TRUCHAS Reference Manual

The TELLURIDE Team

Version 2.4.0
February 14, 2008

LA-UR-08-0816

The TRUCHAS Reference Manual provides a comprehensive summary for the executable command-line invocation and namelist input variable attributes. It is intended as the central repository of defined defaults and also all possible values for input and command variables.

LA-UR-08-0816

Contents

Contents i

1 Invocation 1

1.1 Serial Invocation 2

1.2 Parallel Invocation 3

1.2.1 Linux/LAM 3

1.2.2 Linux/MPICH 3

1.2.3 Linux/Scali 4

1.2.4 Tru64/native MPI 4

1.2.5 Irix/native MPI 4

1.2.6 AIX/native MPI 4

2 `Introduction:` Namelist Variables 7

3 `ALTMESH` Namelist 9

Overview 9

Components 9

Altmesh_Coordinate_Scale_Factor 10

Altmesh_File 10

Grid_Transfer_File 10

Parallel 11

4 BC Namelist **13**

Overview 13

Conic Equation 13

Components 14

BC_Name 15

BC_Type 15

BC_Use_Function 17

BC_Value 17

BC_Variable 19

Bounding_Box 19

Conic_Constant 19

Conic_Relation 19

Conic_Tolerance 20

Conic_X 20

Conic_XX 21

Conic_XY 21

Conic_XZ 21

Conic_Y 21
Conic_YY 22
Conic_YZ 22
Conic_Z 22
Conic_ZZ 23
Inflow_Concentration 23
Inflow_Material 23
Inflow_Temperature 23
Mesh_Surface 24
Node_Disp_Coords 24
Surface_Materials 24
Surface_Name 25
T0_Convection 25

5 BODY Namelist **27**

Overview 27
Components 27
Axis 28
Concentration 28
File_Name 29
Fill 29
Height 29

Length 30
Material_Number 30
Mesh_Material_Number 30
Radius 30
Rotation_Angle 31
Rotation_Pt 31
Rtheta_Tabular_Pt 31
RZ_Tabular_Pt 32
Surface_Name 32
Tabular_Type 32
Temperature 32
Translation_Pt 33
Velocity 33
TG_Origin 33
TG_Axis 34
TG_Z_Constants 34
TG_Z_Exponents 34
TG_Z_Bounds 34
TG_R_Constants 35
TG_R_Exponents 35
TG_R_Bounds 35

6 CHEMICAL_REACTION Namelist **37**

Overview 37

Components 38

CR_Active 38

CR_m 39

CR_n 39

CR_Cmax 39

CR_ko1 39

CR_ko2 39

CR_Ea1 40

CR_Ea2 40

CR_Htot 40

CR_Reactants_id 40

CR_Products_id 41

7 DIFFUSION_SOLVER Namelist **43**

Overview 43

Components 44

Abs_Conc_Tol 44

Abs_Enthalpy_Tol 45

Abs_Temp_Tol 45

AIN_Preconditioner 46

AIN_Tol 46
AIN_Vec_Tol 47
Max_AIN_Itr 47
Max_AIN_Vec 47
Max_Step_Tries 48
PC_AMG_Cycles 48
PC_SSOR_Relax 48
PC_SSOR_Sweeps 49
Rel_Conc_Tol 49
Rel_Enthalpy_Tol 50
Rel_Temp_Tol 50
System_Type 50
Verbose_Stepping 51

8 DS_SPECIES_BC Namelist **53**

Overview 53
Components 53
BC_Type 54
BC_Value 54
Face_Set_IDs 54
Name 55

9 DS_TEMPERATURE_BC Namelist **57**

Overview 57
Components 57
BC_Type 58
BC_Value 58
Face_Set_IDs 58
Name 59

10 ELECTROMAGNETICS Namelist **61**

Overview 61
Components 62
CG_Stopping_Tolerance 62
EM_Domain_Type 63
Epsilon_0 63
Graphics_Output 64
Material_Change_Threshold 64
Maximum_CG_Iterations 65
Maximum_Source_Cycles 65
Mu_0 65
Num_Etasq 66
Output_Level 66
Source_Frequency 67
Source_Times 67

SS_Stopping_Tolerance 68

Steps_Per_Cycle 68

Symmetry_Axis 69

Uniform_Source 69

11 HEAT_SOURCE Namelist **71**

Overview 71

Components 72

cell_set_id_list 73

heat_source_type 73

cell_set_id_list 73

Volumetric_Heat_Source 73

heat_source_center 73

heat_source_constant 74

heat_source_radius 74

heat_source_time 74

travel_speed 75

volumetric_heat_source 75

12 INDUCTION_COIL Namelist **77**

Overview 77

Components 78

Center 78

Current 79

Length 79

NTurns 79

Radius 79

13 INTERFACES Namelist **81**

Overview 81

Components 81

Vof_Method 82

Int_Particles 82

Vof_Particles 82

Vof_Tolerance 83

Vof_Max_Recursion 83

14 LINEAR_SOLVER Namelist **85**

Overview 85

Components 85

Convergence_Criterion 86

Krylov_Vectors 86

Maximum_Iterations 87

Method 87

Name 88

Output_Mode 89

Preconditioning_Method 89

Preconditioning_Preconditioner 90

Preconditioning_Steps 90

Preconditioning_Scope 91

Relaxation_Parameter 91

Status_Frequency 92

Stopping_Criterion 92

15 MATERIAL Namelist **95**

Overview 95

Components 97

CTE_Conc_Bounds 100

CTE_Constants 100

CTE_Exponents 100

CTE_Relation 101

CTE_Temp_Bounds 101

Conductivity_Conc_Bounds 101

Conductivity_Constants 102

Conductivity_Exponents 102

Conductivity_Relation 102

Conductivity_Temp_Bounds 103

Cp_Conc_Bounds 103

Cp_Constants 104
Cp_Exponents 104
Cp_Relation 104
Cp_Temp_Bounds 105
Density 105
Density_Change_Coefficients_C 105
Density_Change_Coefficients_T 106
Density_Change_Exponents_C 106
Density_Change_Exponents_T 106
Density_Change_Relation 107
Density_Conc_Bounds 107
Density_Temp_Bounds 108
EM_Conductivity_Conc_Bounds 108
EM_Conductivity_Constants 108
EM_Conductivity_Exponents 109
EM_Conductivity_Relation 109
EM_Conductivity_Temp_Bounds 109
EM_Permeability_Conc_Bounds 110
EM_Permeability_Constants 110
EM_Permeability_Exponents 110
EM_Permeability_Relation 111

EM_Permeability_Temp_Bounds 111
EM_Permittivity_Conc_Bounds 112
EM_Permittivity_Constants 112
EM_Permittivity_Exponents 112
EM_Permittivity_Relation 113
EM_Permittivity_Temp_Bounds 113
Immobile 113
Lame1_Conc_Bounds 114
Lame1_Constants 114
Lame1_Exponents 115
Lame1_Relation 115
Lame1_Temp_Bounds 115
Lame2_Conc_Bounds 116
Lame2_Constants 116
Lame2_Exponents 116
Lame2_Relation 117
Lame2_Temp_Bounds 117
Material_Feature 118
Material_Name 118
Material_Number 118
MTS_b 119

