.

The **Volume** range type is also the most appropriate choice for visualizing overset grids. When used in conjunction with the **Include Cell Types** toggle buttons, the user may easily identify hole cells and orphan cells created during the hole cutting and interpolation coefficient creation process.

The volume range type is the default value for output nodes created by interactively dropping zone nodes into an output folder.

I Surface, J Surface, K Surface The **I Surface, J Surface** and **K Surface** range types may be used to create a reduced data set for output to a third-party visualization package. When the range type is one of the surface types, two of the **I Range, J Range** and **K Range** sliders will have a minimum and maximum values. The third will have a single value indicating which surface is being viewed.

The **I Surface, J Surface** and **K Surface** range types are also used for performing area integrations. When integrating over the surface area the **Interpolation Type** must be set to **Faces**.

Similarly, when computing viscous quantities such as skin friction or heat transfer, the user must also specify a range type of **Surfaces** and an interpolation type of **Faces**. The surface will determine the appropriate direction for computing these viscous quantities.

I Line, J Line, K Line The **I Line, I Line** and **K Line** range types are most often used to produce line plots or monitor flow profiles. With a line range type, only one of the **I Range, J Range** and **K Range** sliders has a minimum and maximum value. The remaining two have a single value indicating which line is being viewed for output.

Point The **Point** range type is most often used to monitor a point in the solution process. By activating output during the run through the **Output From Solver** option in the **Export** tab, a time or convergence history of a single point can be created. When the **Point** range type is selected, all three of the **I Range, J Range** and **K Range** sliders have a single value indicating which point is being specified.

Multiple Because the **Grid Range** tab can represent a collection of output nodes, the range type for the individual nodes may be different. When all selected nodes have the same range type, the widget will display that single value, and the button color will be normal. However, if the selected nodes have differing range types, the **Multiple** will be selected, and all range types from the selected nodes are colored red. For example, Fig. 17.17 on pg. 357 shows the appropriate display of a collection of nodes which include two surface types (I and J Surfaces).

The discussion above is valid when the mesh is structured (uses I, J, and K logical coordinates). In the case of an unstructured mesh, the range type simplifies to the following: Volume, Surface, Line, Point, and Multiple. The range type is determined and fixed based upon the type of node used to create the file output. If a zone is used to create an output node, then the grid range is set to “Volume” and if a surface is used, the grid range is set to “Surface”. The “Line” and “Point” range types are not used for unstructured grids.

17.5.2 The **Interpolation Type** Selection Widget

Quantities selected for output are naturally defined at particular physical coordinates within the domain. For example, the solution variables (pressure, density, velocity, etc.) are computed and stored at the cell centers. However, grid coordinates are input and stored at the

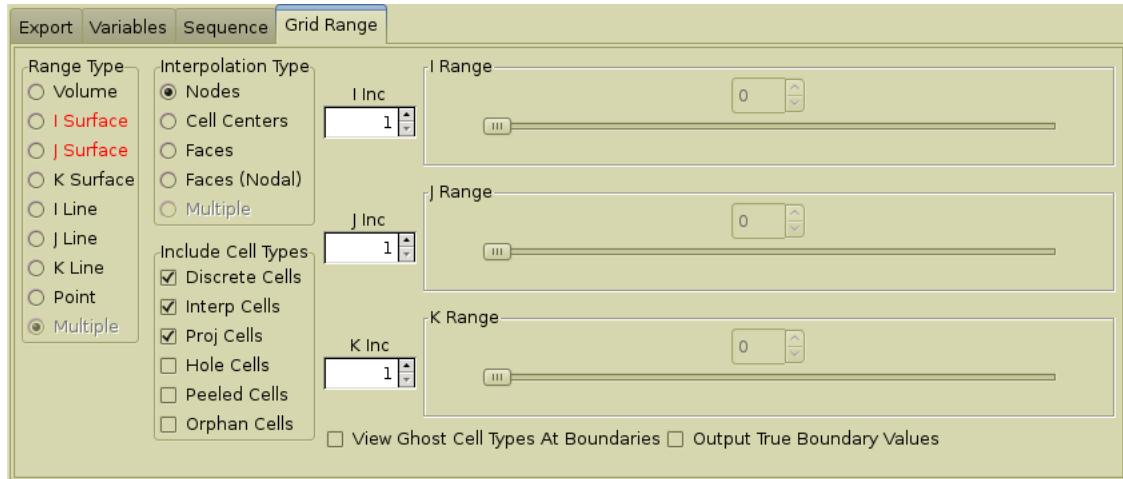


Figure 17.17: The Range Type Widget displaying multiple types.

nodes. Likewise, mass flows, and viscous forces are computed on faces. If the user requires output at a location different from the natural location, interpolation must be performed. The **Interpolation Type** section identifies the interpolation location for the current output node.

Nodes The nodal coordinates are those provided directly from the grid generator to *GASpex*. Because solution variables are stored at cell centers, the cell-centered values must be averaged to the nodes. When the interpolation type is **Nodes**, the solution is averaged from the surrounding cell centers.

Interpolation to nodes is recommended when visualizing a chimera system. When visualizing nodal values with holes or orphans, the blanking information does not need to be averaged. Therefore, interpolation to nodes provides the most accurate representation of hole, peeling, and orphan cells.

Rendering real surfaces is best performed using interpolation to nodes. Because the grid coordinates naturally exist at the nodes, no interpolation of the geometry is necessary. This will yield the most accurate geometric representation.

Cell Centers When the interpolation type is **Cell Centers**, the grid points are interpolated to the cell centers. This point is computed as the centroid of the control volume, which is defined by the surrounding grid points. Selecting the **Line** and **Point** range types in combination with interpolation to cell centers allows the most accurate display of discrete flow-field quantities such as density, pressure, temperature and Mach number.

When a cell-center value is computed at a boundary (minimum or maximum cell-center range), the value is assumed to be on the boundary face. *GASpex* will not output values outside of the grid domain (*i.e.*, ghost cells). Solution data at a boundary is considered located in a “ghost cell” if the flux is split on that boundary. In this case,

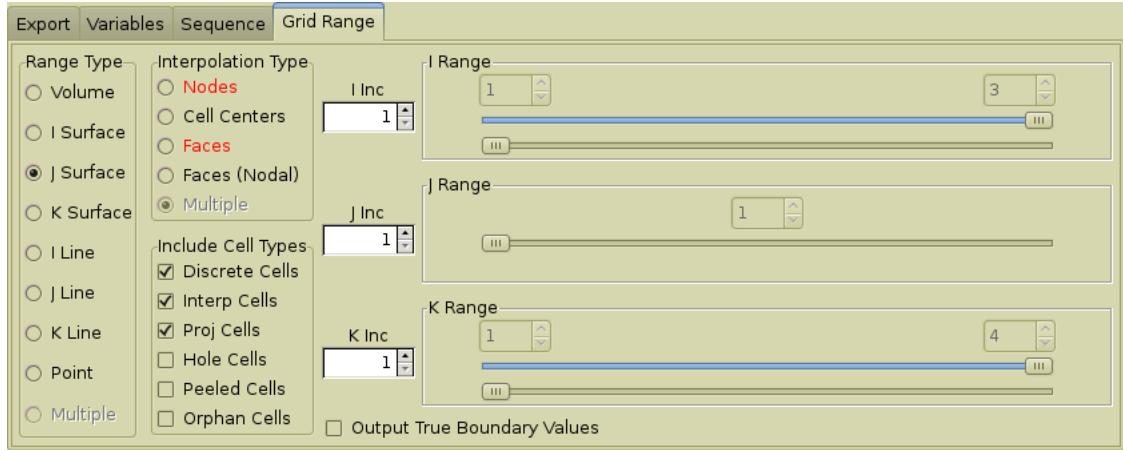


Figure 17.18: The Interpolation Type Widget displaying multiple types.

the first interior data and ghost data are averaged to the cell face. If the boundary is a full-flux boundary, then the actual boundary value is output.

When interpolating blanking information to cell centers, the actual hole cells or orphan cells may appear larger or smaller than in actuality.

Faces Interpolation to faces is most often used in conjunction with output quantities which are related to the evaluation of area based quantities such as a flux (*e.g.*, mass flow, pressure or viscous forces, heat transfer, etc.). Interpolation to faces is only appropriate when the **Range Type** is set to a **Surface**.

Faces (Nodal) In some situations, there may be a need to print or display the face values at the nodal locations on a surface. This can be accomplished using the nodal based face interpolation. The interpolation type must be set to a surface, which is the same constraint for the faces type above.

Multiple Because the **Grid Range** tab may represent a collection of output nodes, the range type for the individual nodes may be different. When all selected nodes have the same range type, the widget will display that single value, and the button color will be normal. However, if the selected nodes have differing range types, **Multiple** will be selected, and all range types from the selected nodes are colored red. For example, Fig. 17.18 on pg. 358 shows the appropriate display of a collection of nodes which include both interpolation to **Nodes** and **Cell Centers**.

17.5.3 The **Include Cell Types** Toggle Buttons

In the case of overset (Chimera) grids, every cell is given a type based on how it used in the grid overlap algorithm. At this time, the use of overset grids is limited to structured grids. For unstructured, all cells are discrete (see below).

The **Include Cell Types** toggle buttons define the cell types that are output to third-party, post-processing packages. Only those points with an active cell type (discrete, interp, projection) will have their blanking value set to true. It is therefore possible to reverse the traditional definition of the blanking field for third-party packages (*e.g.*, the user can force the display of hole cells, and prevent the display of discrete and interpolation cells.) The user can also decide whether or not to include interpolation cells in the display or to treat them as hole cells.

Discrete Cells “Discrete cells” are those cells that are updated by the flow solver. In the absence of overset grids, all points will have the discrete cell type. Toggling the **Discrete Cells** button will enable or disable the display of discrete cells. In general, discrete cells should always be enabled unless the user wishes to identify hole or orphan cells.

Interp Cells “Interp cells” exist next to the surfaces located in the **Chimera ZB** folder, as well as directly adjacent to hole cells assuming that an interpolation stencil can be found for adjacent cells. Toggling **Interp Cells** will enable or disable the display of the interpolation cells in an overset grid system. In general, it is best to include interp cells when visualizing overset grid solutions. Otherwise visual gaps may appear in the solution near the hole cells because the discrete cells may not completely overlap. Interp cells are updated through the passing of information from overlapping grids.

Proj Cells “Proj Cells” are created near surfaces where surface projection occurs – most often near collar grids about highly curved surfaces. Toggling **Proj Cells** will enable or disable the display of the projection cells in an overset grid system.

Hole Cells “Hole Cells” are created during the hole cutting process in the overset grid definition. (Note: hole cells are also created during the peeling process.) Toggling **Hole Cells** will enable or disable the display of the hole cells in an overset grid system. The solution variables associated with hole cells are invalid.

Peeling Cells “Peeled Cells” are created during the hole interpolation coefficient stencil selection process in the overset grid definition. Peeling cells are nothing more than hole cells. Toggling **Peeled Cells** will enable or disable the display of the peeled cells in an overset-grid system. See Sec. 9.4.4 on pg. 137 for more information on peeling.

Orphan Cells “Orphan Cells” are created during the hole interpolation coefficient stencil selection process in the overset grid definition. Orphan cells represent candidate interpolation cells for which no interpolation stencil can be found. Toggling **Orphan Cells** will enable or disable the display of the orphan cells in an overset grid system.

Because the **Grid Range** tab may represent a collection of output nodes, the range type for the individual nodes may be different. When all selected nodes have the same cell display types, the widget will display the toggle buttons with default colors. If any of the different



Figure 17.19: The **Include Cell Types** displaying multiple types.

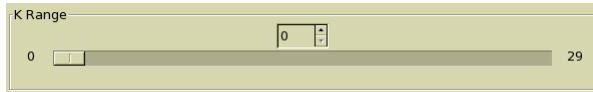


Figure 17.20: The Range Widget in single slider mode

cell type display flags have differing values for the merged values, then the button will be colored red. For example, Fig. 17.19 on pg. 360 shows the appropriate display of a collection of nodes which have different values for the display of **Interp Cells** and **Hole Cells**, but identical values for **Discrete Cells** and **Orphan Cells**.

17.5.4 The I, J and K Range Widgets

The **I Range**, **J Range** and **K Range** sliders, allow the user to view and modify the limits of the logical coordinate directions for structured grids. The range sliders are multi-function, in that they either represent a single value (such as for describing a point), or a minimum and maximum value (such as for describing the limits of a coordinate line).

Single Slider Mode

The range widget will be in single-slider mode whenever there is a single value for the current coordinate direction. For example, the **I Range** widget will be in single-slider mode whenever the **Range Type** is set to **I Line**, **J Line**, **K Line**, or **Point** as shown in Fig. 17.20 on pg. 360.

The absolute minimum and maximum limits of the selection range are displayed to the left and right of the slider widget respectively. In the current example, the minimum value is 1, and the maximum value is 29. The thumb bar on the slider is positioned relative to the current value of the range. The current value of the selection is displayed in a type-in widget at the top center of the widget. For the current example, the current value is 0. The user may adjust the value either by positioning the slider, or by directly entering the desired value into the type-in.

Multiple Slider Mode

The range widget will be in multiple slider mode whenever there is a range of values being represented in the current coordinate direction. For example, the **I Range** widget will be

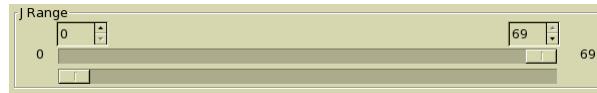


Figure 17.21: The Range Widget in multiple slider mode

in multiple slider mode whenever the Sec. 17.5.1 on pg. 355 is set to **Volume, J Surface, K Surface, or I Line**, as shown in Fig. 17.21 on pg. 361.

The absolute minimum and maximum limits of the selection range are displayed to the left and right of the slider widget respectively. In the current example, the minimum value is 0, and the maximum value is 69. The bottom slider will display and edit the minimum value for the current range, while the top slider will display and edit the maximum value. The thumb bar on the slider is positioned relative to the current value of the limit. The current minimum value of the range is displayed in a type-in widget located at the top left of the slider widget, while the current maximum value of the range is displayed in a type-in widget located at the top right of the slider widget. The user may adjust either value by positioning the slider, or by directly entering the desired value into the type-in. Note that if the minimum is set to a value larger than the maximum, then the maximum value will be adjusted so that it is greater than or equal to the minimum. Likewise the minimum value is adjusted such that it will remain less than or equal to the maximum value when the maximum value is adjusted.

Behavior of the Range Widgets When Viewing Multiple Nodes

Because the **Grid Range** tab may represent a collection of output nodes, the Range widgets for the individual nodes may be different. When all selected nodes have the same range type, the widget will display that single value, and the button color will be normal. However, if the selected nodes have differing range types, there are two different situations. There are three conditions which must be met before a range widget will allow simultaneous editing of multiple ranges. First, the slider mode must be equivalent for all ranges. In other words, it is not possible to simultaneously edit the ranges when one is displayed using Single Slider Mode, while the other is displayed using Multiple slider mode. The final two conditions are that the absolute minimums and absolute maximums must be equivalent. Editing a range with absolute limits of 0-10 simultaneously with a range with absolute limits of 1-12 will not be permitted. If any of these three conditions are not met, then the range widget will be disabled, and the user will not be allowed to change the values. If the previous three conditions are met, but the actual current values of the ranges are different, then the range widget will allow the user to edit the values, and the label of the range will be colored red, to indicate that the values are different, as shown in Fig. 17.22 on pg. 362.

Note that the merging behavior for the I, J and K ranges will be treated independently. In other words, the I range may have identical values, the J Range may be different, and non-editable, while the K Range may be different and editable.

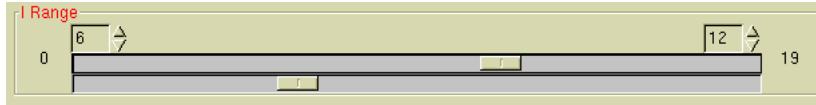


Figure 17.22: The Range Widget in multiple slider mode, displaying multiple ranges with different values.

Absolute Range Limits Based Upon Interpolation Type

The absolute minimum and maximum values for a given range will depend upon the grid dimensions of the selected output node as well as the interpolation type.

Nodes When the interpolation type is **Nodes**, then the minimum values for all three logical directions will be equal to one. The maximum value will be set to the grid dimensions. Absolute minimum and maximum values will correspond to the boundary values of the current zone. For example, if viewing **I Surfaces**, with **Nodes**, setting the **I Range** equals 1 will correspond to viewing the I0 boundary values, interpolated to the nodes.

Cell Centers When the interpolation type is **Cell Centers**, then the minimum values for all three logical directions will be equal to zero. The maximum value will be set to the grid dimensions. Absolute minimum and maximum values will correspond to the boundary values of the current zone. For example, if viewing **I Surfaces**, with **Cell Centers**, setting the **I Range** equal to 0 will correspond to viewing the I0 boundary values.

Faces When the interpolation type is **Faces**, then the minimum values for all three logical directions will be equal to one. The maximum value will depend upon which Surface direction is selected in the Sec. 17.5.1 on pg. 355. If the **J Surface** is selected, then the maximum limits for the I and K directions will be one less than the grid dimension (which represents the total number of faces in that direction), while the J direction maximum will be equal to the grid dimension. The **I Surface** and **K Surface** will have similar limits.

17.5.5 Incremental Viewing

For structured grids, the user may wish to increment which grid lines are displayed or printed. This can be done using the **I Inc**, **J Inc**, and **K Inc** inputs. The default value is unity which corresponds to every grid line displayed. A value of 2 will attempt to display or print every other grid line in that coordinate direction. The use of increments can be useful to reduce the amount of data printed to file or help increase rendering speeds within the GUI.

17.5.6 Viewing of Boundary Ghost Cells

When overlapping grids are used, the user may select a number of surfaces to be Chimera. When this occurs, the default behavior is to use the ghost cells on a surface as the interp

cells. The ghost cells may also become hole cells or peeled cells during the Chimera setup process. Since ghost cells are not displayed in the visualization, the user may select the **View Ghost Cell Types At Boundaries** as a way of viewing the cell type along a surface boundary. When this option is selected, the display of cells along a boundary take on the ghost cell type. This is only for visualization or output purposes and will not change the actual cell type along a boundary.

17.5.7 Output of True Boundary Values

The **Output True Boundary Values** option allows the user to override the handling of boundary values. In *GASPer*, a boundary value is stored at either the boundary face or the ghost cell. The ghost cell is simply a mirror reflection of the adjacent interior cell and is located outside the computational domain. Because of this, *GASPer* averages a boundary value located at a ghost cell with the first interior cell to attain the value at the boundary face. The **Output True Boundary Values** allows the user to override ghost cell averaging, such that no averaging is performed, regardless of the boundary value location. This option is only available for interpolation types of either cell centers or faces.