MTS_d 119
MTS_edot_0i 119
MTS_g_0i 120
MTS_k 120
MTS_mu_0 120
MTS_p_i 121
MTS_q_i 121
MTS_sig_a 121
MTS_sig_i 121
MTS_temp_0 122
Permeability_Constant 122
Priority 122
Pwr_Law_A 123
Pwr_Law_N 123
Pwr_Law_Q 123
Pwr_Law_R 124
Reference_Temperature 124
Reference_Concentration 124
Sound_Speed 125
SPC_Diffusivity_Constants 125
SPC_Diffusivity_Exponents 125

SPC_Diffusivity_Relation 126
Stress_Reference_Temperature 126
Surf10_Conc_Bounds 127
Surf10_Constants 127
Surf10_Exponents 127
Surf10_Material_Number 128
Surf10_Relation 128
Surf10_Temp_Bounds 129
Transmits_Radiation 129
Viscoplastic_Model 129
Viscosity_Conc_Bounds 129
Viscosity_Constants 130
Viscosity_Exponents 130
Viscosity_Relation 130
Viscosity_Temp_Bounds 131
Void_Temperature 131

16 MESH Namelist **133**

Overview 133
Components 133
NCell 134
Coord 134

Ratio 134

Fuzz 135

Gap_Element_Blocks 135

Heps 136

Mesh_File 136

Mesh_File_Format 136

Coordinate_Scale_Factor 137

Partitions_Total 137

Partitions_Per_Process 137

17 NONLINEAR_SOLVER Namelist **139**

Overview 139

Components 139

AIN_Max_Vectors 140

AIN_Vector_Tolerance 140

Convergence_Criterion 141

Damper_Parameters 142

Linear_Solver_Name 142

Maximum_Iterations 143

Method 143

Name 143

Perturbation_Parameter 144

Use_Damper 144

18 NUMERICS Namelist **145**

Overview 145

Components 145

advection_order_energy 147

advection_order_momentum 148

Alittle 148

Body_Force_Face_Method 148

Body_Force_Implicitness 149

CSF_Mollify_Passes 149

Conduction_Implicitness 149

Contact_Distance 150

Contact_Norm_Trac 151

Contact_Penalty 151

Courant_Number 152

Cutvof 152

Cycle_Max 153

Cycle_Number 153

Discrete_Ops_Type 153

Displacement_Nonlinear_Solution 154

Dt_Constant 155

Dt_Grow 155
Dt_Init 155
Dt_Max 156
Dt_Min 156
Enclosure_Flux_Convergence 157
Enclosure_Implicit_Solution 157
Enclosure_Max_Iterations 158
Enclosure_Use_Cell_Temps 158
Energy_Nonlinear_Solution 158
Energy_Sensitivity_Solution 159
Energy_Sensitivity_Pseudo_Load 159
FF_Discrete_Ops_Type 159
Flux_Vol_Iter_Max 160
Fourier_Number 160
HT_Discrete_Ops_Type 161
Interface_Area 162
Interface_Geometry 162
Interface_Smoothing_Length 163
Interface_Topology_Model 163
Limiter_Type 164
Mass_Limiter 164

Mass_Limiter_Cutoff 165
Mechanical_Energy_Bound 165
MinFaceFraction 166
Momentum_Solidify_Implicitness 166
Porous_Implicitness 166
Projection_Linear_Solution 167
Sfront_Number 167
Strain_Limit 168
Surften_Number 168
t 169
Temp_Init_Iter_Max 169
Viscous_Implicitness 170
Viscous_Linear_Solution 170
Viscous_Number 170
Volume_Track_Brents_Method 171
Volume_Track_Interfaces 172
Volume_Track_Iter_Max 172
Volume_Track_Iter_Tol 172
Volume_Track_Subcycles 173

19 OUTPUTS Namelist **175**

Overview 175

Components 175
Enclosure_Diagnostics 176
Int_Output_Dt_Multiplier 176
Long_Edit_Bounding_Coords 176
Long_Output_Dt_Multiplier 177
Output_Dt 177
Output_T 177
Precise_Output 178
Probe_Output_Cycle_Multiplier 178
Retain_Last_Step 178
Short_Output_Dt_Multiplier 178
User_Output_Dt_Multiplier 179
XML_Output_Dt_Multiplier 179

20 PARALLEL_PARAMETERS Namelist **181**

Overview 181
Components 182
Partitioner 182
Processor_Array 182

21 PHASE_CHANGE_PROPERTIES Namelist **183**

Overview 183
Components 183

a2_coeff 184
a3_coeff 185
deltempnucl 185
deltempsigma 185
ndmaxnucl 186
full_coupling_flag 186
alphaCnK 186
Back_Diffusion_Coefficient 187
Eutectic_Concentration 187
Eutectic_Temperature 187
Forward_Diffusion_Coefficient 187
Hi_Temp_Phase_Material_ID 188
Latent_Heat 188
Liquidus_Slope 188
Liquidus_Temp 189
Lo_Temp_Phase_Material_ID 189
Melting_Temperature_Solvent 189
Melting_Temperature 189
Partition_Coefficient_Constants 190
Partition_Coefficient_Exponents 190
Partition_Coefficient_Relation 190

Phase_Change_Active 191

Phase_Change_Type 191

Phase_Change_Model 191

Secondary_Half_Arm_Spacing 192

Solidification_Time 192

22 PHYSICS Namelist **193**

Overview 193

Components 193

Absolute_Zero 194

Body_Force 194

CSF_Normal 195

CSF_Tangential 195

Curvature_Model 195

Diffusion_Solver 196

Electromagnetics 196

Fluid_Flow 196

Heat_Conduction 196

Inviscid 197

Phase_Change 197

Porous_Flow 197

Solid_Mechanics 198

Solid_Mechanics_Phase_Change 198
Stefan_Boltzmann 198
Stokes 199
Surface_Tension_Model 199
Turbulence_CMU 199
Turbulence_KE_Fraction 200
Turbulence_Length 200
Turbulence_Model 200
Void_Pressure 201

23 PROBE Namelist **203**

Overview 203
Components 203
Probe_Name 204
Probe_Description 204
Probe_Coords 204
Probe_Coords_Scale 204

24 RADIATION_ENCLOSURE Namelist **207**

Overview 207
Components 208
EnclosureID 208
Method 208

inFile 209
inFileType 210
outFile 210
outFileType 210
Comment 210
Partial 211
LinearSolver 211
chaparral_Amax 211
chaparral_iParam 212
chaparral_rParam 213
chaparral_symmetry 214

25 RESTART Namelist **217**

Overview 217
Components 217
Ignore_Dt 218
Ignore_Joule_Heat 218
Ignore_Solid_Mechanics 218
Temperature_Is_Primary 218

26 REGION Namelist **221**

Overview 221
Components 221

x1 222
y1 222
z1 222
x2 223
y2 223
z2 223
flow_off 223

27 SENS_VARIABLE Namelist **225**

Overview 225
Components 226
Sens_Variable_Active 226
Sens_Variable_Component 226
Sens_Variable_Identifier 227
Sens_Variable_Iparam 227
Sens_Variable_Lower 228
Sens_Variable_Name 228
Sens_Variable_Pert 228
Sens_Variable_Type 229
Sens_Variable_Upper 229
Sens_Variable_Value 229

28 SENS_FUNCTION Namelist **231**

Overview 231

Components 231

Sens_Function_Active 232

Sens_Function_Iparameters 232

Sens_Function_Name 232

Sens_Function_Rparameters 232

Sens_Function_Type 233

29 SO_SOLVER Namelist **235**

Overview 235

Components 235

Convergence_Criterion 236

Maximum_Iterations 236

Name 236

Output_Mode 237

Stopping_Criterion 237

Bibliography **239**

Index **240**

Chapter 1

Invocation

This section of the reference manual describes the invocation of the TRUCHAS executable.

The format of a TRUCHAS command line looks like:

```
executable [-options] inputfile
```

An input file name is required. The input file name must end in .inp. The suffix .inp may be omitted from the command line.

The naming scheme for TRUCHAS executables is to name them:

```
t-$OS.$PARALLELISM.$DEBUG-$VERSION
```

where the words mean:

$OS	operating system
$PARALLELISM	serial or parallel
$DEBUG	debug or opt (optimized)
$VERSION	version number

This allows us to invoke binaries built for serial or parallel execution, and compiled with or without debugging and optimization, for multiple platforms.

Options are single letters. Options must be specified individually, seperated by white space. Options can not be combined into one argument. Accepted command line options are:

-h	help
-d[:n]	debug level
-o:filenameroot	set output file name root
-r:filename	restart from restart file name

-h prints a summary of the command line and exits.

-d:n controls the amount of debug output that is printed. n may have the values 0, 1, 2. The default is -d:0, or no debug output. -d:1 prints debugging output. -d:2 prints more detailed debugging output. -d is the same as -d:1.

-o:filenameroot sets the output file name root. Normally, the output file names are constructed from the pathname of the input file, omitting the .inp suffix. -o:filenameroot is used to change the root filename and/or directory where the output files are written. This is useful when running several cases from the same input file, or when doing restarts.

-r:filename runs Truchas in restart mode, restarting from the data in filename. filename must have been created as a restart dump, originally written by Truchas as specified in an input file.

1.1 Serial Invocation

Invoking TRUCHAS in serial requires specification of the executable and the input file:

```
t-$OS.$PARALLELISM.$DEBUG-$VERSION inputfile.inp
```

Examples:

Run TRUCHAS in serial with optimized code code on linux, using an input file named flow.inp:

```
t-linux.serial.opt-$VERSION flow.inp
```

Run TRUCHAS in serial with optimized code code on linux, using an input file from the TRUCHAS test suite (named box_fill_ortho.inp), with both the executable and the test problem in their standard places in the distribution:

```
bin/t-linux.serial.opt-$VERSION problems/tests/box_fill_ortho.inp
```

Run TRUCHAS in serial with debug code on linux, using an input file named flow.inp:

```
t-linux.serial.debug-$VERSION flow.inp
```

Run TRUCHAS in serial with optimized code code on linux, using an input file named flow.inp and writing the output files to flow-new.*:

```
t-linux.serial.opt-$VERSION -o:flow-new flow.inp
```

1.2 Parallel Invocation

Parallel invocation depends on the particular computer system and how it invokes MPI jobs. On most systems, simply preface a normal truchas command line with mpirun and appropriate options. You must invoke an executable which was built to run in parallel. On platforms where several MPI implementations are available, make sure you use the mpirun (or similar) command that goes with the MPI implementation that the code was built with.

1.2.1 Linux/LAM

LAM is the default MPI on Linux. LAM requires a machinefile to describe the parallel machine. The file can be located anywhere, and is named when the cluster is booted via the lamboot command. The contents of the file are lines that name the machines in the cluster and the number of cpus to use on each machine, as in this example for 2 machines, each with 2 cpus:

```
machine1 cpu=2
machine2 cpu=2
```

To run TRUCHAS with LAM, the LAM daemons must be started (using the lamboot command). This starts a daemon on each machine listed in the machinefile. The parallel jobs are run, and the daemons are halted:

```
lamboot -v machinefile
mpirun -np 4 t-linux.parallel.opt-$VERSION inputfile.inp
lamhalt
```

1.2.2 Linux/MPICH

MPICH requires a machinefile to describe the parallel machine. The file can be located anywhere, and is named whenever the parallel jobs are started. The contents of the file are lines that name the machines in the cluster and the number of cpus to use on each machine, as in this example for 2 machines, each with 2 cpus:

```
machine1:2
machine2:2
```

To run TRUCHAS with MPICH:

```
mpirun -np 4 -machinefile machinefile
t-linux.parallel.opt-$VERSION inputfile.inp
```

1.2.3 Linux/Scali

Scali MPI is an MPI implementation often used with Dolphinics SCI interconnect hardware. Daemons are started at system boot time. To run TRUCHAS with Scali:

```
mpimon t-linux.parallel.opt-$VERSION inputfile.inp -- machine1 2 machine2
2
```

1.2.4 Tru64/native MPI

To run TRUCHAS with the Tru64 native MPI on a single SMP host:

```
mpirun -np 4 t-osf1.parallel.opt-$VERSION inputfile.inp
```

1.2.5 Irix/native MPI

To run TRUCHAS with the Irix native MPI on a single SMP host:

```
mpirun -np 4 t-irix.parallel.opt-$VERSION inputfile.inp
```

1.2.6 AIX/native MPI

Our experience with AIX is limited to the large ASCI machines at Lawrence Livermore National Laboratory. Parallel jobs are run with poe, the Parallel Operating Environment. You need to know what pool your nodes are in, how many nodes you want, and how many processors you want to use on each node. The following command will run TRUCHAS with the AIX native MPI in pool 0, on one node, with 4 processes:

```
poe t-aix.parallel.opt-$VERSION inputfile.inp -rmpool 0 -nodes 1 -procs
4
```

In all cases, refer to your parallel system documentation for more details.

Chapter 2

`Introduction:` Namelist Variables

TRUCHAS uses Fortran Namelist input to specify a problem to be calculated. Each Namelist group consists of a set of related input variables. A particular Namelist may be required or optional, and it may be invoked only once or multiple times. All Namelist groups are described in subsequent chapters, presented in alphabetical order.

In this manual, the physical dimensions are abbreviated as follows:

M = mass

L = length

t = time

E = energy

T = temperature

θ = angle

I = current

The dimension of every dimensional input variable is thus represented by a combination of the above symbols. Further, certain variables are initialized (”preset”) with values that are not likely to be chosen by users. The integer and real preset values are currently -1836547290 and $-1836547290 \times 10^{17}$, respectively. These values will trigger a fatal error if they are not redefined by the user and so are intended to alert the user to set physical values.