The location of a boundary value is determined from the boundary condition. For example, solid-wall boundary conditions store values at the boundary face, while inflow or outflow boundary conditions store values at the ghost cell. Under normal output conditions, the averaging of the ghost cell value with the interior value will represent the correct output value. But there are times when the user wishes to know the exact values at a boundary being enforced. An instance when this is true is when the user is verifying pointwise boundary values. To do so, the user should select the **Output True Boundary Values** option and output either face or cell center locations for the boundary surface. If output is at cell centers, the 0 index should be selected. In this way the output should be identical to the pointwise data supplied in the physical model.

Chapter 18

Database GUI

The *DBASEMGR* graphical user interface (GUI) is a stand-alone utility for creating, editing, and examining chemistry models, species and reactions within the *GASPEX* thermodynamics and chemistry database file. A default database file (`thermochemdb.tcd`) comes with *GASPEX* which contains a number of gas, liquid and solid models. The user can edit this file or create their own database file for use.

Upon starting the primary *GASPEX* GUI from scratch, the default database file is searched for and loaded (file `$(AEROSOFT_HOME)/share/thermochemdb.tcd`). From within the primary GUI the user can then change the location or file name as desired. Every case requires a database file in order to set up and run the *GASPEX* solver.

18.1 Displaying the GUI

The *DBASEMGR* is run using the same *GASPEX* binary as the primary GUI and solver, but with the following arguments:

```
gaspex -dbasemgr
```

or with the shorter argument:

```
gaspex -d
```

In order to run the *DBASEMGR* and open the database file at the same time, use the `-i` option as in the following example.

```
gaspex -d -i thermochemdb.tcd
```

where the file `thermochemdb.tcd` is the default database file that comes with *GASPEX*. A second option is to simply run the *GASPEX* binary with the file name as a single argument.

gaspex thermochemdb.tcd

where the binary will detect that the file is a thermo-chemical database file based upon the tcd suffix, and the file formatting itself.

The database GUI can also be displayed from the primary *GASPEX* GUI. More details are located in Sec. 5.1 on pg. 69).

18.2 Primary GUI Window

18.2.1 File Menu Bar

The **File** pull-down menu provides the capability to open an existing database file, save an edited copy of a database file, or quit the *DBASEMGR*. The full list of options and its use is given below.

File→New: The new menu item is used to start a new database. All the values in the database are set to their default values and any existing data is lost.

File→Open...: The open menu item allows the user to open a previously created database file. A file may also be opened when launching the GUI using the -i option.

File→Save: The save menu item is used to save the .tcd file. If the database is new (never been saved before), then the user will be prompted for a name. Otherwise, the database will be saved to the file name already in use.

File→Save As...: The save as menu item is used to save the database file with a name specified by the user.

File→Close: The close menu item allows the user to quit an open database without exiting the GUI.

File→Quit: The quit menu item allows the user to quit the database GUI. Data is not saved with this option, so be sure to save changes to the database before selecting quit.

18.2.2 Options Menu Bar

The options menu bar has several import and export features from which to select. The first is **Options→Export Selected**. This allows the user to export any selected data in the tree view to a stand-alone database file. The exported data can range from a single species to the entire database. Simply select one or more folders or data nodes in the tree view and select the **Export Selected** option. If a reaction or model is selected, the export will also ensure that any species associated with the data are also exported. The tree structure for any exported data will also be preserved.

If selected data are exported to an XML database file, it can be used as a database file for the solver or it can be imported into another database using the **Options→Import & Merge** feature. This allows an existing database to import and merge data from another database file. When using this feature, the following rules apply.

- If any imported data (species, reactions, models) is found to already exist in the database, then the duplicate imported data is skipped. In other words, if the data is identical, then nothing changes.
- Species must have unique names. If the existing database has a specie with the same name as an imported specie, then the data will be merged. The new data will replace any existing data. If the imported species does not already exist, it will be added based on the tree structure of the imported data (same location as the database it was exported from).
- Reactions do not have to be unique. If a reaction is imported that is located in a different tree location (parent folders are different), then the reaction is added to the existing database. Reactions are merged only if both reactions are located in the same tree location and have the same rate types (forward and equilibrium reaction rates are the same).
- Models do not have to be unique. Model data is merged only if the models are located in the same tree location. Otherwise the model is added to the existing database.

The next feature is **Options→Import LERC Curve Fit Data**. This is intended to update the Lewis Research Center (LERC) curve fit data for gas species data. Options for curve fit data include thermodynamic data for enthalpy and specific heat, as well as laminar viscosity and thermal conductivity. Selecting this option will allow the user to browse curve fit data files for selection and update. Only those species selected in the tree view will be updated.

The last option is **Options→Import Chemkin Data**. This supports import of species and reactions from Chemkin data files. Selecting this option brings up a dialog that allows the user to select the Chemkin file and options for import. For importing species, a new species is created using default data if the species does not already exist in the database. At this time only the Gordon/McBride curve fit data can be imported from the Chemkin file. Data from the Chemkin file is assumed to be in the correct units for the *GASPE* database (unit checking is not done at this time).

18.2.3 Tree View Frame

The tree view, Fig. 18.1 on pg. 368, located on the left side of the main window provides an hierarchical representation of all of the elements of the database which may be graphically viewed. All objects within the tree view must reside beneath the main folder “All Database Objects”.

Techniques for navigating and editing the tree view contents for the database GUI are identical to those of the primary *GASPE* GUI. See Sec. 7.3.3 on pg. 84 for more information.

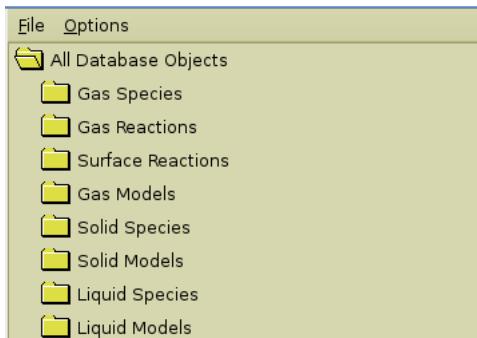


Figure 18.1: The *DBASEMGR* tree view.

18.2.4 Data View Frame

The right side of the database GUI window is reserved for displaying data. The type of data shown in the data view will depend on what is selected in the tree view. For example, selecting a species node in the tree view will cause the data view frame to render all the information associated with that species.

18.3 Species

In *GASPex*, a species is a generalized classification for atoms, molecules, molecular fragments, or ions which are independently accounted for within a particular solver. Species, along with reactions, are the basic building blocks of a chemistry model. Species are further organized by physical phase, whether they are gas, liquid, or solid. A separate species may exist to define a given atom or molecule in the three distinct phases. For example, a gas species for water describes the properties of water as a vapor only, while a liquid species for water describes the properties of water as a liquid.

The current version of *GASPex* includes over 50 species. While most of these are gas species, there are solid and liquid species as well.

Species are created in one of three ways; import, duplication, or creation of a new species. Species may be imported using the **Options→Import & Merge** feature. Assuming the thermochemistry file selected for import and merge contains new species, they will be added to the existing hierarchy in the same relative position that they maintain in the imported file. Species may be duplicated by first selecting one or more species nodes within the tree view. Pressing the right mouse button over the selected nodes will display a pop-up menu, one of whose sub-menus is **Edit**. Within the **Edit** menu, selecting the **Copy** button will duplicate the selected items, and save them in an internal clipboard. Selecting an appropriate parent folder, and depressing the right mouse button, the user can then select **Paste** from the **Edit** sub-menu, in order to duplicate the copied elements.

Finally, new species may be created by selecting a new parent folder, pressing the right mouse button, and selecting an appropriate new species type from within the **New Species**

sub-menu. When creating a new gas species, the user has the following two options.

Create From Default This creates a new species based on default values. The user will need to insert and set up data features.

Create From a Mixture This creates a new species from a mixture of existing species.

Selecting this option will result in a pop-up dialog, allowing the user to build a table of species with corresponding mass fractions. The user can plot a comparison of the data (if Gnuplot is supported) to visually confirm the new species data. Features are created for viscosity, thermal conductivity, and Gordon/McBride Thermodynamic data when all the mixture species contain the corresponding data.

Note: Remember that a folder must be selected in order to paste a new species. This is also true if creating a new species using the edit menu.

18.3.1 Gas Species Main Frame

Selecting a single gas species node will display the gas species **Main** frame as seen in Fig. 18.2 on pg. 370. The gas species main frame contains several inputs such as molecular weight, a comment field, and a table listing all the available species data models. The gas species node should be labeled using either the specie's molecular formula or the species name.

The first option in this frame concerns species parsing. Parsing is the process in which the atomic content of a species is determined. This is beneficial for verifying the mass balance of reactions and automatically computing the molecular weight. Species parsing can only be performed if the molecular formula is given for the node name.

Note that if the species is entered as a molecular formula, each element of the molecule must begin with a capital letter (*e.g.*, NaCl). No blank spaces are allowed in the species name. The number of atoms in the molecule must follow the individual element, and the charge (if applicable) may follow either the element name or the molecular formula. All characters between capital letters are assumed part of an element group. When elements in the molecular formula are recognized as members of the periodic table, the molecular weight of the molecule may be computed using the **Calculate** button. As examples, H_2O and O_2^+ are written as H₂O and O₂⁺.

If the name of the species is given rather than the molecular formula, the user is free to name the species as desired. In this case the user must enter the molecular weight by hand and species parsing will not be available. Only molecular formulas (and not species names) may be used in chemical reactions.

The **Heat of formation/R** input represents the species heat of formation divided by the species gas constant, $h_{f_i}^0/R_i$. The heat of formation should be given at the standard state of $T = 298.15\text{ K}$ and $p = 1\text{ atm}$. Recall that the heat of formation for any element in its stable form at STP conditions is zero (*e.g.*, $h_{f_i}^0 = 0$ for O_2). From general chemistry, species with a positive heat of formation are generally less stable and more reactive than species with a

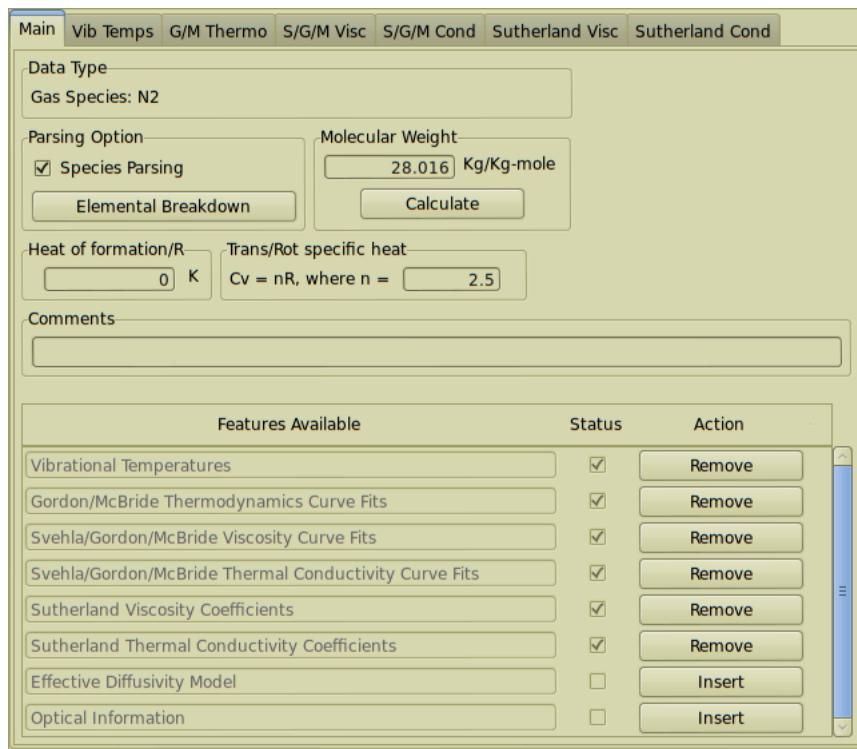


Figure 18.2: The main interface frame of a gas species.

negative heat of formation. This input is important for chemically reacting flows or flows with phase changes.

The **Trans/Rot specific heat** input represents the specific heat contribution from translational and rotational energy normalized by the species gas constant (*i.e.*, supply n where $n = (\tilde{c}_{v_{tr}} + \tilde{c}_{v_{rot}})/R_i$). In *GASPE*, all molecules are considered in translational equilibrium, so the translational contribution ($\tilde{c}_{v_{tr}}/R_i$) is always 3/2. The rotational contribution depends on the type (*i.e.*, monatomic, diatomic or polyatomic) and linearity of the molecule. This implies three choices for n :

$$n = \begin{cases} 3/2, & \text{for monatomic species,} \\ 5/2, & \text{for diatomic and linear polyatomic molecules,} \\ 6/2, & \text{for non-linear polyatomic molecules.} \end{cases}$$

This input is required if using the equilibrium translation and rotation thermodynamic option (see Sec. 11.2.6 on pg. 168).

A **Comments** section is provided for documentation purposes. Comments are stored in the database file and serve as personal notes for the species.

Below the comments input is a table listing all the available features for the species. Each available feature will have a description, followed by a **Status** flag. If checked, then the data is available for the selected species. Finally, there is an action button for each feature. Pressing the button for a feature will either insert or remove the feature from the species

data. Removing a feature will result in a loss of data associated with the feature. Therefore, removing a feature and then inserting it again will serve to re-initialize the data. Some features are required for a given species. Required features may not be removed. In this case the label on the **Action** button will say “Permanent”, and the button will be disabled. There will be a notebook frame under an appropriate heading for each available feature which is used to display and modify the relevant data.

18.3.2 Gas Species Summary

Below is a listing of the possible inputs for a gas species. Not all, but most inputs are connected to a modeling option in the *GASpex* GUI. Note that to use a modeling option within the *GASpex* flow solver, the data must be available for all the species in the selected gas model.

Vibrational Temperatures Located in the “Vib Temps” frame and representing the vibrational contributions to the internal energy. Inputs consist of the characteristic temperatures of vibration. This option is required if using the non-equilibrium vibrational thermodynamic option (see Sec. 11.2.6 on pg. 168).

Gordon/McBride Thermodynamics Curve Fits Located in the “G/M Thermo” frame and representing curve fits for specific heat, enthalpy, and entropy. The curve fits are a function of temperature and support up to four temperature ranges. This option is required if using the Gordon/McBride thermodynamic option (see Sec. 11.2.6 on pg. 168).

Viscosity: Svehla/Gordon/McBride Curve Fits Located in the “Svehla/Gordon/McBride” frame under Viscosity and representing a NASA Lewis curve fit for laminar viscosity. This input is required if using the Svehla/Gordon/McBride viscosity model for viscous flows (see Sec. 11.4 on pg. 181).

Conductivity: Svehla/Gordon/McBride Curve Fits Located in the “Svehla/Gordon/McBride” frame under Conductivity and representing a NASA Lewis curve fit for thermal conductivity. This input is required if using the Svehla/Gordon/McBride conductivity model for viscous flows (see Sec. 11.4 on pg. 181).

Viscosity: Sutherland Coefficients Located in the “Sutherland” frame under Viscosity and representing a curve fit for laminar viscosity. The Sutherland model is a default and should always exist for a species in the chemical database if solving viscous flows.

Conductivity: Sutherland Coefficients Located in the “Sutherland” frame under Conductivity and representing a curve fit for thermal conductivity. This input is required if using the Sutherland conductivity model for viscous flows (see Sec. 11.4 on pg. 181).

Lennard-Jones Data Located in the “Lennard-Jones” frame and representing data for several models. The Lennard-Jones data is used for an effective molecular diffusion model (see Sec. 11.4 on pg. 181), as well as collision integrals used for viscosity, conductivity, and diffusion.

Optical Information Located in the “Optics” frame and representing inputs for simulations involving radiation (*i.e.*, lasers). The Gladstone-Dale constant is the primary input and is required to compute the optical path difference.

Energy Level Splitting Data Located in the “Energy Levels” frame and representing inputs for simulations involving radiation (*i.e.*, lasers). Supports band-to-band emission and absorption modeling for gas or solid state lasers.

Gas Species Vib Temps Frame

The Vib Temps frame represents the vibrational contributions to the internal energy for diatomic and polyatomic species (monatomic species have no vibrational mode). Inputs consist of the characteristic temperatures of vibration. This option is required if using the non-equilibrium vibrational thermodynamic option (see Sec. 11.2.6 on pg. 168) and is added to the species data using the feature table located in the species main frame.

To discuss the input of the vibrational temperature data, take the gas species C_2N_2 for example. Using statistical mechanics, the internal energy associated with vibrational equilibrium can be calculated as a function of the translational temperatures, the species gas constant, and the characteristic temperatures of vibration. The JANAF Thermochemical Tables provide vibrational frequencies which can be converted to characteristic temperatures according to

$$\Theta_v = \frac{h(\bar{c}\nu^{(\text{JANAF})})}{k}$$

where $h = 6.626176 \times 10^{-34} J \cdot s$, $\bar{c} = 299,792,458 m/s$ and $k = 1.380662 \times 10^{-23} J/K$. The vibrational frequencies in the JANAF tables correspond to the following characteristic temperatures of vibration:

ν, cm^{-1}		Θ_v, K	N_v
240.	\Rightarrow	345.3087	(2)
507.2	\Rightarrow	729.7524	(2)
850.6	\Rightarrow	1223.832	(1)
2149.	\Rightarrow	3091.952	(1)
2328.5	\Rightarrow	3350.214	(1)

After inserting the vibrational temperature feature (recall this is done from the Main frame), select the **New** button. This will cause a pop-up window to appear with options on how to set up the data. The user may either import the vibrational data from an external file or enter the data by hand. If entering the data by file, the `vibtemp.inp` ASCII file should exist. The file should contain the vibrational temperatures, one per line.

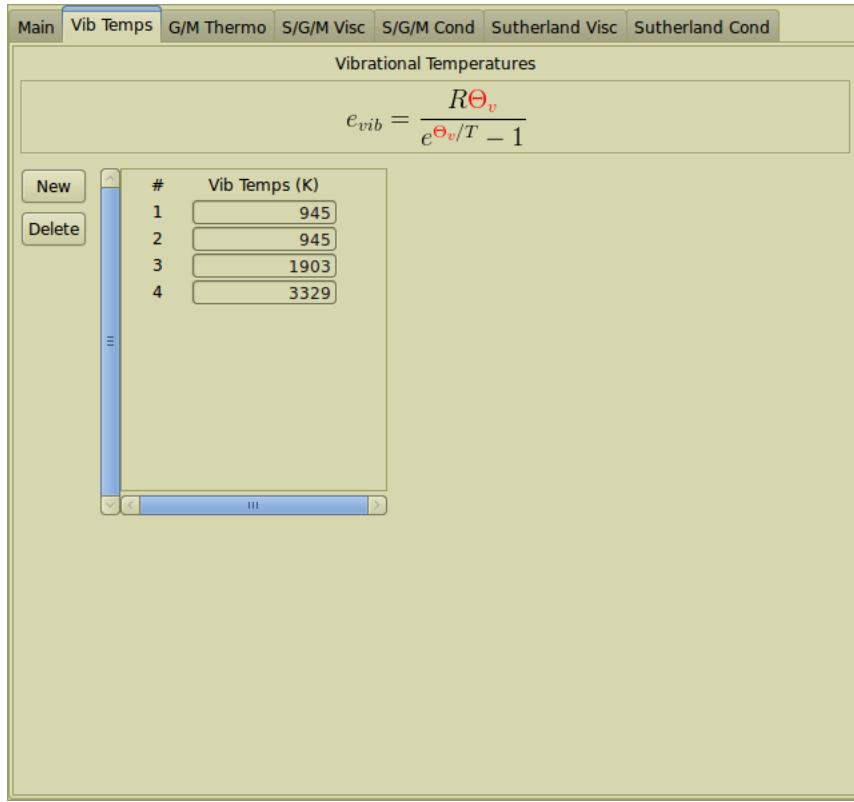


Figure 18.3: The Vib Temps gas species frame for C_2N_2 .