The user is expected to ensure consistencies among the units of all input variables. The only internal variable that assumes a set of units in TRUCHAS is the gas constant R required for chemical reactions. This inbuilt constant's value assumes the SI system. However, a few physical constants have been assigned default values in SI units. Examples of these are the Stefan-Boltzmann constant for thermal radiation and the permeability and permittivity of free space, used in electromagnetic calculations. If a system of units other than SI is chosen, the user must redefine these variables.

When "valid values" for the input variables are discussed in this manual, we will use the following nomenclature. Square brackets [and] are used to indicate that the range is inclusive of the value that is bracketed, while parentheses (and) are used to indicate that the range is exclusive of the value that are in parenthesis. For example, if X has a range of $(0.0, 1.0]$ then we mean that valid values fall in the range $0 < X \leq 1.0$. Similarly it should be noted that a valid value indicated by $(-\infty, \infty)$ means that the range of valid values is in between a large negative and large positive number, where the large number is within the range of the machine limits for the real numbers or integers as appropriate.

Chapter 3

ALTMESH Namelist

Overview

The ALTMESH namelist describes the secondary ("alternative") mesh which will be a tetrahedral mesh employed for TRUCHAS calculations involving electromagnetics.

ALTMESH Namelist Features

Required / Optional : Optional

Single / Multiple Instances : Single

Components

- Altmesh_Coordinate_Scale_Factor
- Altmesh_File
- Grid_Transfer_File
- Parallel

Altmesh_Coordinate_Scale_Factor

Description: If specified, all vertex coordinates are scaled by this factor.

Physical Dimension: N/A

Type: `real`

Default: 1.0

Valid values: $(0.0, \infty)$

Altmesh_File

Description: Specifies the path of the file from which to read the alternative (tetrahedral) mesh used by the electromagnetics solver. The file must be a native ExodusII/Genesis data file.

Type: case-sensitive string

Default: (none)

Valid values: An absolute path or relative path to the input file directory.

Grid_Transfer_File

Description: Certain fields must be mapped between the main and alternative meshes during the course of calculations involving electromagnetics. These mappings are accomplished using some fixed grid-mapping data that depend only on the two meshes. This optional variable specifies the path of a file containing this grid mapping data. If specified and if the file exists, it will be read and its mapping data checked to ensure that it corresponds to the two meshes being used. If it corresponds, the data will be used for the calculation. Otherwise, the grid mapping data is computed and written to the file `altmesh_mapping_data.bin` in the output directory for use in future calculations, avoiding a needless and potentially costly recomputation of the same data.

Type: case-sensitive string

Default: (none)

Valid values: An absolute path or relative path to the input file directory.

Note: The mapping data depends on the internal ordering of the nodes and cells of each mesh, in addition to the meshes themselves. Thus it is recommended that this file be named in such a way that reflects the identity of the two meshes *and* the number of processors used to compute the mapping data; mapping data computed with one number of processors will not be usable in a calculation with a different number of processors, even when the same pair of meshes is used.

`Parallel`

Description: Controls the parallel execution of the electromagnetics solver. For a parallel execution of TRUCHAS, electromagnetic computations (only) will be done in serial if this flag is `.false.`

Type: `logical`

Default: `.true.`

Note: This flag has been retained for backward compatibility but will likely be removed in a future release. There should no longer be any difficulties running the electromagnetics solver in parallel.

Chapter 4

BC Namelist

Overview

The BC namelists define boundary conditions at external computational boundaries as well as at internal material interfaces.

The Boundary conditions are generally applied on a volume bounded by a surface defined by the conic function:

$$\begin{aligned} BC\ Value \quad & \{Conic_Relation[< | = | >]\}\ Conic_Constant \\ + \quad & Conic_X * X + Conic_Y * Y + Conic_Z * Z \\ + \quad & Conic_XX * X^2 + Conic_YY * Y^2 + Conic_ZZ * Z^2 \\ + \quad & Conic_XY * X * Y + Conic_XZ * X * Z + Conic_YZ * Y * Z \end{aligned} \tag{4.1}$$

where X, Y, and Z are the three coordinates and the rest of the terms are explained in the sections below. Any face center that lies within `Conic_Tolerance` of this surface is considered part of the boundary.

BC geometry may also be specified from side sets obtained from the mesh data file, from the interfaces between materials, or, in the case of solid mechanics BCs, as node sets. Additionally, the geometric region may be restricted to lie within a "bounding box", as described below.

BC Namelist Features

Required / Optional : Optional for free-slip flow only problems
Single / Multiple Instances : Multiple

Components

- BC_Name
- BC_Type
- BC_Use_Function
- BC_Value
- BC_Variable
- Bounding_Box
- Conic_Constant
- Conic_Relation
- Conic_Tolerance
- Conic_X
- Conic_XX
- Conic_XY
- Conic_XZ
- Conic_Y
- Conic_YY
- Conic_YZ
- Conic_Z
- Conic_ZZ
- Inflow_Concentration
- Inflow_Material
- Inflow_Temperature
- Mesh_Surface
- Node_Disp_Coords
- Surface_Materials
- Surface_Name
- T0_Convection

BC_Name

Description : The name of this BC namelist.

Physical Dimension : N/A

Type : string

Default : (none)

Valid values : Any string

Note : This is optional unless it is referred to by a Sens_Variable namelist.

BC_Type

Description : The type of boundary condition this BC namelist defines for the particular BC_Variable.

Physical Dimension : N/A

Type : string

Default : Depends on BC_Variable:

'temperature': (none)

'velocity': 'free-slip'

'pressure': (none)

'displacement': 'x-traction', 'y-traction', 'z-traction'

Valid values : Depends on BC_Variable:

'temperature': 'neumann', 'hneumann', 'dirichlet', 'htc', 'htc_gap', 'radiation', 'enclosure_radiation'

'velocity': 'free-slip', 'no-slip', 'dirichlet'

'pressure': 'dirichlet'

'displacement': 'x-traction', 'y-traction', 'z-traction', 'x-displacement', 'y-displacement', 'z-displacement', 'normal-displacement', 'normal-traction', 'free-interface', 'normal-constraint', 'contact'

Notes :

- A temperature solution requires that temperature boundary conditions be specified for all cell faces on the exterior mesh boundary.

- If `BC_Type 'htc'` is specified for an external interface, the energy flux into the cell at the boundary face, q is computed using a simple convection model

$$q = -\beta(T - T_0) \tag{4.2}$$

where T is the temperature at the boundary face, T_0 is the reference temperature specified in `T0_Convection` and β is the heat transfer coefficient specified in `BC_Value`.

- If `'htc'` or `'htc_gap'` is specified for an internal interface, a simple thermal resistance condition is applied of the form

$$q_1 = -\beta(T_1 - T_2) \tag{4.3}$$

where T_1 and T_2 are the temperatures at the boundary face for cells 1 and 2 on either side of the face, and q_1 is the heat flux at the boundary face coming into cell 1. β is the heat transfer coefficient specified in `BC_Value`.