If entering the data by hand, the user must first enter the number of vibrational temperatures (which is 7 for C_2N_2) and press **Type in Data**. The first characteristic temperature 345.3087 can then be entered (type in the value and press return). After entering the remaining six values, the vibrational temperature should appear as Fig. 18.3 on pg. 373.

Gas Species G/M Thermo Frame

The G/M Thermo frame, as seen in Fig. 18.4 on pg. 374, provides display and editing of Gordon and McBride Thermodynamic curve fits for specific heat, enthalpy, and entropy. Energy modes are modeled implicitly through the curve-fit data as a function of temperature. The curve fits are a function of temperature and support up to four temperature ranges.

The letters G and M stand for Gordon and McBride, authors from the original papers that presented the curve fits. The curve fits come from what is now called the NASA Glenn Chemical Equilibrium Application Program (CEA). More information on this program can be found at <http://www.grc.nasa.gov/WWW/CEAWeb>.

The curve fits should contain all of the forms of internal energy since they are compiled from experimental data. Be aware that the Gordon/McBride curve fits are valid over a limited range of temperatures (normally 200 K to 6,000 K), and *GASPEX* will *not* stop execution when a local flow-field temperature exceeds the range of validity. The actual

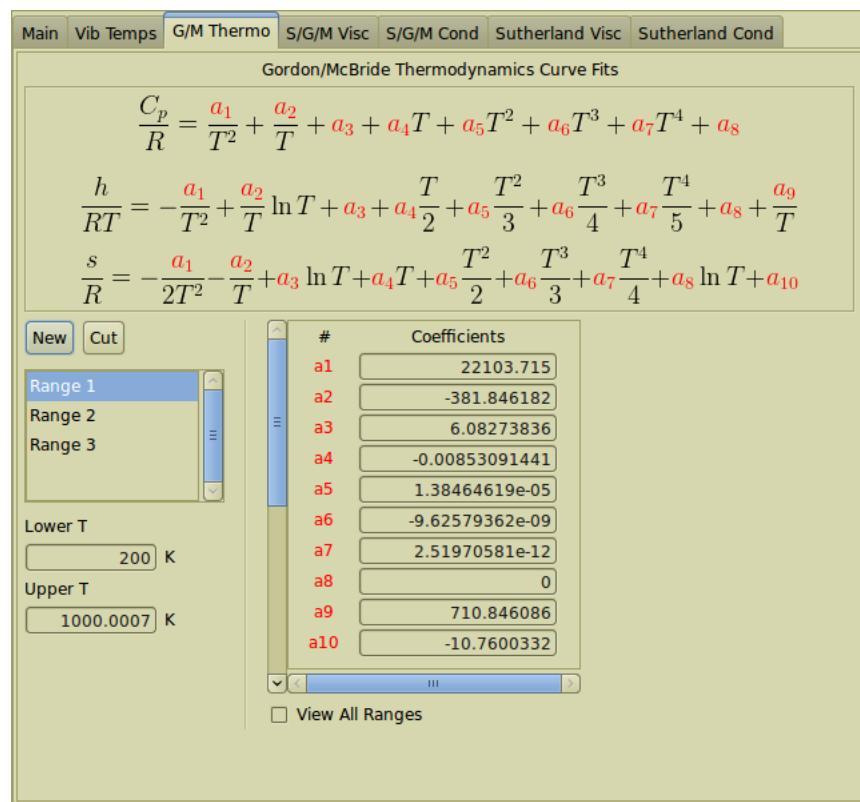


Figure 18.4: The Gordon and McBride Thermodynamics Curve Fits frame.

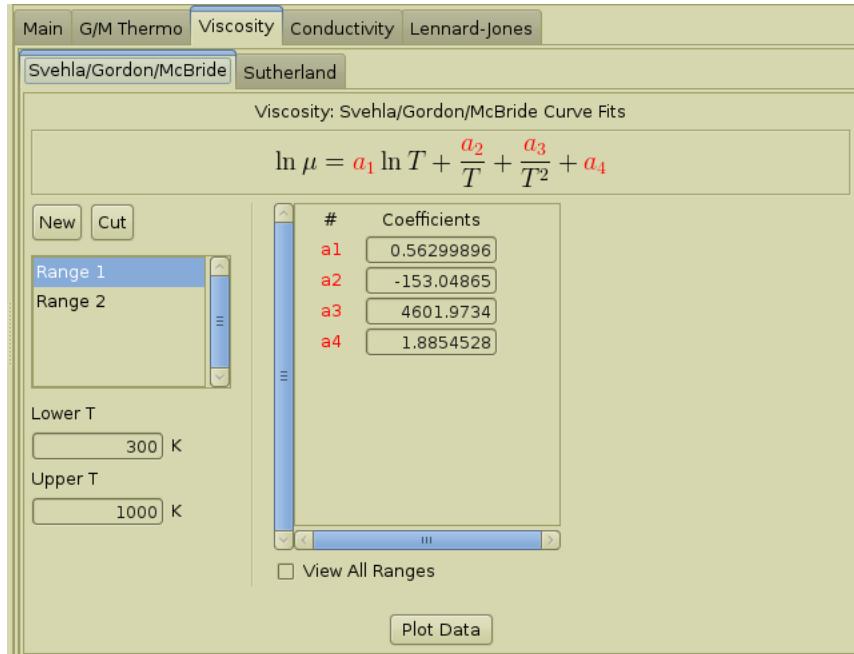


Figure 18.5: The Svehla, Gordon and McBride Viscosity Curve Fit frame.

temperature range is defined by the user in the database manager.

This feature is required if using the Gordon/McBride thermodynamic option (see Sec. 11.2.6 on pg. 168) and is added to the species data using the feature table located in the species main frame.

Gas Species Viscosity Svehla/Gordon/McBride Frame

The Svehla/Gordon/McBride frame, as seen in Fig. 18.5 on pg. 375, represents a NASA Lewis curve fit for viscosity. The names Svehla, Gordon and McBride represent authors from the original papers that introduced the curve fits.

This input is required if using the Svehla/Gordon/McBride viscosity model for viscous flows in the *GASPE* Navier-Stokes physical model (see Sec. 11.4 on pg. 181).

A **Plot Data** button is available to show the curve fit data. In order to use this feature, the Gnuplot software must be available on the computer system.

Gas Species Viscosity Sutherland Frame

The Sutherland frame under the Viscosity section, as seen in Fig. 18.6 on pg. 376, represents a curve fit for laminar viscosity. The curve fit is based on Sutherland's formula which is displayed in the frame.

The units for the Sutherland formula must be given consistently with the original reference. The Sutherland curve fit is consistent with the data presented in White [30] where

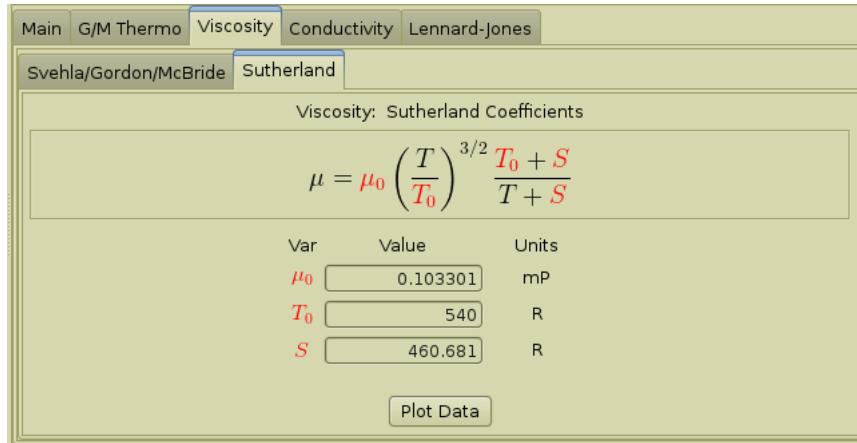


Figure 18.6: The Sutherland Viscosity Curve Fits frame.

viscosity is given in milliPoise and the temperature constants (T_0 and S) are given in Rankine.

If Sutherland data is not known for a gas species, the user may determine the inputs using a method attributed to Pitalo [60]. The method requires three data points for viscosity which can then be used to determine the best values for the Sutherland formula.

The Sutherland viscosity model is the default viscosity model in the Navier-Stokes physical model. Because of this, it should always be supported for a species in the chemical database if solving viscous flows.

A **Plot Data** button is available to show the curve fit data. In order to use this feature, the Gnuplot software must be available on the computer system.

Gas Species Conductivity Svehla/Gordon/McBride Frame

The Svehla/Gordon/McBride conductivity frame, as seen in Fig. 18.7 on pg. 377, represents a NASA Lewis curve fit for thermal conductivity. The names Svehla, Gordon and McBride represent authors from the original papers that introduced the curve fit.

This input is required if using the Svehla/Gordon/McBride conductivity model for viscous flows (see Sec. 11.4 on pg. 181).

A **Plot Data** button is available to show the curve fit data. In order to use this feature, the Gnuplot software must be available on the computer system.

Gas Species Conductivity Sutherland Frame

The Sutherland frame under the Conductivity section, as seen in Fig. 18.8 on pg. 377, represents a curve fit for thermal conductivity.

The units in for the Sutherland formula must be given consistently with the original reference. The Sutherland curve fit is consistent with the data presented in White [30]

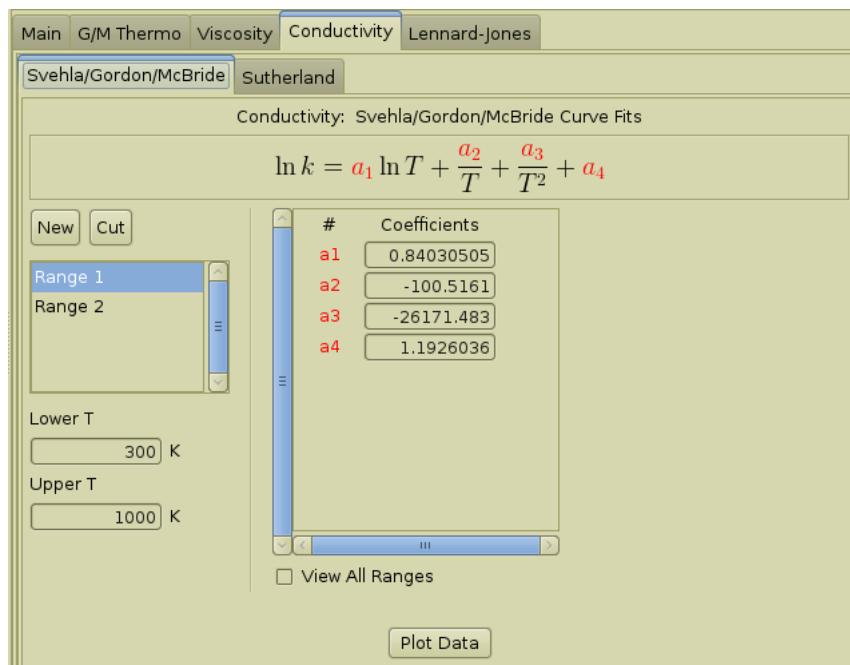


Figure 18.7: The Svehla, Gordon and McBride Thermal Conductivity Curve Fit frame.

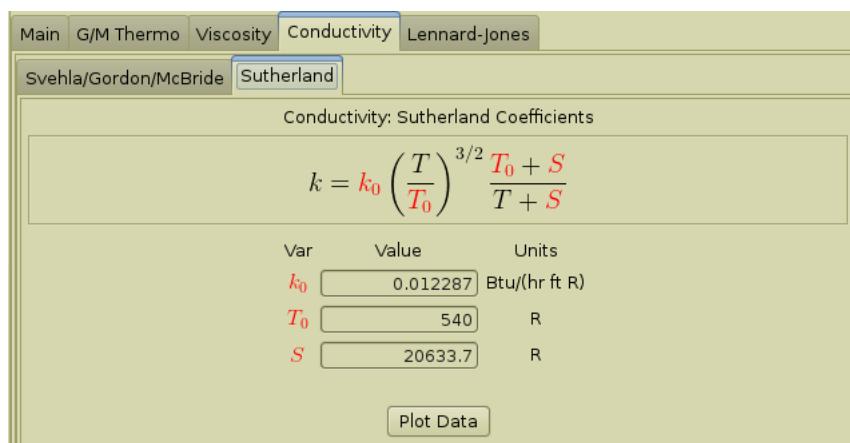


Figure 18.8: The Sutherland Thermal Conductivity Curve Fit frame.

where thermal conductivity is in $Btu/(hr \cdot ft^{\circ}R)$ and the temperature constants (T_0 and S) are given in Rankine.

This input is required if using the Sutherland conductivity model for viscous flows (see Sec. 11.4 on pg. 181).

A **Plot Data** button is available to show the curve fit data. In order to use this feature, the Gnuplot software must be available on the computer system.

Lennard-Jones Frame

The Lennard-Jones frame is used to input the molecular collision diameter and the molecular energy of interaction. Once available, the data can be used with the effective diffusivity model and collision integrals. The collision integrals can be used to model viscosity, conductivity, and diffusion.

The effective diffusivity model represents a model for molecular diffusion. This model accounts for individual species diffusion coefficients and is required if using the Effective or Effective w/Pressure diffusivity models for viscous flows (see Sec. 11.4 on pg. 181). Details of the effective diffusion model can be found in Eppard *et al.* [61].

Collision integrals are based on either Gupta [31] or a formulation based on Lennard-Jones data inserted in this frame. The option for collision integrals is appropriate for high-speed flows with non-equilibrium effects. See Sec. 11.4.5 on pg. 186 for selecting collision integrals in the primary *GASPE* GUI.

Gas Species Optics Frame

The Optics frame, as seen in Fig. 18.10 on pg. 380, represents inputs for simulations involving radiation (*i.e.*, lasers). The Gladstone-Dale constant is the primary input and is required to compute the optical path difference.

Gas Species Energy Levels

The energy level frame is used for simulations involving radiation (*i.e.*, lasers). The data in this frame enables band-to-band emission and absorption for gas lasers. The required data comes from the relationship relating the emission cross section to the absorption cross section. The required data includes the energy for each sub level (ei) along with the corresponding degeneracy value (gi). The number of sub-levels or splittings at each band will vary depending on the system. For each species involved in the emission or absorption transitions, energy level data must be provided.

18.3.3 Solid Species Main Frame

Selecting a single solid species node will display the solid species **Main** frame as seen in Fig. 18.11 on pg. 381. The main frame for solid species contains several inputs such as molecular weight, a comment field, and a table listing all the available species data models.

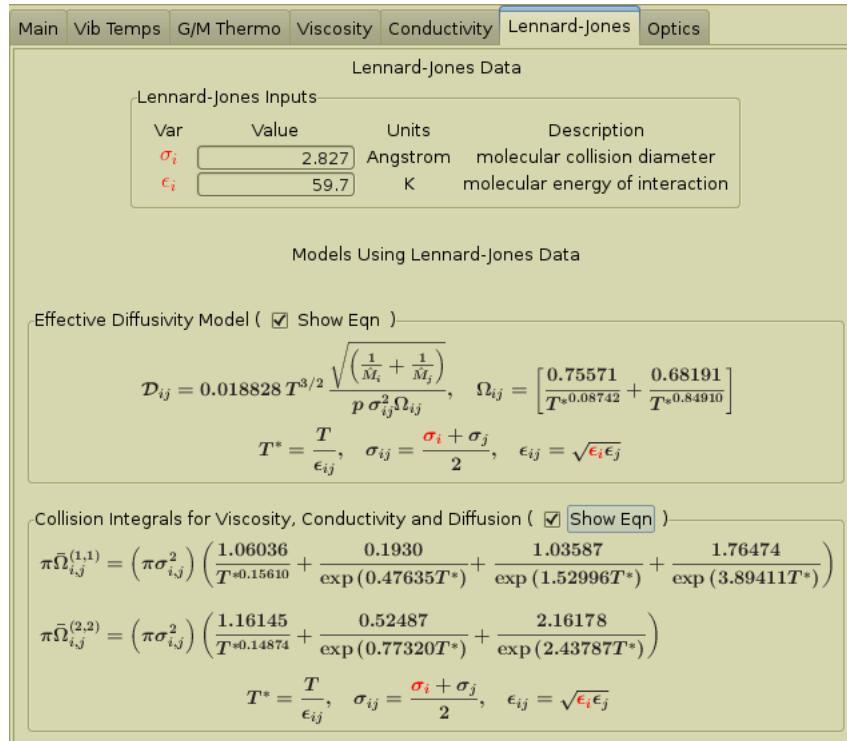


Figure 18.9: The Lennard-Jones data frame.

The solid species node should be labeled using either the specie's molecular formula or the species name.

The first option in this frame concerns species parsing. Parsing is the process in which the atomic content of a species is determined. This is beneficial for verifying the mass balance of reactions and automatically computing the molecular weight. Species parsing can only be performed if the molecular formula is given for the node name.

Note that if the species is entered as a molecular formula, each element of the molecule must begin with a capital letter (*e.g.*, NaCl). No blank spaces are allowed in the species name. The number of atoms in the molecule must follow the individual element, and the charge (if applicable) may follow either the element name or the molecular formula. All characters between capital letters are assumed part of an element group. When elements in the molecular formula are recognized as members of the periodic table, the molecular weight of the molecule may be computed using the **Calculate** button. As examples, H₂O and O₂⁺ are written as H2O and O2+.

If the name of the species is given rather than the molecular formula, the user is free to name the species as desired. In this case the user must enter the molecular weight by hand and species parsing will not be available. Only molecular formulas (and not species names) may be used in chemical reactions.

The value of **Emissivity** will be used when a radiation boundary condition is applied within the solid thermodynamics (ST) solver. Its value represents the ability of the solid



Figure 18.10: The Gas Species Optics frame.