- If an internal interface has gap elements, then a BC namelist with `BC_Type` = `'htc_gap'` must be defined for that surface.
- The `'radiation'` type calculates the flux q at an external boundary face using

$$q = -\epsilon k(T^4 - T_0^4) \tag{4.4}$$

where T is the temperature at the boundary face, ϵ and T_0 are the emmissivity and reference temperature specified in `BC_Value`, and k is the `Stefan_Boltzmann` constant. Note that `Stefan_Boltzmann` is a PHYSICS namelist variable and the default value is in Watt/m^{2}K^4. Further note that both temperatures are shifted by the value of `Absolute_Zero` specified in the PHYSICS namelist.

- The `'enclosure_radiation'` type calculates the flux q at an external boundary face using the temperatures on all other faces and the ambient temperature specified in the `BC_Value` array as:

$$q_i = -\epsilon_i k T_i^4 - (1 - \rho_i) \sum_j F_{ij} q_j \tag{4.5}$$

where ϵ_i is the emissivity of face i, T_i is the temperature at the boundary face, F_{ij} is the view factor of face j as seen from face i, and the sum on j is over all faces in the system and includes a special term for the contribution from the ambient environmant.

The first three entries in the `BC_Value` array correspond to `EnclosureID`, ϵ, and $T_{ambient}$ for the given surface. The value of k is obtained from the `Stefan_Boltzmann` constant. Note that `Stefan_Boltzmann` is a PHYSICS namelist variable and the default value is in Watt/m^{2}K^4. All temperatures are shifted by the value of `Absolute_Zero` specified in the PHYSICS namelist.

- The solid mechanics displacement solution defaults to a traction-free surface with no displacement constraints (`'x-traction', 'y-traction',` and `'z-traction'` set to zero.)

- The `'free-interface'`, `'normal-constraint'` and 'contact' types can only be specified for interfaces with gap elements.
- It is not permitted to specify a `'pressure' 'dirichlet'` condition and any type of `'velocity'` boundary condition on the same face.

BC_Use_Function

Description : Switch to activate the use of a function to specify values for `BC_Value` that can be dependent on space, time, or other variables.

Physical Dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.` or `.false.`

Note : This option is only currently implemented for heat transfer Dirichlet or Neumann boundaries. Stub versions are provided for the subroutines HT_SPECIFY_DIRICHLET and HT_SPECIFY_FLUX which are invoked for these boundary types.

Note : Parameter values, specified on the `BC_Value` record, are accessible in the subroutines via the `bc_use_function_array` array.

Caution : These stubs must be modified, compiled and linked into Truchas if this option is selected. The stub versions contain problem specific functions that will not work with most simulation specifications.

BC_Value

Description : Value(s) for the constant(s) used in this BC definition.

Physical Dimension : varies

Type : `real` (up to 24 values depending on `BC_Type`)

Default : 0.0

Valid values : $(-\infty, \infty)$

Notes : The meaning of the items in the BC_Value list depends on the particular boundary condition:

BC_Variable	BC_Type	Value Description	Physical Dimension	Number of values
'temperature'	'neumann'	energy flux [1]	E/L^2	1 – 23
'temperature'	'hneumann'	not used	N/A	0
'temperature'	'dirichlet'	temperature, [time, T data pairs] [2]	T, [t,T]	1 – 23
'temperature'	'htc'	energy flux / temperature difference	$E/(L^2T)$	1
'temperature'	'htc_gap'	flux / ΔT, flux / ΔT,distance	$E/(L^2T,)E/(L^2T),L$ [3]	3
'temperature'	'radiation'	emissivity, reference T, [time, T data pairs][4]	none, T, [t,T]	2 – 24
'temperature'	'enclosure_radiation'	Enclosure ID, emissivity , ambient Temperature,	none, none, T	3
'velocity'	'dirichlet'	velocity	L/T	3
'velocity'	'free-slip'	not used	N/A	0
'velocity'	'no-slip'	not used	N/A	0
'pressure'	'dirichlet'	pressure	$M/(Lt^2)$	1
'displacement'	'x-displacement'	displacement	L	1
'displacement'	'y-displacement'	displacement	L	1
'displacement'	'z-displacement'	displacement	L	1
'displacement'	'x-traction'	traction (force/area)	$M/(Lt^2)$	1
'displacement'	'y-traction'	traction (force/area)	$M/(Lt^2)$	1
'displacement'	'z-traction'	traction (force/area)	$M/(Lt^2)$	1
'displacement'	'normal-displacement'	displacement	L	1
'displacement'	'free-interface'	not used	N/A	0
'displacement'	'normal-constraint'	not used	N/A	0
'displacement'	'contact'	not used	N/A	0

1. If multiple values are specified and **if `BC_Use_Function is false`** then the first value is the reference flux and the rest are ignored. Otherwise **if `BC_Use_Function is true`** then the values are made accessible to subroutine HT_SPECIFY_DIRICHLET via the `bc_use_function_array` array.

2. If more than one value is specified for the `'dirichlet'` BC_Type for temperature, the first value is the reference temperature at time zero, and subsequent pairs of values are times and reference temperatures at those times. The temperatures at intermediate times are calculated by linear interpolation.

3. The three values for `'htc_gap'` are the heat transfer coefficient used when the gap is closed, the heat transfer coefficient used when the gap is open, and a distance parameter. When the gap displacement is zero or negative, the closed value is used and when the gap displacement is greater than the distance parameter the open value is used. Linear interpolation between these two conditions is used when the gap displacement is between zero and the distance parameter.

4. If more than two values are specified for the `'radiation'` BC_Type for temperature, the first value is the emissivity of the surface (which is constant), the second value is the reference temperature at time zero, and subsequent pairs of values are times and reference temperatures at those times. The reference temperatures at intermediate times are calculated by linear interpolation.

BC_Variable

Description : The solution variable this BC definition affects.

Physical Dimension : N/A

Type : string

Default : (none)

Valid values : `'temperature'`, `'velocity'`, `'pressure'`, `'displacement'`

Bounding_Box

Description : The minimum and maximum coordinates in each dimension limiting the extent of the surface space defined by either the `'conic'` function, `'material boundary'`, `'external material boundary'` or face set `'from mesh file'` as specified in `Surface_Name`. The limits are specified as $x_{min}, x_{max}, y_{min}, y_{max}, z_{min}, z_{max}$.

Physical Dimension : L

Type : `real` array with 6 elements as described above

Default : -10^{10} to 10^{10} for the limits in each dimension

Valid values : min $\leq$ max in each dimension

Conic_Constant

Description : Value of the constant term in the conic equation defining the BC surface. See `Conic Equation` for the form of the equation.

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Conic_Relation

Description : The equality or inequality relation used in the conic equation defining the BC surface. See

`Conic Equation` for the form of the equation.

Physical Dimension : N/A

Type : string

Default : (none)

Valid values : '=', '<', '>'

Conic_Tolerance

Description : The tolerance in the function value allowed when checking whether a face centroid lies on a BC surface. The value of the function at a face centroid is compared to the tolerance using `'='`, `'<'` or `'>'`, as specified in `Conic_Relation`. If `'='` is specified, the absolute value of the conic function must be less than the tolerance for the face to be included. For `'<'` or `'>'` the function must be less than or greater then the tolerance, respectively. See `Conic Equation` for the form of the equation.