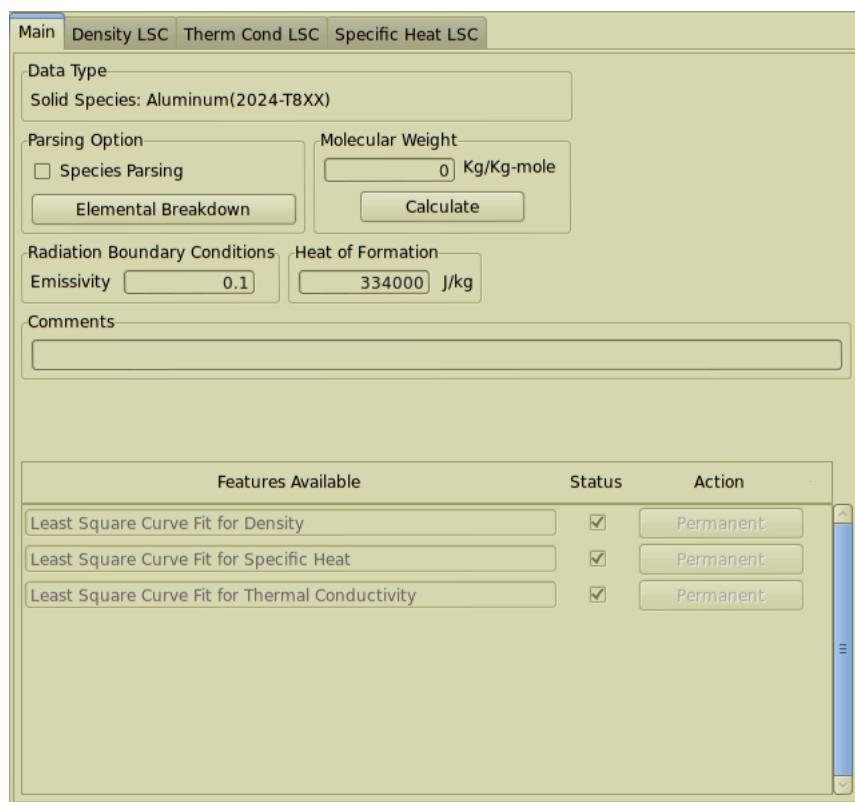


Figure 18.11: The main interface frame of a solid species.

surface to emit energy by radiation.

The **Heat of Formation** is used when calculating the internal energy of a species. For governing equations which apply to solid models with a single species, the Heat of Formation will have no direct bearing on the solution. However, if multiple species exist in a model and species are allowed to react, the Heat of Formation will potentially affect the solution.

The **Comments** section is provided for documentation purposes. Comments are stored in the database file and serve as personal notes for the species.

Below the comments input is a table listing all the available features for the species. Each available feature will have a description, followed by a **Status** flag. If checked, then the data is available for the selected species. Finally, there is an action button for each feature. Pressing the button for a feature will either insert or remove the feature from the species data. Removing a feature will result in a loss of all the data associated with the feature. Therefore, removing a feature and then inserting it again will serve to re-initialize the data. Some features are required for a given species. Required features may not be removed. In this case the label on the given species. Required features may not be removed. In this case the label on the **Action** button will say “Permanent”, and the button will be disabled. There will be a notebook frame under an appropriate heading for each available feature which is used to display and modify the relevant data.

18.3.4 Solid Species Summary

In *GASPE*, solid species are used to build solid models in the database. The solid models are then used by solvers for solid heat transfer, material stress, and solid particle tracking.

Below is a summary of the data inputs for solid species.

Density Least Squares Curve Fit Located in the Density LSC frame and represents the density as a function of temperature. The curve fit is based on a 3rd order polynomial which *GASPE* uses to compute the density. This is a required input.

Thermal Conductivity Least Squares Curve Fit Located in the Thermal Cond LSC frame and represents the thermal conductivity as a function of temperature. The curve fit is based on a 3rd order polynomial which *GASPE* uses to compute the conductivity. This is a required input.

Specific Heat Least Squares Curve Fit Located in the Specific Heat LSC frame and represents the specific heat as a function of temperature. The curve fit is based on a 3rd order polynomial which *GASPE* uses to compute the specific heat. This is a required input.

Energy Level Splitting Data Located in the Energy Levels frame and representing inputs for simulations involving radiation (*i.e.*, lasers). Supports band-to-band emission and absorption modeling for solid state lasers.

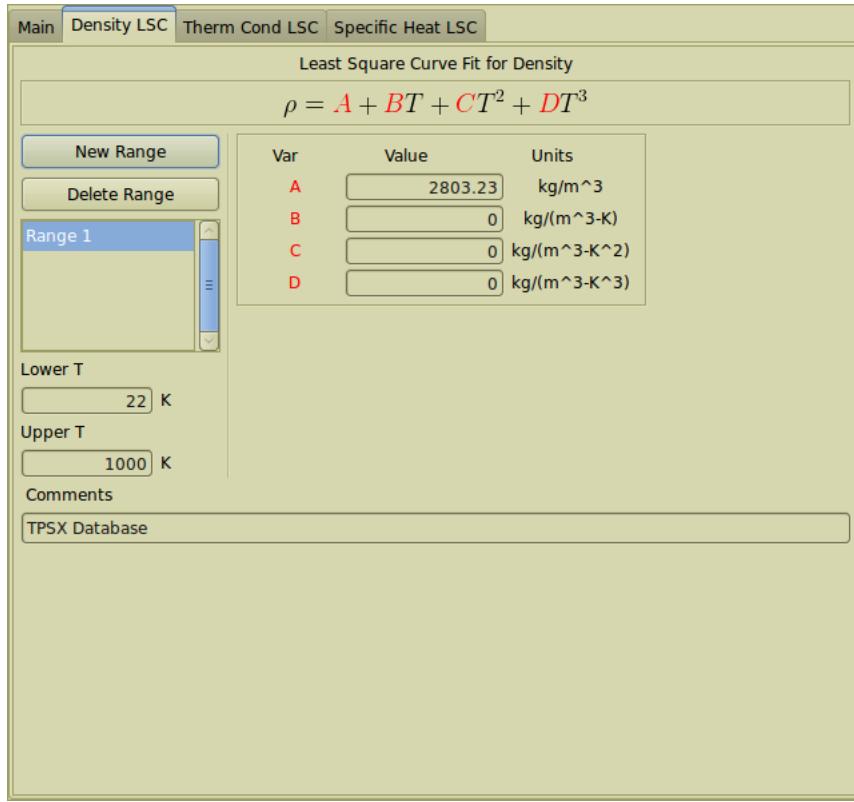


Figure 18.12: The Least Squared Curve fit for Density frame.

Density LSC Frame

The Density LSC frame, as seen in Fig. 18.12 on pg. 383, provides a mechanism for displaying and editing Least Squared Curve fits of the mass density of the solid as a function temperature.

Therm Cond LSC Frame

The Therm Cond LSC frame, as seen in Fig. 18.13 on pg. 384, provides a mechanism for displaying and editing Least Squared Curve fits of the thermal conductivity of the solid as a function temperature.

Specific Heat LSC Frame

The Specific Heat LSC frame, as seen in Fig. 18.14 on pg. 385, provides a mechanism for displaying and editing Least Squared Curve fits of the specific heat of the solid as a function temperature.

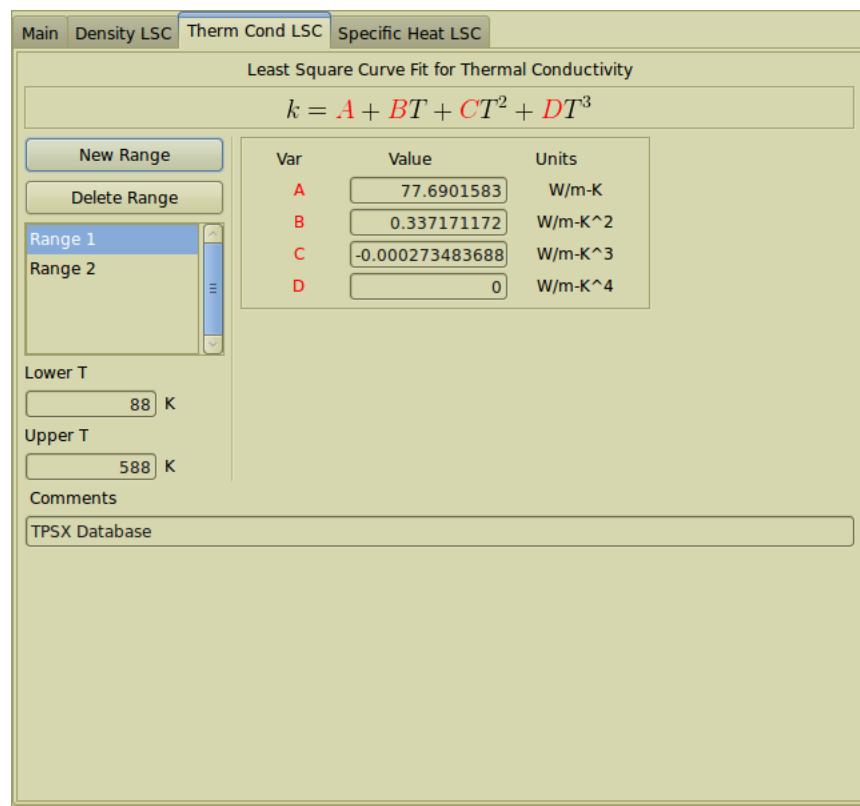


Figure 18.13: The Least Squared Curve fit for Thermal Conductivity frame.

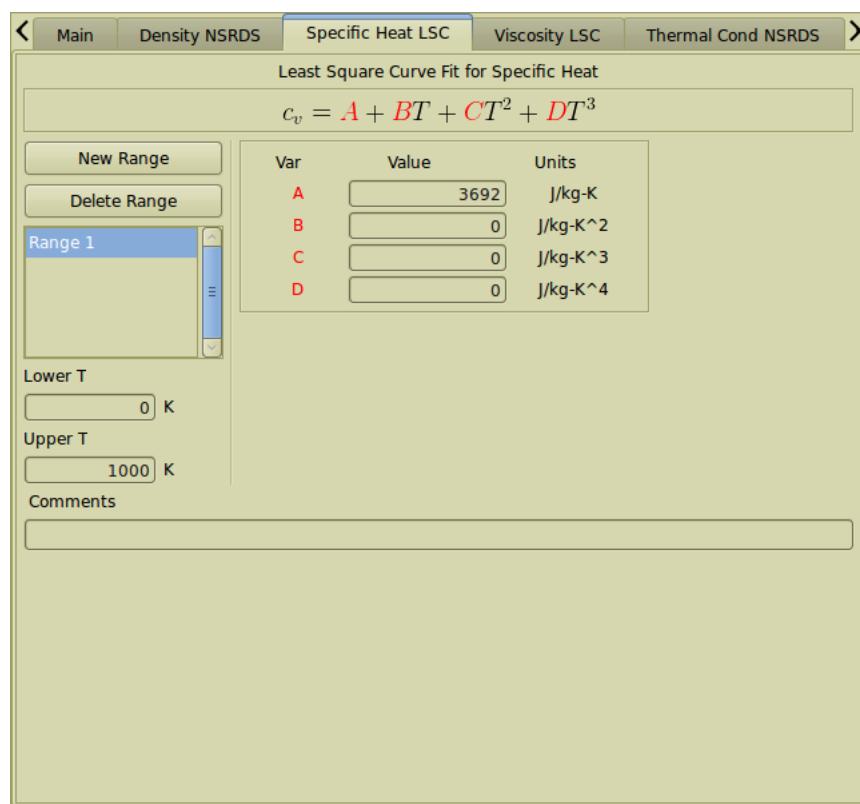


Figure 18.14: The Least Squared Curve fit for Specific Heat frame.

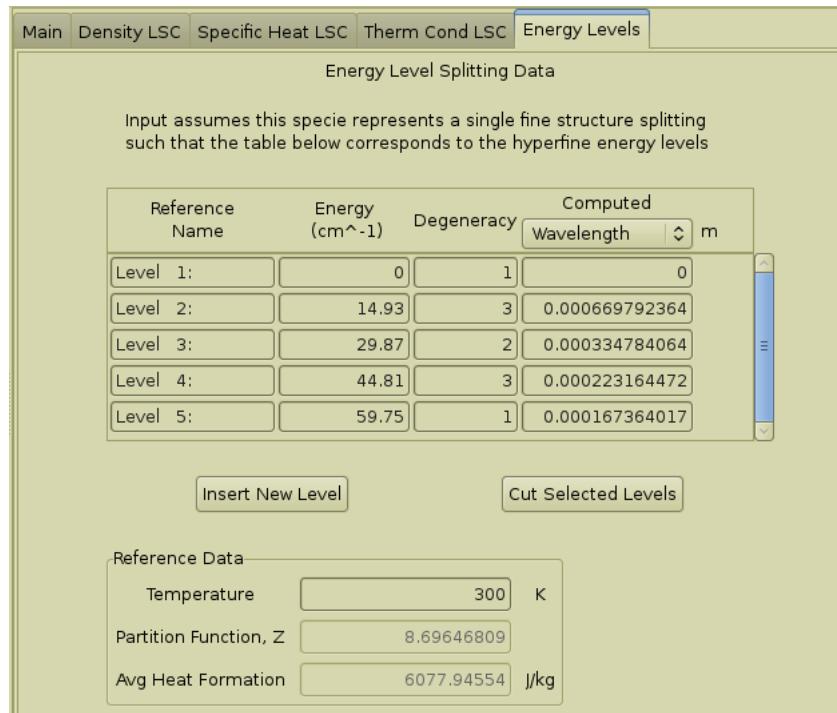


Figure 18.15: The Energy Levels frame for a solid species.

Solid Species Energy Levels

The energy level frame is used for simulations involving radiation (*i.e.*, lasers). The data in this frame enables band-to-band emission and absorption for solid state lasers. The required data comes from the relationship relating the emission cross section to the absorption cross section. The data includes the energy for each sub level (ei) along with the corresponding degeneracy value (gi). The number of sub-levels or splittings at each band will vary depending on the system. For each species involved in the emission or absorption transitions, energy level data must be provided.

18.3.5 Liquid Species Main Frame

Selecting a single liquid species node will display the liquid species **Main** frame as seen in Fig. 18.16 on pg. 387. The species main frame contains several inputs such as molecular weight, a comment field, and a table listing all the available species data models. The liquid species node will be labeled using either the specie's molecular formula or the species name.

The first option in this frame concerns species parsing. Parsing is the process in which the atomic content of a species is determined. This is beneficial for verifying the mass balance of reactions and automatically computing the molecular weight. Species parsing can only be performed if the molecular formula is given for the node name.

Note that if the species is entered as a molecular formula, each element of the molecule

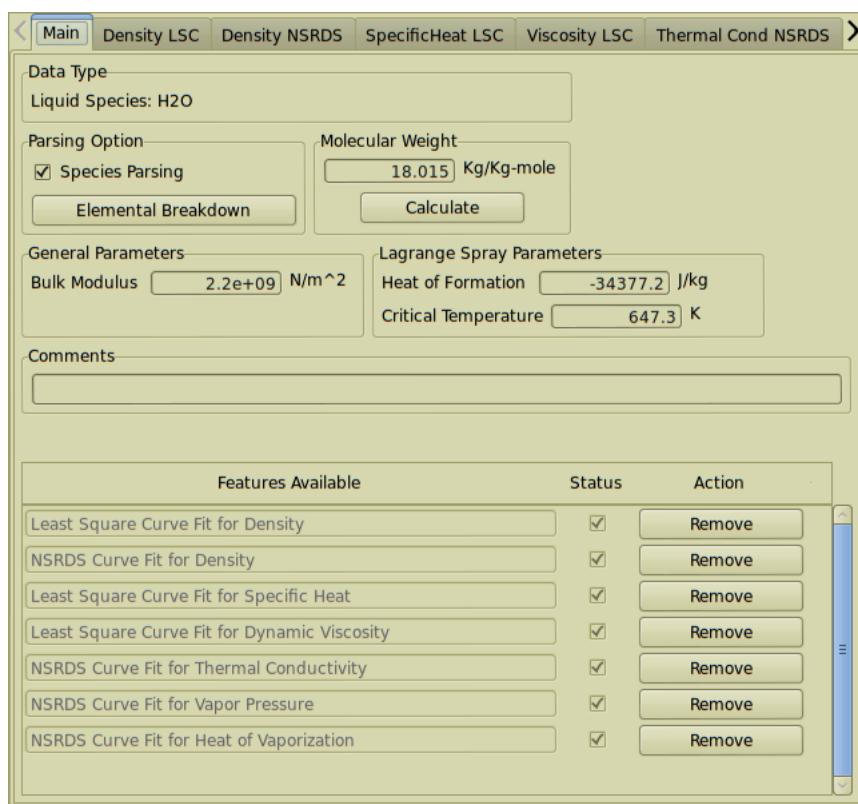


Figure 18.16: The main interface frame of a liquid species.

must begin with a capital letter (*e.g.*, NaCl). No blank spaces are allowed in the species name. The number of atoms in the molecule must follow the individual element, and the charge (if applicable) may follow either the element name or the molecular formula. All characters between capital letters are assumed part of an element group. When elements in the molecular formula are recognized as members of the periodic table, the molecular weight of the molecule may be computed using the **Calculate** button. As examples, H_2O and O_2^+ are written as H2O and O2+.

If the name of the species is given rather than the molecular formula, the user is free to name the species as desired. In this case the user must enter the molecular weight by hand and species parsing will not be available. Only molecular formulas (and not species names) may be used in chemical reactions.

The **Molecular Weight** for a liquid species can be calculated based upon the elemental breakdown, or entered by hand. The molecular weight is used within multi-species models for computing mixture concentrations.

The **Bulk Modulus** (K) measures the liquids resistance to uniform compression. It is used to compute the speed of sound using the formula $a = \sqrt{K/\rho}$.

The **Heat of Formation** parameter is used in the Lagrange spray solver when modeling liquid particle evaporation.

The **Critical Temperature** is the temperature at the liquid critical point. The parameter is used in the Lagrange spray solver when modeling liquid particle evaporation.

18.3.6 Liquid Species Summary

Below is a summary of the data inputs for liquid species.

Density Least Squares Curve Fit Located in the Density LSC frame (under the Density heading) and representing the density as a function of temperature. The curve fit is based on a 3rd order polynomial which *GASPE* uses to compute the density.

Density National Standard Reference Data Series (NSRDS) Curve Fit Located in the Density NSRDS frame (under the Density heading) and representing the density as a function of temperature.

Specific Heat Least Squares Curve Fit Located in the Specific Heat LSC frame and representing the specific heat as a function of temperature. The curve fit is based on a 3rd order polynomial which *GASPE* uses to compute the specific heat.

Viscosity Least Squares Curve Fit The dynamic viscosity as a function of temperature. The curve fit is based on a 3rd order polynomial that the user can specify or allow *GASPE* to curve fit using a least squares method. The data is used when solving the Navier-Stokes equations for a liquid model.

Thermal Conductivity National Standard Reference Data Series Curve Fit Located in the Thermal Cond NSRDS frame and representing the thermal conductivity as a function of temperature.

Vapor Pressure National Standard Reference Data Series Curve Fit Located in the Vapor Pressure NSRDS frame and representing the liquid vapor pressure as a function of temperature. The parameter is used in the Lagrange spray solver when modeling liquid particle evaporation.

Heat Vapor National Standard Reference Data Series Curve Fit Located in the Heat Vapor NSRDS frame and representing the liquid heat of vaporization as a function of temperature. The parameter is used in the Lagrange spray solver when modeling liquid particle evaporation.

Density LSC Frame

The Density LSC frame, as seen in Fig. 18.12 on pg. 383, provides a mechanism for displaying and editing Least Squared Curve fits of the mass density of the liquid as a function of temperature.

Density NSRDS Frame

The Density NSRDS frame, as seen in Fig. 18.17 on pg. 390, provides a mechanism for displaying and editing National Standard Reference Data Series Curve Fits of density of the liquid as a function of temperature.

Specific Heat LSC Frame

The Specific Heat LSC frame, as seen in Fig. 18.14 on pg. 385, provides a mechanism for displaying and editing Least Squared Curve fits of the specific heat of the liquid as a function of temperature.

Viscosity LSC Frame

The Viscosity LSC frame, as seen in Fig. 18.18 on pg. 391, provides a mechanism for displaying and editing Least Squared Curve fits of the viscosity of the liquid as a function of temperature.

Thermal Cond NSRDS Frame

The Thermal Cond NSRDS frame, as seen in Fig. 18.19 on pg. 392, provides a mechanism for displaying and editing National Standard Reference Data Series Curve Fits of the thermal conductivity of the liquid as a function of temperature.

Vapor Pressure NSRDS Frame

The Vapor Pressure NSRDS frame, as seen in Fig. 18.20 on pg. 393, provides a mechanism for displaying and editing National Standard Reference Data Series Curve Fits of the vapor pressure of the liquid as a function of temperature.

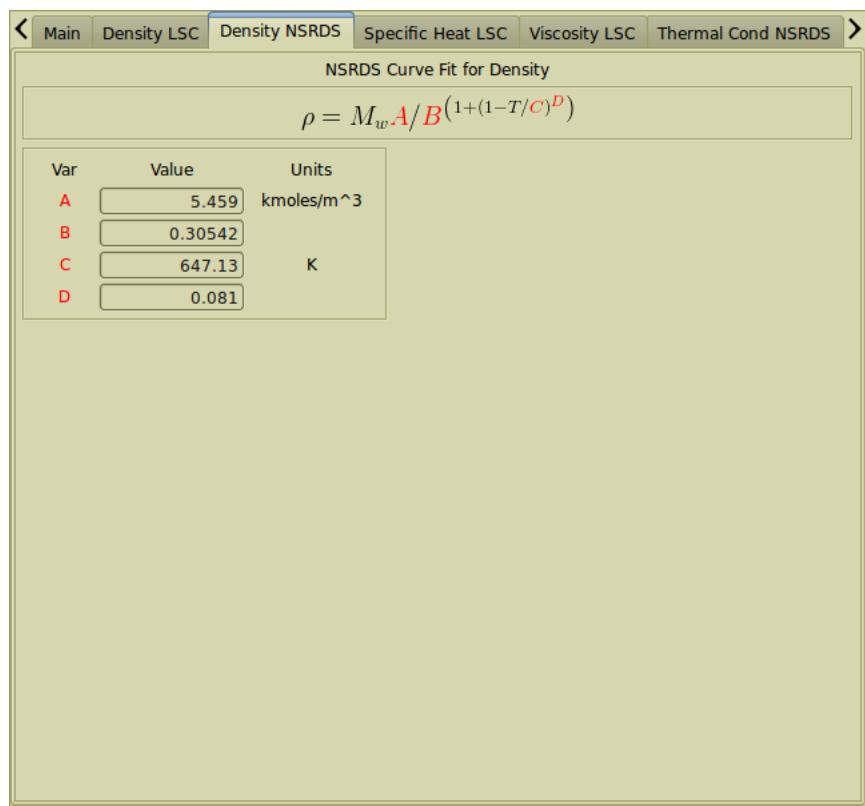


Figure 18.17: The NSRDS Curve Fit for Density frame.

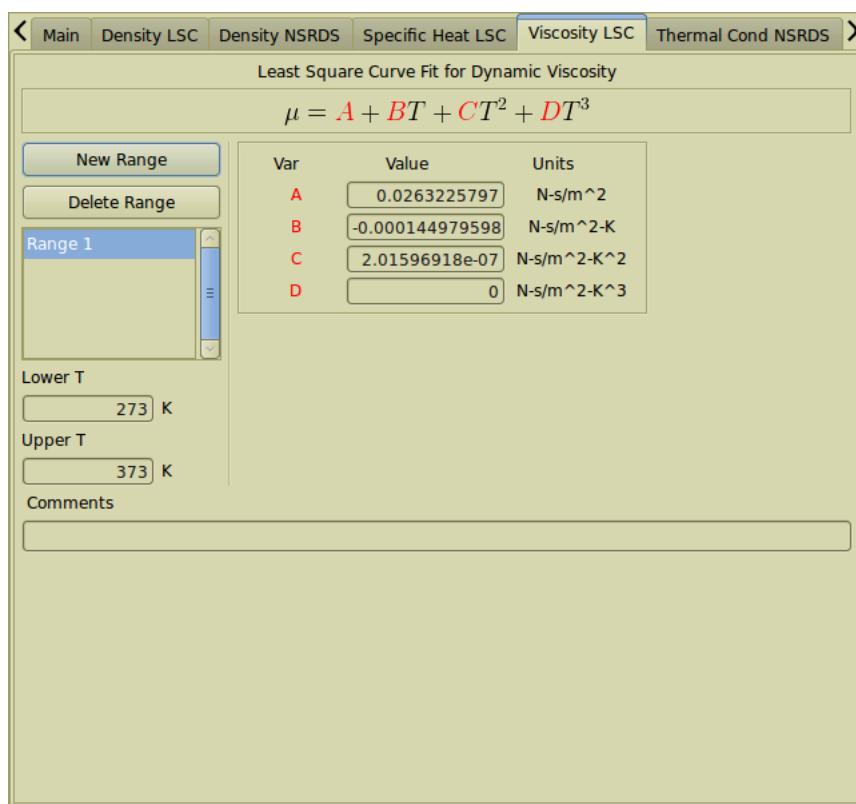


Figure 18.18: The Least Squared Curve fit for Viscosity frame.

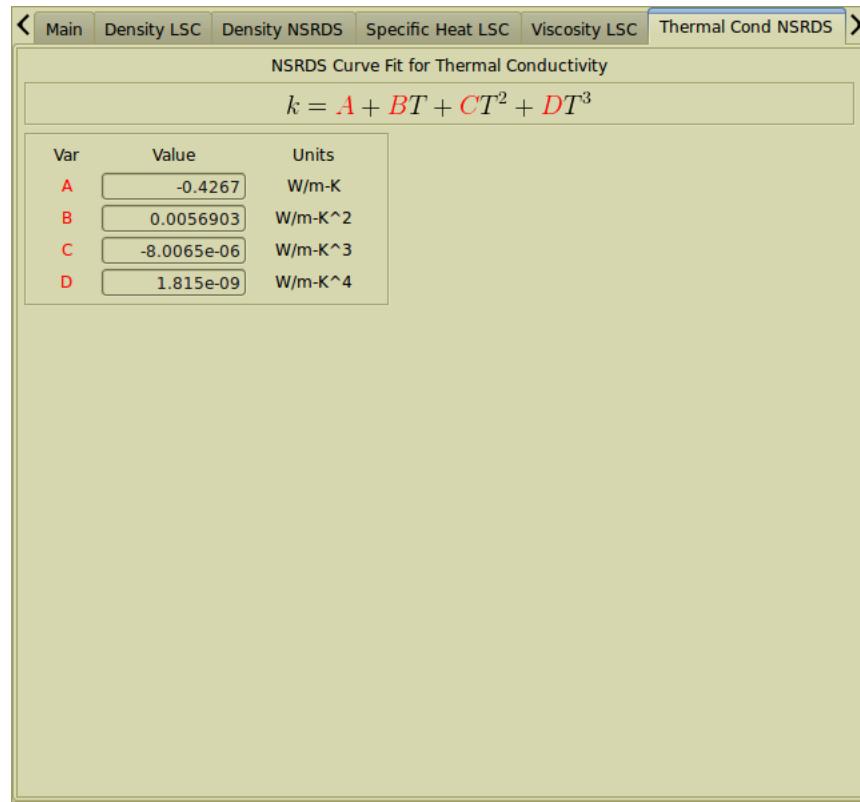


Figure 18.19: The NSRDS Curve Fits for Thermal Conductivity frame.

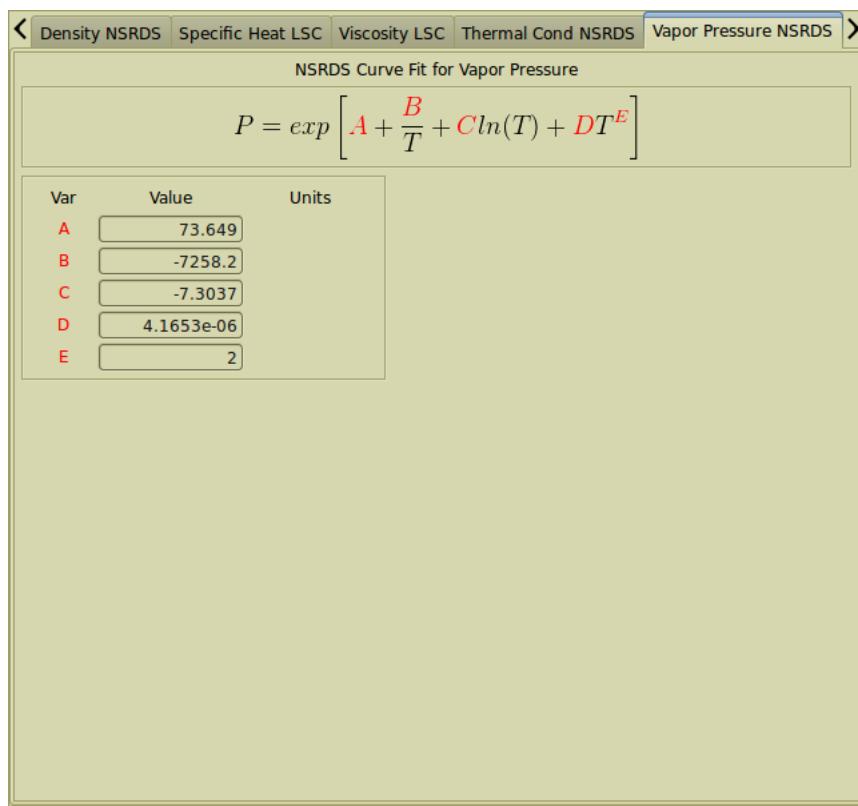


Figure 18.20: The NSRDS Curve Fits for Vapor Pressure frame.

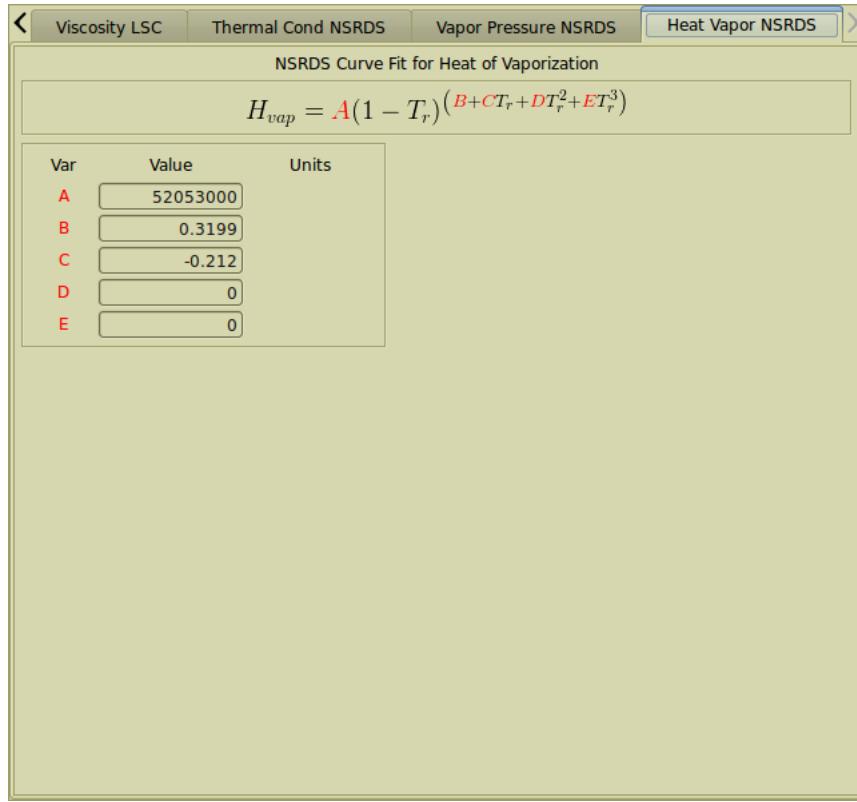


Figure 18.21: The NSRDS Curve Fits for Heat of Vaporization frame.

Heat Vapor NSRDS Frame

The Heat Vapor NSRDS frame, as seen in Fig. 18.21 on pg. 394, provides a mechanism for displaying and editing National Standard Reference Data Series Curve Fits of the heat of vaporization of the liquid as a function of temperature.

18.4 Reactions

Reactions are used to describe the chemical process in which atoms or molecules interact with each other, resulting in a molecular change. In *GASPE*, reactions are made up of species that already exist in the database. If a reaction needs a species that does not exist, the user must first create the species before creating the reaction.

The practical aspect of creating and using reactions are discussed here. For more in-depth details on chemical kinetics, the user is referred to the *GASPE* Technical Manual.

The species initially involved in a reaction are called the reactants and are normally placed on the left hand side of the reaction. The species that result from the chemical process are referred to as the products and are normally located on the right hand side of reactions. This nomenclature will be used in the discussion that follows.

There are a number of different types of reactions in the database. A summary of each possible reaction type is given below.

Gas Reaction Gas reactions are used with a gas model for Navier-Stokes applications. A gas reaction contains a forward rate, which is normally calculated with an Arrhenius equation, and an equilibrium rate. A number of methods are available for calculating the equilibrium rate. The equilibrium methods are: Arrhenius, Park 1, Park 2, minimization of Gibbs free energy using the NASA LeRC curve fits, and a collisional mixing rate.

Surface Reaction Surface reactions are used with a gas model for the Navier-Stokes solver. Two numerical models for surface reactions are available. The first reaction type is a basic catalytic surface, where the reactants can be either all gaseous, or a combination of gaseous and solid. The product of this type of reaction will be gaseous. The reaction is assumed to be non-reversible such that there is no backward rate. The second reaction models sublimation/evaporation. For this reaction, the reactant is solid, while the product is gaseous. The reaction will attempt to resolve an equilibrium condition where the product achieves a partial pressure equal to the vapor pressure for the gas.

Solid Reaction Solid reactions are used with solid models. Reactions are based on either the Arrhenius equation or an exponential decay rate.

18.4.1 Creating a Reaction

Reactions are created in one of three ways; import, duplication, or creation of a new, empty reaction. Reactions may be imported using the **Options→Import & Merge** feature. Assuming the thermo-chemistry file selected for import and merge contains new reactions, they will be added to the existing hierarchy in the same relative position that they maintain in the imported file.

Reactions may be duplicated by first selecting one or more Reaction Nodes within the tree view. Pressing the right mouse button over the selected nodes will display a pop-up menu, one of whose sub-menus is **Edit**. Within the **Edit** sub-menu, selecting the **Copy** button will duplicate the selected items, and save them in an internal clipboard. Selecting an appropriate parent folder, and depressing the right mouse button, the user can then select **Paste** from the **Edit** sub-menu, in order to duplicate the copied elements.

Finally, new reactions may be created by selecting a new parent folder, pressing the right mouse button, and selecting an appropriate new reaction type from within the **New Reaction** sub-menu. Before creating a new reaction, the user must create any new species which are found in the reactants or products of the new reaction.

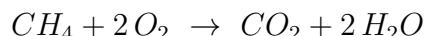
18.4.2 Entering the Reaction Equation

Once a reaction is created inside the database, the user must edit the reaction name in order to specify the chemical reaction equation. Editing the reaction is done by double-clicking the reaction name in the tree view.

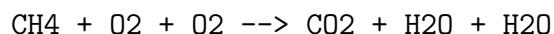
The atoms and molecules of a reaction must follow the letter convention set for the species. That is,

- No spaces are allowed in a specie name.
- Characters between capital letters are assumed part of a single element. The first letter in the specie name must be a capital letter. The only exception is for an electron, which is written as e^- .

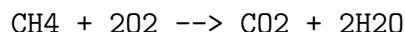
Additionally, if multiple molecules of the same type appear in the chemical equation, they may be entered either individually or in coefficient form. For example, the combustion of methane written as



may be entered as either



or

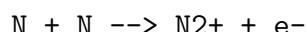


Each element of the equation must be separated by a space and the forward arrow (\rightarrow) must be represented by \rightarrow to separate products from reactants. Notice that no space is allowed between the 2 and the O_2 when $2O_2$ is written as 202 .

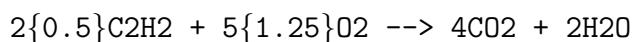
A chemical reaction which contains charge should be written with the charge given last. For example, the chemical equation for the combination/ionization of nitrogen



is written as



In general, the stoichiometric coefficients in a reaction should be integer based. In special circumstances, a simplified reaction mechanism may wish to be used in which real coefficients are utilized in the reaction. For example, Westbrook and Dryer [62] discuss such models for hydrocarbon fuel reactions. From their paper, a reaction for acetylene would appear in the database as



The real-valued coefficients must be inserted using brackets, while integer coefficients representing the stoichiometric mass balance must be placed in front of the brackets.

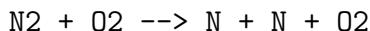
When you press return after entering your chemical reaction, the equation is checked for atom and charge conservation. If any errors are detected, a dialog box will appear informing you the equation is not valid. Invalid reactions will turn red to indicate that there is a problem with the equation.

18.4.3 The Main Reaction Frame

All reactions, whether they be gas, surface, or solid reactions share common types of information. The information types will be displayed in the **Main** reaction frame, as shown in Fig. 18.22 on pg. 398, whenever a single reaction node is selected. The **Data Type** label will identify the selected reaction as either a Gas, Surface, or Solid Reaction, as well as the equation for the reaction itself.

If the **Mass Balance Check** check box is selected, an internal algorithm will test to see that mass is conserved between reactants and products. In order for the mass balance check to succeed, all species present in the reaction must conform to the species naming rules for elemental compositions. If generic species names are used, such as “fuel” and “oxidizer”, then a mass balance check cannot be performed.