Physical Dimension : N/A

Type : `real`

Default : 10^{-6}

Valid values : $(0.0, \infty)$

Notes :

- Note that the value of the tolerance is not always in units of length or length^2, since the tolerance is compared to the value of the conic function, not a distance from the surface defined by the function.
- The behavior for `'>'` is probably not useful, and will probably be modified in a later release.

Conic_X

Description : Value of the constant multiplying the x term in the conic equation defining the BC surface. See `Conic Equation` for the form of the equation.

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Conic_XX

Description : Value of the constant multiplying the x^2 term in the conic equation defining the BC surface. See `Conic Equation` for the form of the equation.

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Conic_XY

Description : Value of the constant multiplying the xy term in the conic equation defining the BC surface. See `Conic Equation` for the form of the equation.

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Conic_XZ

Description : Value of the constant multiplying the xz term in the conic equation defining the BC surface. See `Conic Equation` for the form of the equation.

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Conic_Y

Description : Value of the constant multiplying the y term in the conic equation defining the BC surface. See `Conic Equation` for the form of the equation.

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Conic_YY

Description : Value of the constant multiplying the y^2 term in the conic equation defining the BC surface. See `Conic Equation` for the form of the equation.

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Conic_YZ

Description : Value of the constant multiplying the yz term in the conic equation defining the BC surface. See `Conic Equation` for the form of the equation.

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Conic_Z

Description : Value of the constant multiplying the z term in the conic equation defining the BC surface. See `Conic Equation` for the form of the equation.

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Conic_ZZ

Description : Value of the constant multiplying the z^2 term in the conic equation defining the BC surface. See `Conic Equation` for the form of the equation.

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Inflow_Concentration

Description : Species concentration of the fluid flowing into the computational domain.

Physical Dimension : N/A

Type : `real`

Default : (none)

Valid values : $[0.0, 1.0]$

Inflow_Material

Description : Material number of the fluid flowing into the computational domain.

Physical Dimension : N/A

Type : `integer`

Default : If no inflow material is specified, Truchas will bring more of the existing interior fluid material into the mesh, in the proportion that fluid materials exist.

Valid values : existing material number $[1, \infty]$

Inflow_Temperature

Description : Temperature of the fluid flowing into the computational domain.

Physical Dimension : T

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Mesh_Surface

Description : One integer corresponding to a side set identifier in a Genesis (Exodus, Cubit) mesh file. This side set will be used to make a list of faces for the application of the boundary condition for the namelist. This variable is only applicable for the `Surface_Name` `'from mesh file'`.

Physical Dimension : N/A

Type : `integer`

Default : 0

Valid values : Side set numbers that exist in the mesh file.

Node_Disp_Coords

Description : List of coordinates that will be matched with nodes (cell vertices) for displacement boundary conditions. Up to 50 points can be specified as comma separated x,y,z triplets. This specification can only be used if the `Surface_Name` is `'node set'`.

Physical Dimension : L

Type : `real`

Default : (none)

Valid values : $(-\infty, \infty)$

Surface_Materials

Description :

Material number(s) of material(s) contacting the BC surface, which can be internal or external to the computational domain.

Physical Dimension : N/A

Type : `integer` (doublet to define interface between two materials)

Default : (none)

Valid values : existing material number(s) $[1, \infty]$

`Surface_Name`

Description : Name of the method to be used for specifying the BC surface location. An alternative for solid mechanics displacement BCs is to specify a set of discrete points using `'node set'`.

Physical Dimension : N/A

Type : string

Default : (none)

Valid values : `'conic'`, `'material boundary'`, `'external material boundary'`, `'from mesh file'`, `'node set'`

Note : The value `'from mesh file'` is only valid if `Mesh_File_Format` is equal to `'exodus'` or `'genesis'` and the mesh file was generated by Cubit with side sets defined.

`T0_Convection`

Description : Bulk temperature to be used for determining the convective heat transfer rate. Can be input as a list of time, temperature pairs in the same way as the `BC_Value` for `'dirichlet'` and `'radiation'`. The first value is at time 0, and up to 10 subsequent pairs of time and temperature values can be specified.

Physical Dimension : T,[t,T]

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Chapter 5

BODY Namelist

Overview

The BODY namelists define initial material distributions and conditions.

BODY Namelist Features

Required / Optional : Required

Single / Multiple Instances : Multiple

Components

- Axis
- Concentration
- File_Name
- Fill
- Height
- Length
- Material_Number

- Mesh_Material_Number
- Radius
- Rotation_Angle
- Rotation_Pt
- Rtheta_Tabular_Pt
- RZ_Tabular_Pt
- Surface_Name
- Tabular_Type
- Temperature
- Translation_Pt
- Velocity
- TG_Origin
- TG_Axis
- TG_Z_Constants
- TG_Z_Exponents
- TG_Z_Bounds
- TG_R_Constants
- TG_R_Exponents
- TG_R_Bounds

Axis

Description : The axis to be used for defining a cone, cylinder, plane, or tabular body

Physical Dimension : N/A

Type : string

Default : (none)

Valid values : `'x', 'y', 'z'`

Concentration

Description : Initial species concentration in the material body

Physical Dimension : N/A

Type : `real`

Default : (none)

Valid values : $[0.0, 1.0]$

`File_Name`

Description : Name of file containing tabular (r, z) or (r, θ) data points for defining this body

Physical Dimension : N/A

Type : string

Default : (none)

Valid values : existing file name

`Fill`

Description : The side of the surface to which material is to be inserted for this body

Physical Dimension : N/A

Type : string

Default : (none)

Valid values : `'+', '-', 'inside', 'outside'`

`Height`

Description : Height of the (cone or cylinder) body

Physical Dimension : L

Type : `real`

Default : (none)

Valid values : $(0.0, \infty)$

Length

Description : Length of each side of the box body

Physical Dimension : L

Type : `real` triplet

Default : (none)

Valid values : $(0, \infty)$

Material_Number

Description : Material number of material occupying the volume of this body

Physical Dimension : N/A

Type : `integer`

Default : (none)

Valid values : existing material number

Mesh_Material_Number

Description : Material number associated with the cells as defined in the mesh file

Physical Dimension : N/A

Type : `integer`

Default : (none)

Valid values : existing material number in mesh file (if the mesh file is in Exodus/Genesis format, this is the mesh block number)

Radius

Description : Radius of the geometric body (cone, cylinder, sphere, ellipsoid)

Physical Dimension : L

Type : `real`

Default : (none)

Valid values : $(0.0, \infty)$

`Rotation_Angle`

Description : Angle (degrees) about the (x, y, z) axes this body is to be rotated

Physical Dimension : θ

Type : `real` triplet

Default : 0.0, 0.0, 0.0

Valid values : $(-\infty, \infty)$

`Rotation_Pt`

Description : Location of the point about which this body is to be rotated

Physical Dimension : L

Type : `real` triplet

Default : 0.0, 0.0, 0.0

Valid values : $(-\infty, \infty)$

`Rtheta_Tabular_Pt`

Description : A list of (r, θ) points describing an n-sided planar polygon

Physical Dimension : (L, θ)

Type : `real` array

Default : (none)

Valid values : clockwise or counterclockwise list of at least two (r, θ) points

RZ_Tabular_Pt

Description : A list of (r, z) points describing an n-sided planar polygon

Physical Dimension : L

Type : `real` array

Default : (none)

Valid values : clockwise or counterclockwise list of at least two (r, z) points

Surface_Name

Description : Type of surface characterizing the interface topology for this body

Physical Dimension : N/A

Type : string

Default : (none)

Valid values : `'background', 'plane', 'box', 'sphere', 'ellipsoid', 'cylinder', 'cone', 'tabular', 'from mesh file'`

Tabular_Type

Description : Method by which a 3D surface will be constructed from an n-sided planar polygon

Physical Dimension : N/A

Type : string

Default : (none)

Valid values : `'rotation', 'translation'`

Temperature

Description : Initial temperature of the material body

Physical Dimension : T

Type : `real`

Default : (none)

Valid values : $(-\infty, \infty)$

`Translation_Pt`

Description : Location to which each surface origin of this body is translated

Physical Dimension : L

Type : `real` triplet

Default : 0.0, 0.0, 0.0

Valid values : $(-\infty, \infty)$

`Velocity`

Description : Initial velocity of the material body

Physical Dimension : L/t

Type : `real` triplet

Default : 0.0, 0.0, 0.0

Valid values : $(-\infty, \infty)$

`TG_Origin`

Description : Coordinate origin for the body temperature gradient

Physical Dimension : length

Type : `real` triplet

Default : 0.0, 0.0, 0.0

Valid values : $(-\infty, \infty)$

TG_Axis

Description : Axis (direction) for the body temperature gradient. Note: The code normalized the vector. The only invalid value is a vector of zero length (0,0,0).

Physical Dimension : unit vector (internally normalized)

Type : `real` triplet

Default : 0.0, 0.0, 1.0 (the simulation Z axis)

Valid values : any finite vector

TG_Z_Constants

Description : Constants used for the axial gradient polynomial

Physical Dimension : n/a

Type : `real` triplet

Default : 0.0, 0.0, 0.0

Valid values : $(-\infty, \infty)$

TG_Z_Exponents

Description : Exponents used for the axial gradient polynomial

Physical Dimension : n/a

Type : `integer` triplet

Default : 1, 2, 3

Valid values : $(0, \infty)$

TG_Z_Bounds

Description : Limits for application for axial variation of the body temperature gradient. Measured from the gradient origin along the gradient axis. Beyond the bounds, temperature is constant along the axis at the value determined by the gradient parameters at the bounds.

Physical Dimension : length

Type : `real` doublet

Default : $-\infty, \infty$

Valid values : $(-\infty, \infty)$

TG_R_Constants

Description : Constants used for the radial gradient polynomial

Physical Dimension : n/a

Type : `real` triplet

Default : 0.0, 0.0, 0.0

Valid values : $(-\infty, \infty)$

TG_R_Exponents

Description : Exponents used for the radial gradient polynomial

Physical Dimension : n/a

Type : `integer` triplet

Default : 1, 2, 3

Valid values : $(0, \infty)$

TG_R_Bounds

Description : Limits for application for temperature variation perpendicular to (radially from) the gradient axis. Measured perpendicularly from the gradient axis. Beyond the bounds, temperature is constant along the axis at the value determined by the gradient parameters at the bounds.

Physical Dimension : length

Type : `real` doublet

Default : 0.0, ∞

Valid values : $(0.0, \infty)$

Chapter 6

CHEMICAL_REACTION Namelist

Overview

The CHEMICAL_REACTION namelist identifies the different chemical reactions that are present during the simulation. Currently only one model is available for chemical reactions, but additional models are under development. The current model describes a self-catalytic reaction through the following equation:

$$\begin{aligned} \frac{\partial}{\partial t}C(t) &= (k_1 + k_2 * C(t)^m)(C_{max} - C(t))^n \\ k_1 &= k_1^o * e^{-E_1^a/RT} \\ k_2 &= k_2^o * e^{-E_2^a/RT} \end{aligned} \tag{6.1}$$

C is concentration of the product, t is time, k_1^o and k_2^o the reaction constants for the two stages and are both a function of temperature, T. The order of the reaction is given by $n + m$. R is the gas constant. It is further assumed that the heat of reaction, H_{tot} is evenly distributed as the reaction progresses. i.e.

$$\frac{\partial H}{\partial t} = H_{tot}\frac{\partial C}{\partial t} \tag{6.2}$$

NOTE: R is currently assumed to be 8.314 J/mol^oK which limits us to using MKS units. Consequently the temperature must be the absolute temperature (K), the activation energies, E_1^a and E_2^a, should be in consistent units (J), and the the time and rates k_1, k_2, should be consistent.

CHEMICAL_REACTION Namelist Features

Required / Optional : Optional

Single / Multiple Instances : Multiple

Components

- CR_Active
- CR_m
- CR_n
- CR_Cmax
- CR_ko1
- CR_ko2
- CR_Ea1
- CR_Ea2
- CR_Htot
- CR_Reactants_id
- CR_Products_id

CR_Active

Description : Activate or deactivate the current reaction

Physical Dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.` or `.false.`

CR_m

CR_n

Description : Order of the reaction coefficients. The total order of the reaction is $n + m$.

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

CR_Cmax

Description : The maximum concentration of the product(s) allowed by the model

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $[0.0, 1.0]$

CR_ko1

CR_ko2

Description : Pre-exponential factors

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

CR_Ea1

CR_Ea2

Description : Activation Energies

Physical Dimension : E

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

CR_Htot

Description : The heat of reaction. This is assumed to be released linearly with the progress of the reaction.

Physical Dimension : E

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

CR_Reactants_id

Description : An integer array of reactants. Currently only one reactant is allowed.

Physical Dimension : N/A

Type : `integer`

Default : 0

Valid values : Number should correspond to a `Material_Number` specified for a material in a `MATERIAL` namelist

CR_Products_id

Description : An integer array of products. Currently only one product is allowed.

Physical Dimension : N/A

Type : `integer`

Default : 0

Valid values : Number should correspond to a `Material_Number` specified for a material in a `MATERIAL` namelist

Chapter 7

DIFFUSION_SOLVER Namelist

Overview

The DIFFUSION_SOLVER namelist sets the parameters that are specific to the diffusion solver. The namelist is read only when the PHYSICS namelist variable Diffusion_Solver is assigned the value true. The diffusion solver uses a variable step-size BDF2 method to integrate one of several possible different coupled diffusion systems. An accelerated inexact Newton (AIN) method is used to solve the nonlinear system required for each BDF2 step, and a relaxation process, either SSOR or algebraic multigrid, with an approximate system Jacobian is used to precondition the nonlinear system. The BDF2 integrator manages the local truncation error of each step by suggesting a new step size to use on the next step in order to keep an estimate of the error within a user-specified range, and by rejecting those steps where the estimate is too large. A failed step may be retried with successively smaller step sizes.