Pressing the **Stoichiometric Breakdown** button will display a pop-up dialog which contains the computed stoichiometric coefficients for each species present in the reaction. For example, if the reaction given as



is selected, and the button is pressed, Fig. 18.23 on pg. 398 will be displayed.

Forward and Equilibrium reaction rates are computed for each reaction present in a given model. While the forward reaction rate is almost exclusively computed using an Arrhenius formulation, the form of the equation for the equilibrium rate will vary depending upon the selected chemistry model. The algorithms available for computing reaction rates are presented through the **Reaction Rate Type** option menu. The user may change the equations used to predict reaction rates to coincide with a particular chemistry model.

A **Comments** section is provided for documentation purposes. Comments are stored in the database file and serve as personal notes for the reactions.

Specific to the Surface Reaction is the **Heat of Reaction**. This value represents a latent heat of reaction for surface reactions which involve a mass transfer from solid to gas. For example, for a sublimation reaction, the heat of reaction represents the latent heat of sublimation or evaporation.

18.4.4 Rate Equation Types

The different methods of calculating the reaction rates are now discussed. The forward rate and equilibrium methods are set in the reactions **Main** frame under **Reference Info**. Note

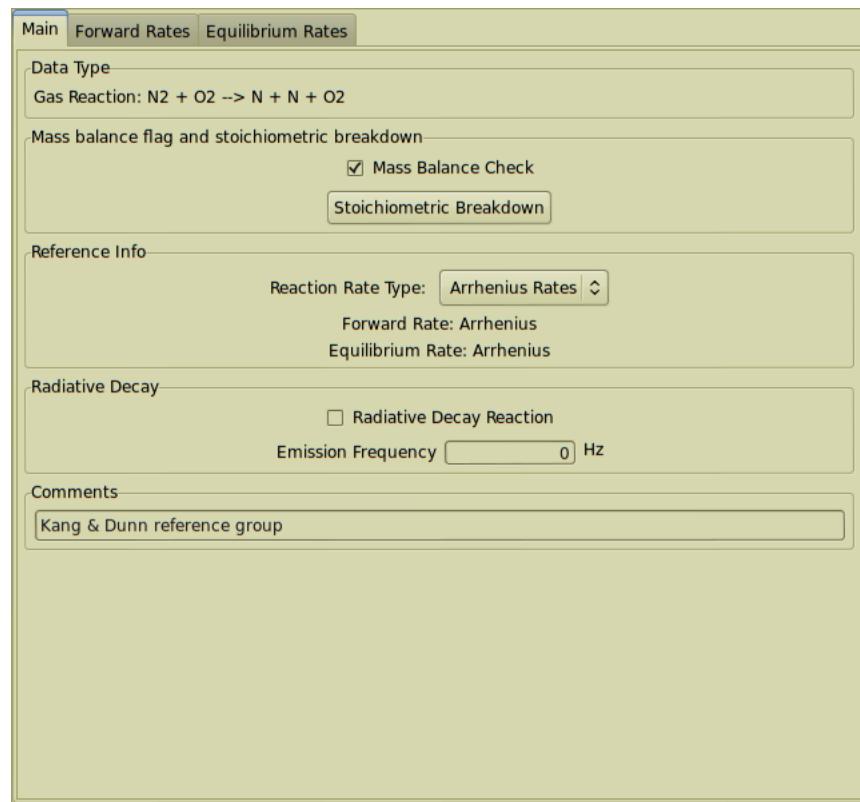


Figure 18.22: The Main Reaction Frame.

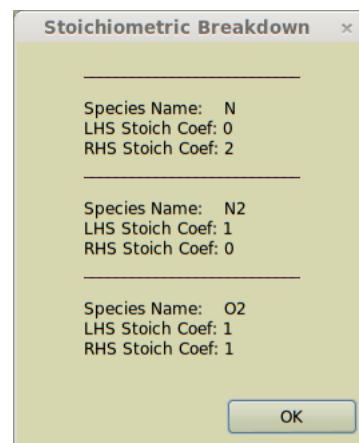


Figure 18.23: The Stoichiometric Coefficients displayed for an example reaction.

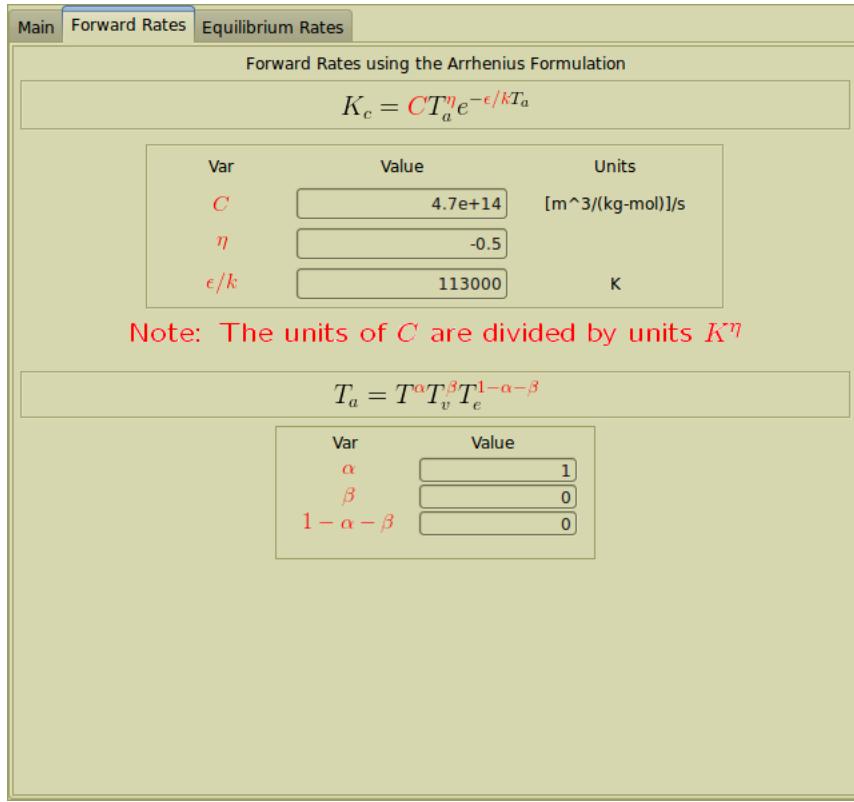


Figure 18.24: The Arrhenius Forward Reaction Rate Frame.

that the equilibrium rate equation, along with the forward rate, is used to calculate the backward reaction rate. Not all rate type combinations are present for each Reaction type.

Forward Arrhenius Rate Equation

The forward gas reaction rate, k_f , is normally calculated with an Arrhenius equation which is displayed in the **Forward Rates** frame, Fig. 18.24 on pg. 399. The effective temperature (T_a) used in calculating k_f contains contributions from the translational, vibrational and electron/electronic temperatures. The exponents for each temperature are given at the bottom of the frame. The effective temperature is used only with preferential dissociation.

Forward Magnussen Rate Equation

An alternative to the Arrhenius equation for computing forward rates is the Magnuseen eddy-dissipation model [63, 64], whose inputs and equation are shown in Fig. 18.25 on pg. 400. This model assumes that chemical reactions occur much faster than turbulence can mix the reactants and heat into the reaction region. The method can be a good assumption for many combustors because most useful fuels are fast burning. Using this rate equation requires a two-equation turbulence model such as $K-\epsilon$ or $K-\omega$. The equation has two input constants

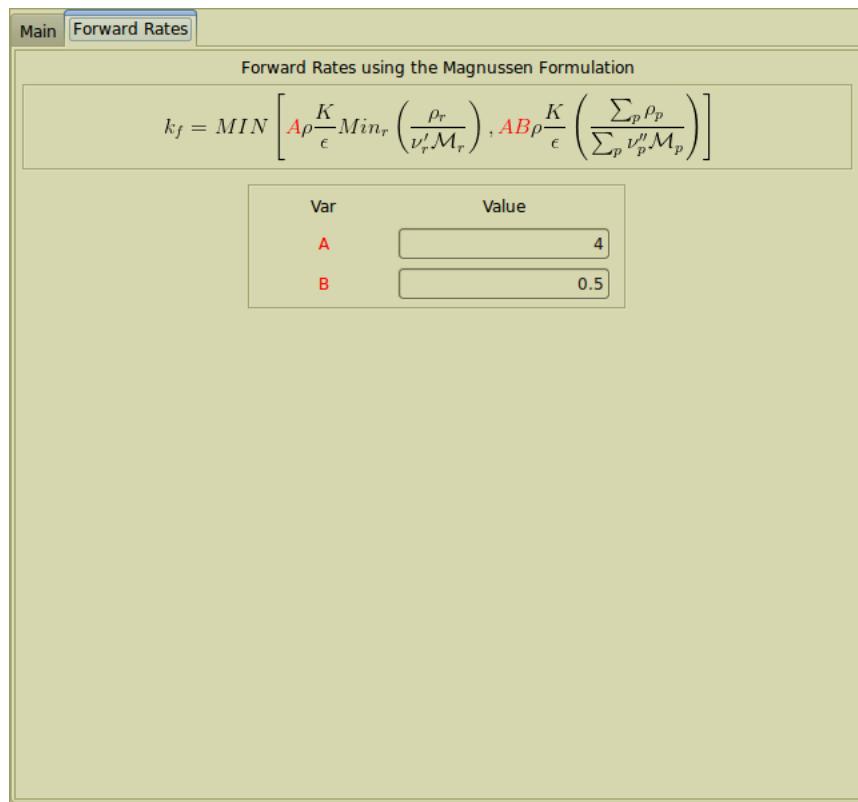


Figure 18.25: The Magnussen Forward Reaction Rate Frame.

(A and B) corresponding to reactants and products respectively. The default value for A is 4.0 and the default value for B is 0.5. Setting B to a large number effectively removes the limit on products. There is no corresponding equilibrium rate with this method and it is only available for gas reactions.

Forward Exponential Decay Equation

The exponential decay equation is a sub-set of the modified Park 2 equation, which is expressed as

$$k_f = e^{A_1/T + A_2 + A_3 \ln T + A_4 T + A_5 T^2}$$

This allows support for either constant or temperature dependent decay rates. For a constant decay rate, a single input is used to set up the exponential rate, which is the parameter τ representing the time scale. For temperature dependent rates, the user inputs (either by file or manually) a table with the temperature and corresponding time scale (τ). The GUI will then curve fit the data to the modified Park 2 equation.

The forward reaction assumes a first order equation and is normally used to model radiative or non-radiative decay of a species in an excited energy state. For example, if an atom or molecule is decaying from a high energy state to a lower one (note that each state

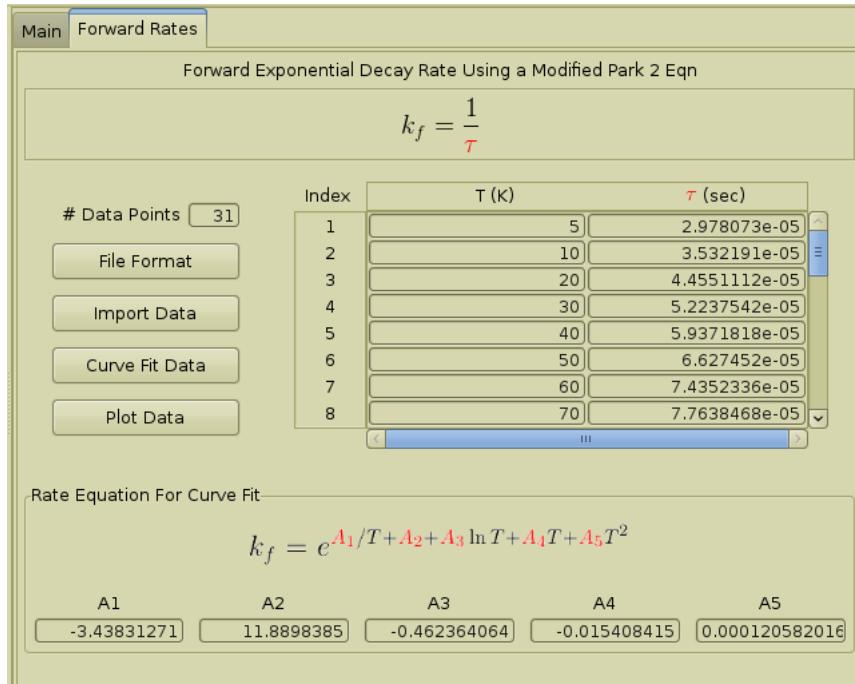


Figure 18.26: The exponential decay forward rate frame.

must have its own species in the database), then τ would represent the half-life of the decay process. When using this option for the forward rate, the backward rate is assumed to be zero (non-reversible reaction).

Forward Collisional Mixing Rate

The collisional mixing rate is a sub-set of the Arrhenius equation and is shown in Fig. 18.27 on pg. 402. It models a second order reaction in which two different atoms or molecules collide and result in an energy transfer. In order to compute the forward rate, the collisional cross section (σ , given in SI units of m^2) is required. The terms m_A and m_B represent the mass of the reactants (species on the left-hand side of the reaction). N_A is Avogadro's number, and k_B is the Boltzmann constant.

Forward Rate Equation for Basic Surface Catalysis Reactions

The forward surface reaction rate, k_f , is calculated with an Arrhenius type equation which is displayed in the **Forward Rates** frame, Fig. 18.28 on pg. 403. This reaction rate is a function of the thermal velocity of the reactant, a critical Temperature, T_{crit} , and a catalytic efficiency, β , which represents the fraction of reactants intersecting the surface which actually react. If the surface rate does not include the exponential term as a function of T and T_{crit} , simply set T_{crit} to 0..

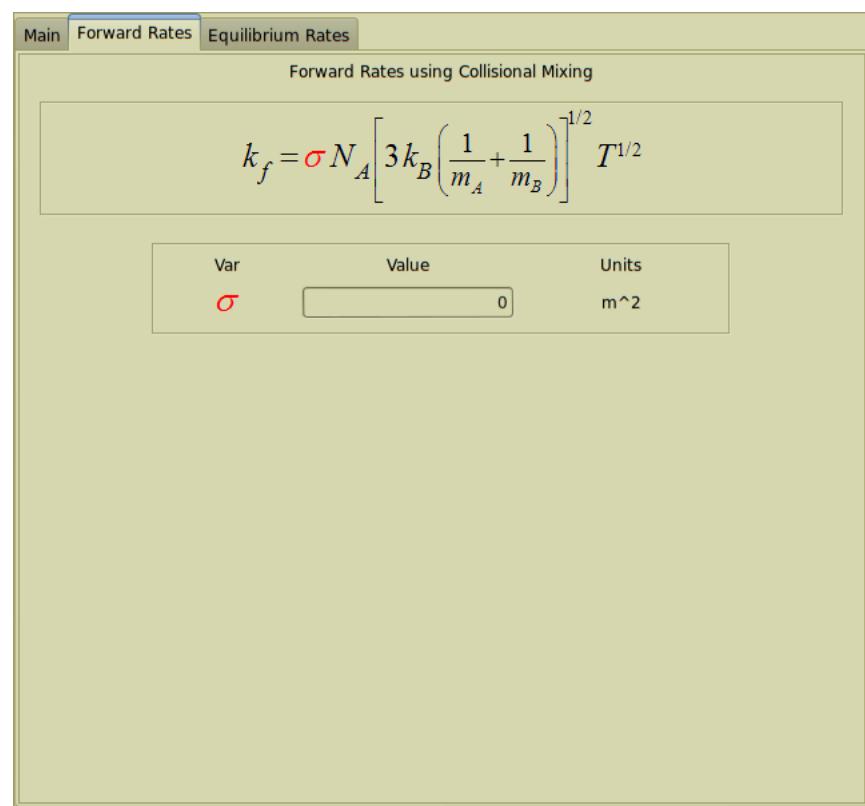


Figure 18.27: The forward rate equation for collisional mixing.

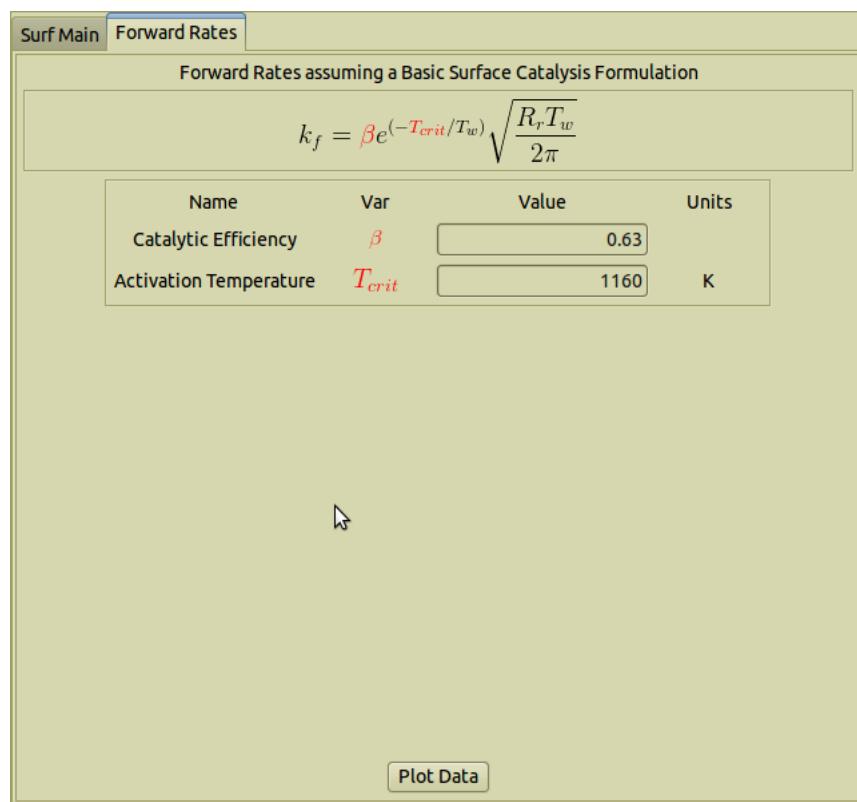


Figure 18.28: The Basic Surface Catalysis Forward Rate Frame.

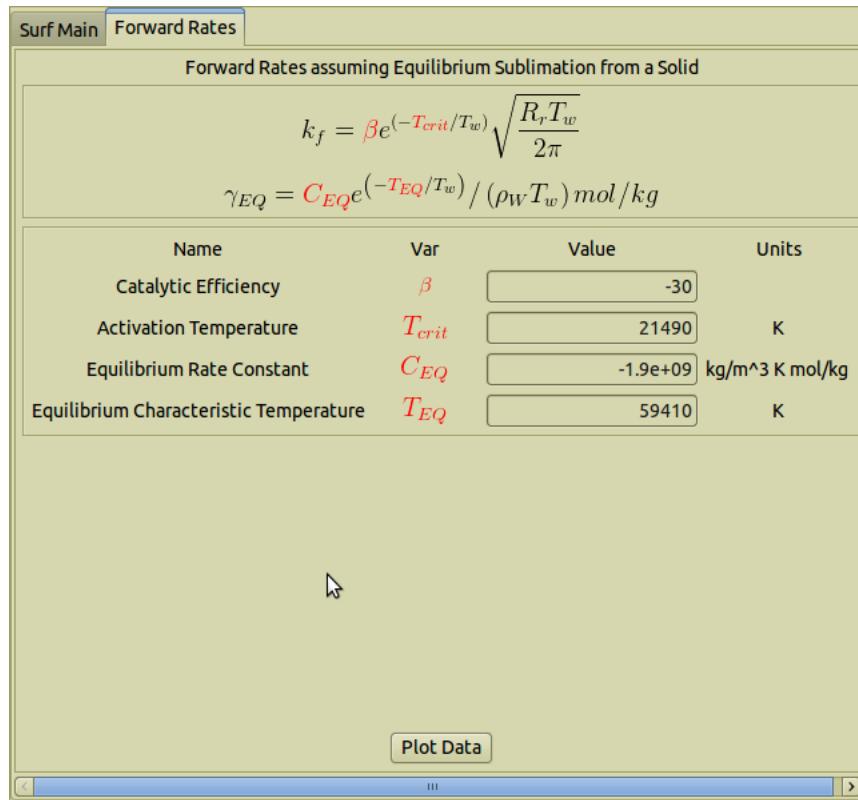


Figure 18.29: The Surface Sublimation Rate Frame.

Forward Rate Equation for Surface Sublimation Reactions

Surface sublimation/evaporation is modelled after the formulation of Park [65], and all modeling coefficients are shown in the **Forward Rates** frame, Fig. 18.28 on pg. 403. The forward rate for condensation/evaporation uses the same formulation as the Basic Surface Catalysis Reaction, but the sublimation reaction is modelled as an equilibrium rate which is associated with an equilibrium concentration or vapor pressure. The equilibrium concentration γ_{EQ} requires a rate constant, C_{EQ} and critical temperature, T_{EQ} .

Equilibrium Arrhenius Rate Equation

The equilibrium Arrhenius rate equation is similar to the forward Arrhenius Rate and is located in the **Equilibrium Rates** frame. The only difference between the Arrhenius forward and equilibrium rate input is that the effective temperature (T_a) is equal to the translational temperature for the equilibrium rate.

Main Forward Rates Equilibrium Rates

Equilibrium Rates using the Arrhenius Formulation

$$K_c = CT^\eta e^{-\epsilon/kT}$$

Var	Value	Units
C	18000	[$m^3/(kg\cdot mol)$]/s
η	0	
ϵ/k	113000	K

Note: The units of C are divided by units K^η

Figure 18.30: The Arrhenius Equilibrium Reaction Rate Frame.

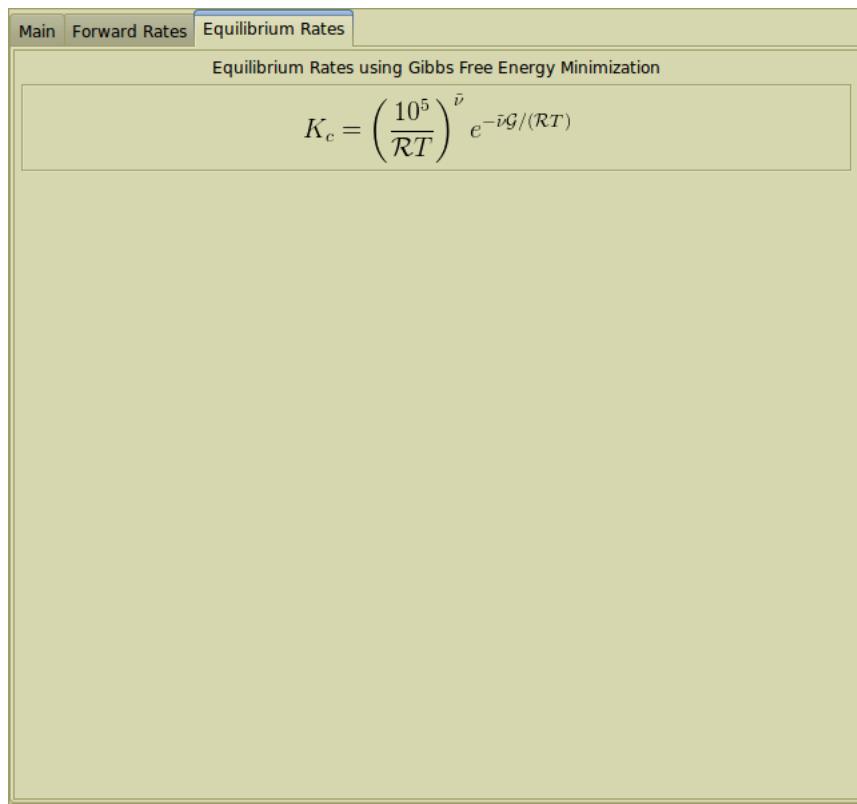


Figure 18.31: The Gibbs Free Energy Minimization based Equilibrium Reaction Rate Frame.

Equilibrium with Gibbs Free Energy Minimization

This method uses the Gordon/McBride curve fits and minimization of the Gibbs free energy. With this option the user is not required to input any data since all the necessary rate information is extracted from the Gordon/McBride curve fit data. The one requirement to use this rate method is that all the species involved in the reaction must have the curve fit data. This method is only available for gas reactions.

Equilibrium Park 1 Reaction Model

The Park [8] 1 reaction model uses a fourth-order polynomial in $z \equiv 10,000/T$ to curve-fit the natural log of the equilibrium constant. The log of the equilibrium constant is nearly linear with the inverse of the temperature for many reactions. This method is only available for gas reactions.

Equilibrium Park 2 Reaction Model

The Park [66] 2 reaction model uses the structure of diatomic molecules and their partition functions to devise a second method for calculating the equilibrium constant. This method

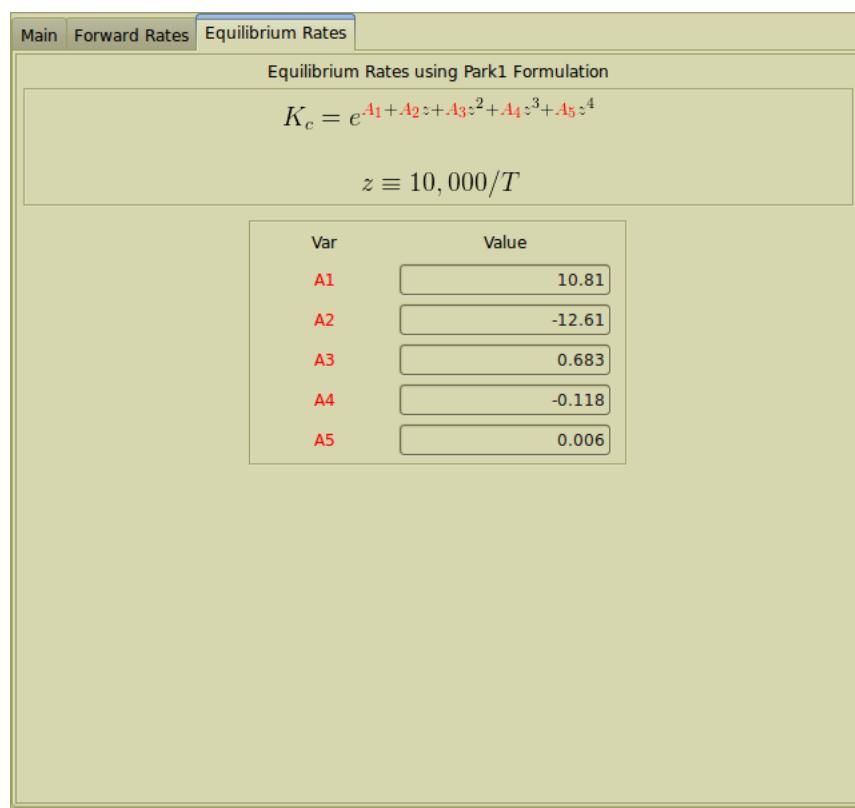


Figure 18.32: The Park 1 Equilibrium Reaction Rate Frame.

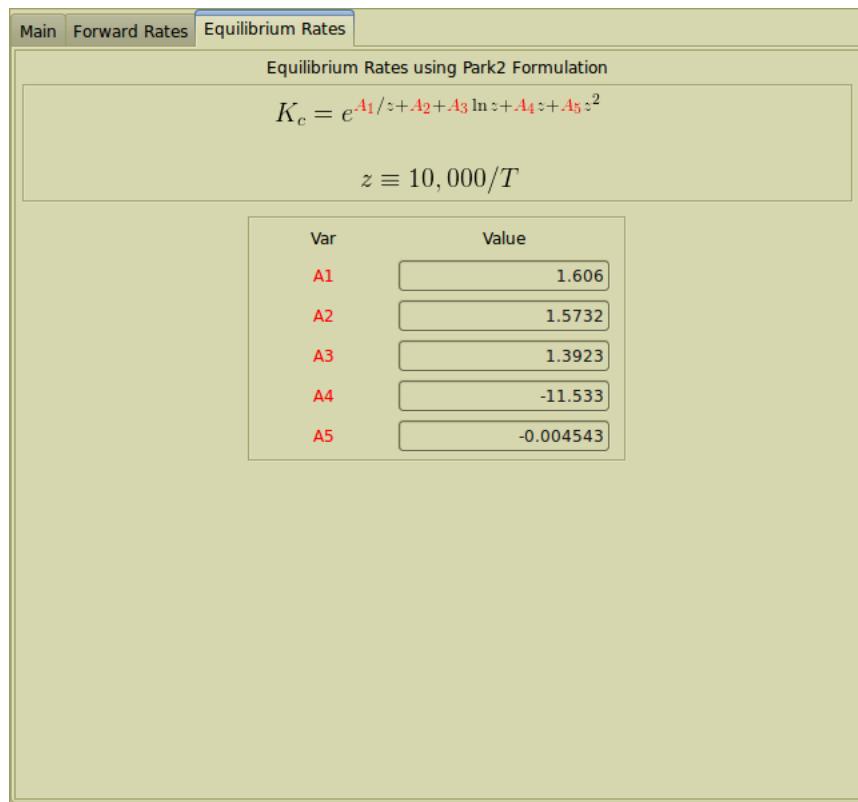


Figure 18.33: The Park 2 Equilibrium Reaction Rate Frame.

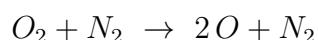
is only available for gas reactions.

Equilibrium Collisional Mixing Rate

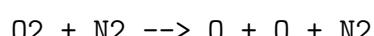
Similar to the forward collisional mixing rate, the equilibrium rate is also a sub-set of the Arrhenius equation. The rate equation is shown in Fig. 18.34 on pg. 409. The inputs to the equilibrium rate are a modeling constant and the energy change in Joules due to the collision.

18.4.5 Entering Rate Data

As an example, consider the catalytic dissociation of diatomic oxygen as



which is entered as a gas reaction with the equation



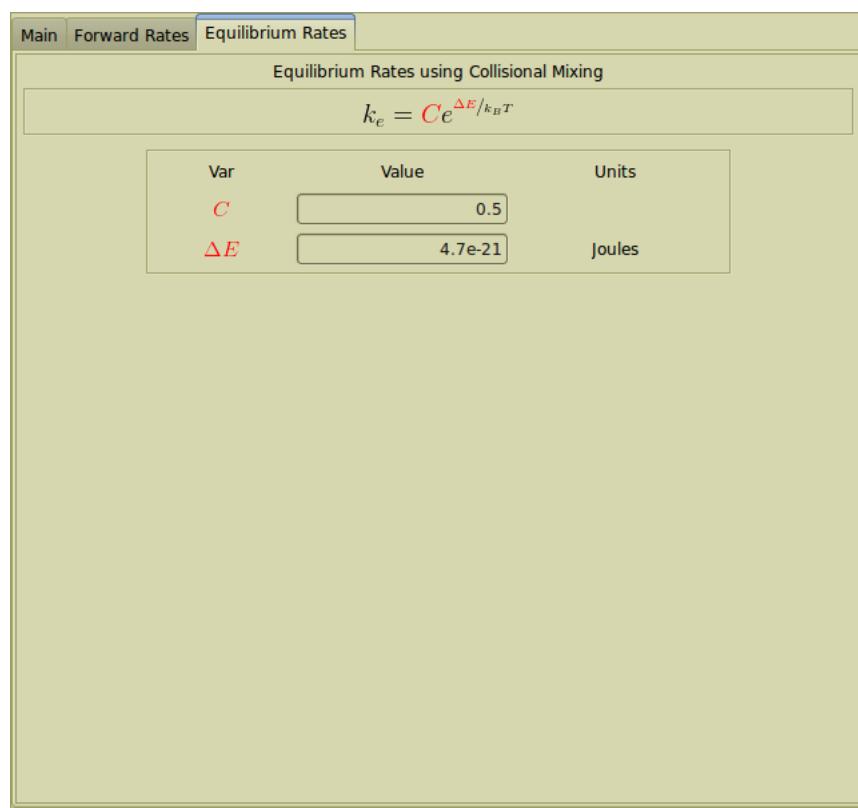


Figure 18.34: The equilibrium rate equation for collisional mixing.

With a correct chemical equation entered, data must then be provided to calculate the forward reaction rate and the equilibrium constant. With the reaction selected in the tree view, the forward rate data is located in the **Forward Rates** frame and the equilibrium data is located in the **Equilibrium Rates** frame.

For the forward rates, a minimum of three constants are necessary to define the Arrhenius equation for k_f . Note that the units for k_f must be in terms of $kg \cdot mol, m$ and s as

$$k_f \sim \left(\frac{m^3}{kg \cdot mol} \right)^{(\mathcal{O}_j - 1)} s^{-1}, \quad (18.1)$$

where \mathcal{O}_j is the sum of the number of moles of reactants in the chemical equation. In our case, $\mathcal{O}_j = 2$ because a mole of O_2 reacts catalytically with a mole of N_2 (*i.e.*, a total of two moles). The units of k_f are then $m^3/(kg \cdot mol)(s)$. Note that the temperature units in C cancel those from T^η where the temperature is given in Kelvin. The activation energy (ϵ/k) is given in Kelvin.

The reaction-rate data for each constant is entered by selecting the corresponding variable in the table and typing the correct value in the type-in box. Entering the data for the equilibrium constant is performed identically and the end result is shown in

18.4.6 Final Word on Reactions

Reactions in the *DBASEMGR* may be cut, copied and pasted. Once a reaction has been copied, select the folder in which the reaction will be pasted. The pasted reaction will be identical to the original, so it is up to the user to organize appropriately.

18.5 Models

In *GASPEX*, a database model consists of one or more species and possibly reactions. At a minimum, a model must contain at least one species. Reactions are not necessary to create a valid database model, but are needed if performing finite-rate chemistry in the solver.

There are a number of different types of models in the database. A summary of each possible model type is given below.

Gas Model A gas model is needed by the Navier-Stokes (NS) physical model and contains gas species, gas reactions, and surface reactions. The gas reactions are necessary if doing equilibrium or finite-rate chemistry (see Sec. 11.2.5 on pg. 168). Surface reactions are used with select boundary conditions that solve for surface energy balances.

Solid Model A solid model is needed by the solid thermodynamics (ST) physical model, the material stress (MS) physical model, or the Lagrange Spray physical model if tracking solid particles. The solid model should contain a single solid species.

Liquid Model A liquid model is needed by the Lagrange Spray physical model if tracking liquid particles, or the Navier-Stokes (NS) physical model modeling liquid flow. The liquid model should contain a single liquid species.

18.5.1 Creating a Model

Before creating a new model, all the necessary species and reactions must be available in the database. Collecting the necessary data is sometimes a time-consuming task, but once the species and reaction data are entered, creating a model is a relatively easy task. The steps for creating a new model are as follows:

1. Run **DBASEMGR** and open the current database.
2. Add any new species or reactions that are not present in the database but needed by the model.
3. Select a folder from the tree view which will contain the new model.
4. Using the right mouse button, select the type of new model you wish to create using the **Edit→New Model** pop-up menu. This operation will insert a new model inside the currently selected folder and place it at the bottom of the folder's contents.
5. Select the species you wish to associate with the new model and drag-n-drop them into the models species folder. Depending on the type of model created, a species sub-folder will be available when the model folder is opened.
6. If using reactions with the model, select the reactions and drag-n-drop them into the models reaction folder.
7. Supply any modeling data which is specific to the new model.
8. Save the new model to the database file using the **File** pull-down menu on the menu bar and select **File→Save** or **File→Save As....**

18.5.2 Gas Models

A gas model is needed by the Navier-Stokes (NS) physical model and contains gas species, gas reactions, and surface reactions. The gas reactions are necessary if equilibrium or finite-rate chemistry is to be included (see Sec. 11.2.5 on pg. 168). Surface reactions are used with select boundary conditions that solve for surface mass/energy balances.

When a gas model is selected, a summary of the model is presented, as in Fig. 18.35 on pg. 412. In addition to presenting the number of species and reactions present for the model, the summary also indicates the availability of thermodynamic and transport property models. All species present in the model must have a particular set of modeling coefficients in order for that feature to be available for use in the selected gas model. For example, under

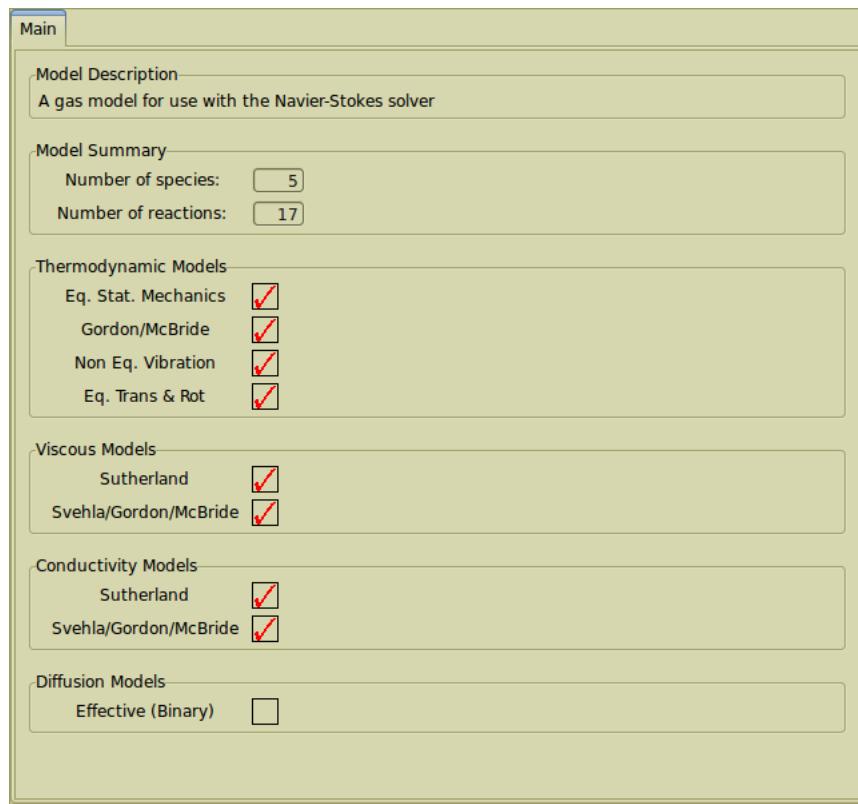


Figure 18.35: The Main Gas Model Frame.

the Thermodynamics Models section, the check box will be selected under Gordon/McBride if there are Gordon/McBride thermodynamics coefficients present for each species in the model.

Note: If non-equilibrium vibrational energy is being solved for, then those species with vibrational energy (*i.e.*, polyatomic species) must be ordered first in the species list. Also, if solving for electrons (e^-), the electron species should be placed last in the species list.

18.5.3 Solid Models

A solid model is needed by the solid thermodynamics (ST) physical model, the material stress (MS) physical model, or the Lagrange Spray physical model if tracking solid particles. The solid model should contain a single solid species.

Solid Stress Properties

Every solid model has a stress property frame which contains input related to material stresses, as displayed in the Stress Properties frame, Fig. 18.36 on pg. 413. The information

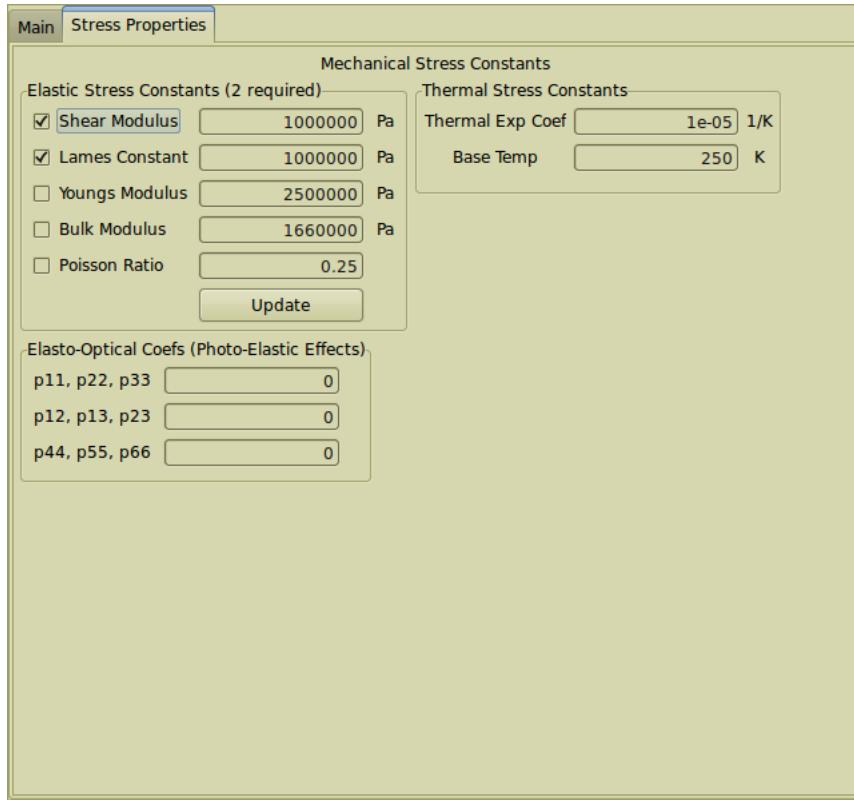


Figure 18.36: The Stress Properties Frame.

in this frame is used primarily by the material stress solver (via a material stress physical model).

The first set of inputs in this frame are directly related to the mechanical stress. The **Shear Modulus** (also known as the modulus of rigidity, G) is defined as the ratio of shear stress to shear strain. The data must be given in the metric units of Pascal.

The next input is **Lames Constant** (given the symbol λ) which is an elastic constant commonly used in Hooke's law. Both the shear modulus G and Lames constant λ are used to relate stress to strain in an isotropic, elastic material. The units of Lames constant must be given in Pascal.

A number of other properties are commonly used in place of G and λ . These include the Bulk modulus K , Young's modulus E , and Poisson's ratio ν . Conversions between these parameters and those input in the database (G and λ) can easily be found online.

Forces acting upon a solid create stresses which may in turn cause changes in the material shape (deformations). Changes in temperature may also create deformations. A parameter which helps model this process is the **Thermal Exp Coef** (thermal expansion coefficient). The units of this coefficient are per degree Kelvin (or per Celsius).

The **Base Temp** input represents the base temperature in Kelvin where a material has zero thermal stress.

The elasto-optical coefficients are used to model photo-elastic effects contributing to optical birefringence. The inputs are only used in computing the index of refraction. For most common materials a certain degree of symmetry exists for the elasto-optical tensor. Using the reduced notation by Nye [67], the tensor inputs can be reduced to three independent coefficients for isotropic or cubic crystal materials.

18.5.4 Liquid Models

A liquid model is needed by the Lagrange Spray physical model if tracking liquid particles. A liquid model may also be used with the Navier Stokes Solver, when a liquid model is selected, instead of a gas model. The liquid model should contain a single liquid species.

Bibliography

- [1] Vincenti, W. G. and Kruger, C. H., **Introduction to Physical Gas Dynamics**, Robert E. Krieger Publishing Company, Malabar, Florida, 1965.
- [2] Svehla, R. A. and McBride, B. J., “FORTRAN IV Computer Program for Calculation of Thermodynamic and Transport Properties of Complex Chemical Systems,” NASA TN D-7056, 1973.
- [3] Eppard, M., McGrory, W., and Applebaum, M., “The Effects of Water-Vapor Condensation and Surface Catalysis on COIL Performance,” AIAA Paper 2002-2132, May, 2002.
- [4] Masuda, W., Satoh, M., and Yamada, H., “Effects of Water Vapor Condensation on the Performance of Supersonic Flow Chemical Oxygen-Iodine Laser,” *JSME Int J.*, Vol. 39, No. 2, 1996, pp. 273–279.
- [5] Perrell, E., Candler, G., and Erickson, W., “Computation of Hypersonic Flows with Finite Rate Condensation and Evaporation of Water,” AIAA Paper 93-0796, Jan, 1993.
- [6] Perrell, E., Erickson, W., and Candler, G., “Numerical Simulation of Nonequilibrium Condensation in a Hypersonic Wind Tunnel,” *Journal of Thermophysics and Heat Transfer*, Vol. 10, No. 2, 1996.
- [7] Kang, S.-W., Dunn, M. G., and Jones, W. L., “Theoretical and Measured Electron-Density Distributions for the RAM Vehicle at High Altitudes,” AIAA Paper 72-689, June, 1972.
- [8] Park, C., “On Convergence of Computation of Chemically Reacting Flows,” AIAA Paper 85-0247, January 14-17, 1985.
- [9] Park, C., “Review of Chemical-Kinetic Problems of Future NASA Missions, I: Earth Entries,” *Journal of Thermophysics and Heat Transfer*, Vol. 7, No. 3, 1993, pp. 385–398.
- [10] Evans, J. S. and Schexnayder, C. J., “Influence of Chemical Kinetics and Unmixedness on Burning in Supersonic Hydrogen Flames,” *AIAA Journal*, Vol. 18, No. 2, 1979, pp. 188–193.

- [11] Drummond, J. P., Rogers, R. C., and Hussaini, M. Y., "A Detailed Numerical Model of a Supersonic Reacting Mixing Layer," AIAA Paper 86-1427, June 16-18, 1986.
- [12] Drummond, J. P. and Mekunda, H. S., "A Numerical Study of Mixing Enhancement in Supersonic Reacting Flow Fields," AIAA Paper 88-3260, July 11-13, 1988.
- [13] Drummond, J. P., Carpenter, M. H., Riggins, D. W., and Adams, M. S., "Mixing Enhancement in a Supersonic Combustor," AIAA Paper 89-2794, July 10-12, 1989.
- [14] Baulch, D. L., Drysdale, D. D., Horne, D. G., and Lloyd, A. C., "Evaluated Kinetic Data for High Temperature Reactions, Vol. I Homogeneous Gas Phase Reactions of the $H_2 - O_2$ System," 1972.
- [15] Baulch, D. L., Drysdale, D. D., Horne, D. G., and Lloyd, A. C., "Evaluated Kinetic Data for High Temperature Reactions, Vol. II Homogeneous Gas Phase Reactions of the $H_2 - O_2$ System," 1973.
- [16] Glass, I. I. and Takano, A., "Nonequilibrium Expansion Flows of Dissociated Oxygen and Ionized Argon Around a Corner," *Progress in Aerospace Sciences*, Vol. **6**, June 1965, pp. 163–249.
- [17] Igra, O. and Barcessat, M., "Supersonic Nonequilibrium Corner Expansion Flow of Ionized Argon," *The Physics of Fluids*, Vol. **20**, No. 9, September 1977, pp. 1449–1457.
- [18] Baurle, R. A., Mathur, T., Gruber, M. R., and Jackson, K. R., "A Numerical and Experimental Investigation of a Scramjet Combustor for Hypersonic Missile Applications," AIAA Paper 98-3121, Jul. 1998, Joint Propulsion Conference, Cleveland Ohio.
- [19] Liou, M.-S., "A Sequel to AUSM: AUSM+," *Journal of Computational Physics*, Vol. 129, 1996, pp. 364–382.
- [20] Weiss, J. M. and Smith, W. A., "Preconditioning Applied to Variable and Constant Density Flows," *AIAA Journal*, Vol. 33, No. 11, 1995, pp. 2050–2057.
- [21] Park, S. H. and Kwon, J. H., "An Improved HLLE Method for Hypersonic Viscous Flows," AIAA Paper 2001-2633, June 11-14, 2001.
- [22] van Albada, G. D., van Leer, B., and Roberts, W. W., "A Comparative Study of Computational Methods in Cosmic Gas Dynamics," *Astronomy and Astrophysics*, Vol. **108**, 1982, pp. 76–84.
- [23] Hirsch, C., **Numerical Computation of Internal and External Flows, Volumes 1 and 2**, John Wiley and Sons, New York, 1992.
- [24] Godfrey, A. G., Mitchell, C. R., and Walters, R. W., "Practical Aspects of Spatially High Accurate Methods," AIAA Paper 92-0054, January 6-9, 1992.

- [25] Roe, P., "Some Contributions to the Modeling of Discontinuous Flows," *Proc. 1983 AMS-SIAM Summer Seminar on Large Scale Computing in Fluid Mechanics, Lectures in Applied Mathematics*, Vol. **22**, 1985, pp. 163–193.
- [26] Spekreijse, S., "Multigrid Solution of Monotone Second-Order Discretizations of Hyperbolic Conservation Laws," *Mathematics of Computation*, Vol. **49**, No. 179, 1987, pp. 135–155.
- [27] Barth, T. J. and Jespersen, D. C., "The Design and Application of Upwind Schemes on Unstructured Meshes," AIAA Paper 89-0366, January 9-12, 1989.
- [28] Venkatakrishnan, V., "On the Accuracy of Limiters and Convergence of Steady-State Solutions," AIAA Paper 93-0880, January 11-14, 1993.
- [29] Wilke, C. R., "A Viscous Equation for Gas Mixtures," *Journal of Computational Physics*, Vol. **18**, No. 4, 1950, pp. 517–519.
- [30] White, F. M., **Viscous Fluid Flow**, McGraw-Hill, Inc., New York, NY, 1974.
- [31] Gupta, R. N., Yos, J. M., Thompson, R. A., and Lee, K.-P., "A Review of Reaction Rates and Thermodynamic and Transport Properties for the 11-Species Air Model for Chemical and Thermal Nonequilibrium Calculations to 30,000 K," NASA TM-101528, 1989, Also NASA Reference Publication RP-1232, August 1990.
- [32] Wright, M. J., Bose, D., Palmer, G. E., and Levin, E., "Recommended collision integrals for transport property computations, Part 1: Air species," *AIAA Journal*, Vol. 43, No. 12, 2005, pp. 2558–2564.
- [33] Spalart, P. R. and Allmaras, S. R., "A One Equation Turbulence Model for Aerodynamic Flows," *La Recherche Aerospatiale*, Vol. 1, 1994, pp. 5–21.
- [34] Baldwin, B. S. and Lomax, H., "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows," AIAA Paper 78-257, January 16-18, 1978.
- [35] Spalart, P. R., "Trends in Turbulence Treatments," AIAA Paper 2000-2306, Jun. 2000, Fluids 2000 Conference.
- [36] Spalart, P. R., Deck, S., Shur, M. L., Squires, K. D., Strelets, M. K., and Travin, A., "A new version of detached-eddy simulation, resistant to ambiguous grid densities," *Theor. Comput. Fluid Dyn.*, Vol. 20, 2006, pp. 181–195.
- [37] Lam, C. K. G. and Bremhorst, K., "A Modified Form of the $K-\epsilon$ Model for Predicting Wall Turbulence," *Journal of Fluids Engineering*, Vol. **103**, 1981, pp. 456–460.
- [38] Chien, K.-Y., "Predictions of Channel and Boundary-Layer Flows with a Low-Reynolds-Number Turbulence Model," *AIAA Journal*, Vol. **20**, No. 1, 1982, pp. 33–38.

- [39] Coakley, T. J. and Huang, P. G., "Turbulence Modeling For High Speed Flows," AIAA Paper 92-0436, January, 1992.
- [40] Wilcox, D. C., **Turbulence Modeling for CFD**, DCW Industries, Inc., La Cañada, CA, 1993.
- [41] Wilcox, D. C., *Turbulence Modeling for CFD*, DCW Industries, 2nd ed., 1998.
- [42] Wilcox, D. C., *Turbulence Modeling for CFD*, DCW Industries, 3rd ed., 2006, ISBN 978-1-928729-08-2.
- [43] Menter, F. R., "Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications," *AIAA Journal*, Vol. **32**, No. 8, Aug. 1994, pp. 1598–1605.
- [44] Sainte-Rose, B., Bertier, N., Deck, S., and Dupoirieux, F., "Delayed Detached Eddy Simulation of a Premixed Methane-Air Flame behind a Backward-Facing Step," AIAA Paper 2008-5134, 2008.
- [45] Menter, F. R. and Egorov, Y., "A Scale-Adaptive Simulation Model using Two-Equation Models," AIAA Paper 2005-1095, Jan. 2005.
- [46] Davidson, L., "The SAS model: A Turbulence Model with Controlled Modelled Dissipation," 20th Nordic Seminar on Computational Mechanics, Nov, 2007.
- [47] Gieseking, D. A., Choi, J., Edwards, J. R., and Hassan, H. A., "Compressible-Flow Simulations Using a New Large-Eddy Simulation/Reynolds-Averaged Navier-Stokes Model," *AIAA Journal*, Vol. 49, No. 10, Oct. 2011, pp. 2194–2209.
- [48] Gieseking, D. A. and Edwards, J. R., "Study of a Compression-Ramp Interaction Using Large-Eddy Simulation/Reynolds-Averaged Navier-Stokes Models," *AIAA Journal*, Vol. 50, No. 10, Oct. 2012, pp. 2057–2269.
- [49] Menter, F. R., Langtry, R., and Volker, S., "Transition Modelling for General Purpose CFD Codes," *Flow Turbulence Combustion*, Vol. **77**, 2006, pp. 277–303.
- [50] Menter, F. R., Langtry, R., Likki, S. R., Suzen, Y. B., Huang, P., and Volker, S., "A Correlation-Based Transition Model Using Local Variables Part I: Model Formulation," *Journal of Turbomachinery*, Vol. **128**, 2006, pp. 413–422.
- [51] Smith, B., "A Near Wall Model for the k-l Two Equation Turbulence Model," AIAA 94-2386, 1994.
- [52] Goldberg, U. and Chakravarthy, S., "A k- Turbulence Closure for Wall-Bounded Flows," AIAA 2005-4638, 2005.
- [53] Tabakoff, W., "Measurements of Particles Rebound Characteristics on Materials Used in Gas Turbines," *AIAA Journal Propulsion and Power*, Vol. 7, No. 5, 1991, pp. 805–813.

- [54] Loth, E., "Compressibility and Rarefaction Effects on Drag of a Spherical Particle," *AIAA Journal*, Vol. 46, No. 9, Sep. 2008, pp. 2219–2228.
- [55] Balay, S., Abhyankar, S., Adams, M., Brown, J., Brune, P., Buschelman, K., Eijkhout, V., Gropp, W., Kaushik, D., Knepley, M., McInnes, L., Rupp, K., Smith, R., and Zhang, H., *PETSc Users Manual, Revision 3.5*, Argonne National Laboratory, 2014, ANL-95/11.
- [56] Schroeder, W., Martin, K., and Lorensen, B., *The Visualization Toolkit An Object-Oriented Approach To 3D Graphics*, Kitware, Inc., 4th ed., 2005, ISBN 1-930934-19-X.
- [57] Roth, M., *Automatic extraction of vortex core lines and other line-type features for scientific visualization*, Ph.D. thesis, ETH, Zurich (Switzerland), 2000.
- [58] Haimes, R. and Kenwright, D., "On the Velocity Gradient Tensor and Fluid Feature Extraction," AIAA Paper 99-3288, 1999.
- [59] Jeong, J. and Hussain, F., "On the identification of a vortex," *Journal of Fluid Mechanics*, Vol. 285, Feb. 1995, pp. 69–94.
- [60] Pitalo, G., "A method for finding, T_0 , S , μ_0 and k_0 ," private communication.
- [61] Eppard, M., McGrory, W., Godfrey, A., Cliff, E., and Borggaard, J., "Recent Advances in Numerical Techniques for the Design and Analysis of COIL Systems," AIAA Paper 2000-2576, June, 2000.
- [62] Westbrook, C. K. and Dryer, F. L., "Simplified Reaction Mechanisms for the Oxidation of Hydrocarbon Fuels in Flames," *Combustion Science and Technology*, Vol. **27**, 1981, pp. 31–43.
- [63] Magnussen, B. F., "On the Structure of Turbulence and a Generalized Eddy Dissipation Concept for Chemical Reaction in Turbulent Flow," AIAA Paper 81-0042, Jan, 1981.
- [64] Magnussen, B. F. and Hjertager, B. H., "On Mathematical Modeling of Turbulent Combustion with Special Emphasis on Soot Formation and Combustion," *In Sixteenth Symposium (International) on Combustion*, 1976, pp. 719–729, The Combustion Institute.
- [65] Park, C., Jaffe, R. L., and Partridge, H., "Chemical-Kinetic Parameters of Hyperbolic Earth Entry," *Journal of Thermophysics and Heat Transfer*, Vol. **15**, No. 1, 2001, pp. 76–90.
- [66] Park, C., **Nonequilibrium Hypersonic Aerothermodynamics**, John Wiley and Sons, New York, 1990.
- [67] Nye, J. F., **Physical Properties of Crystals: Their Representation by Tensors and Matrices**, Oxford at the Clarendon Press, 1957, ISBN 13: 978-0-19-851165-6.