The initial step size and upper and lower bounds for the step size are set in the NUMERICS namelist. If the diffusion solver is the only physics enabled (currently the only option), then the NUMERICS variables Dt_Grow and Dt_Constant are ignored and the step size determined solely by the diffusion solver. Otherwise, the step size suggested by the diffusion solver may be limited by the other physics or by Dt_Constant or Dt_Grow.

DIFFUSION_SOLVER Namelist Features

Required / Optional Optional

Single / Multiple Instances Single

Components

- Abs_Conc_Tol
- Abs_Enthalpy_Tol
- Abs_Temp_Tol
- AIN_Preconditioner
- AIN_Tol
- AIN_Vec_Tol
- Max_AIN_Itr
- Max_AIN_Vec
- Max_Step_Tries
- PC_AMG_Cycles
- PC_SSOR_Relax
- PC_SSOR_Sweeps
- Rel_Conc_Tol
- Rel_Enthalpy_Tol
- Rel_Temp_Tol
- System_Type
- Verbose_Stepping

Abs_Conc_Tol

Description: The tolerance ϵ for the absolute error component of the concentration error norm used by the BDF2 integrator. If $\delta\mathbf{c}$ is a concentration field increment with reference concentration field $\mathbf{c}$, then this error norm is

$$|||\delta\mathbf{c}||| \equiv \max_j |\delta c_j|/(\epsilon + \eta|c_j|).$$

The relative error tolerance η is given by Rel_Conc_Tol. This variable is only relevant to diffusion systems that include concentration as a dependent variable.

Physical dimension: same as the 'concentration' variable

Type: real

Default: none

Valid values: ≥ 0

Notes: The error norm is dimensionless and normalized. The BDF2 integrator will accept time steps where the estimated truncation error is less than 2, and chooses the next suggested time step so that its prediction of the next truncation error is $\frac{1}{2}$.

For c_j sufficiently small the norm approximates an absolute norm with tolerance ϵ, and for c_j sufficiently large the norm approximates a relative norm with tolerance η. If $\epsilon = 0$ then the norm is a pure relative norm and the concentration must be bounded away from 0.

Abs_Enthalpy_Tol

Description: The tolerance ϵ for the absolute error component of the enthalpy error norm used by the BDF2 integrator. If $\delta\mathbf{H}$ is a enthalpy field increment with reference enthalpy field $\mathbf{H}$, then this error norm is

$$|||\delta\mathbf{H}||| \equiv \max_j |\delta H_j|/(\epsilon + \eta|H_j|).$$

The relative error tolerance η is given by `Rel_Enthalpy_Tol`. This variable is only relevant to diffusion systems that include enthalpy as a dependent variable.

Physical dimension: E/(T*L^3)

Type: `real`

Default: none

Valid values: ≥ 0

Notes: The error norm is dimensionless and normalized. The BDF2 integrator will accept time steps where the estimated truncation error is less than 2, and chooses the next suggested time step so that its prediction of the next truncation error is $\frac{1}{2}$.

For H_j sufficiently small the norm approximates an absolute norm with tolerance ϵ, and for H_j sufficiently large the norm approximates a relative norm with tolerance η. If $\epsilon = 0$ then the norm is a pure relative norm and the enthalpy must be bounded away from 0.

Abs_Temp_Tol

Description: The tolerance ϵ for the absolute error component of the temperature error norm used by the BDF2 integrator. If $\delta\mathbf{T}$ is a temperature field increment with reference temperature field $\mathbf{T}$, then this error norm is

$$|||\delta\mathbf{T}||| \equiv \max_j |\delta T_j|/(\epsilon + \eta|T_j|).$$

The relative error tolerance η is given by `Rel_Temp_Tol`. This variable is only relevant to diffusion systems that include temperature as a dependent variable.

Physical dimension: T

Type: `real`

Default: none

Valid values: ≥ 0

Notes: The error norm is dimensionless and normalized. The BDF2 integrator will accept time steps where the estimated truncation error is less than 2, and chooses the next suggested time step so that its prediction of the next truncation error is $\frac{1}{2}$.

For c_j sufficiently small the norm approximates an absolute norm with tolerance ϵ, and for c_j sufficiently large the norm approximates a relative norm with tolerance η. If $\epsilon = 0$ then the norm is a pure relative norm and the temperature must be bounded away from 0.

AIN_Preconditioner

Description: The choice of preconditioner for the AIN iteration. There are currently two preconditioners to choose from: SSOR and HYPRE_AMG. The former is symmetric over relaxation, and the later is an algebraic multigrid preconditioner from the *hypre* library.

Type: `string`

Default: `'SSOR'`

Valid values: `'SSOR'` or `'HYPRE_AMG'`

Notes: If SSOR is the chosen as the preconditioner, the user can set `PC_SSOR_Relax` to the over relaxation parameter and `PC_SSOR_Sweeps` to the number of SSOR sweeps. If HYPRE_AMG is the chosen as the preconditioner, the user can set `PC_AMG_Cycles` to the number of AMG cycles per preconditioning step.

AIN_Tol

Description: The convergence tolerance for the AIN nonlinear iteration. The nonlinear system is considered solved by the current iterate if the BDF2 integrator norm of the last solution correction is less than this value.

Type: `real`

Default: 0.1

Valid values: $(0, 1)$

Notes: This tolerance is relative to the dimensionless and normalized BDF2 integrator norm; see `Abs_Conc_Tol`, for example. The nonlinear system only needs to be solved to an accuracy equal to the acceptable local truncation error for the step, which is roughly 1. Solving to a greater accuracy is wasted effort. Using a tolerance in the range $(0.01, 0.1)$ is generally adequate to ensure a sufficently converged nonlinear iterate.

AIN_Vec_Tol

Description: The AIN accelerator vector drop tolerance. When assembling the acceleration subspace vector by vector, a vector is dropped when the sine of the angle between the vector and the subspace less than this value.

Type: `real`

Default: 0.001

Valid values: > 0

Max_AIN_Itr

Description: The maximum number of AIN nonlinear iterations allowed.

Type: `integer`

Default: 5

Valid values: ≥ 2

Notes: The failure of an AIN iteration to converge is *not* necessarily fatal; the BDF2 integration procedure expects that this will occur, using it as an indication that the preconditioner for the nonlinear system needs to be updated. If still unsuccessful, the step will be retried with a halved time step size, perhaps repeatedly. Therefore it is important that the maximum number of iterations not be set too high, as this merely delays the recognition that some recovery strategy needs to be taken, and can result in much wasted effort.

Max_AIN_Vec

Description: The maximum number of vectors to use in the AIN acceleration procedure.

Type: `integer`

Default: `Max_AIN_Itr` -1

Valid values: > 0

Notes: The acceleration vectors are derived from the difference of successive nonlinear function iterates accumulated over the course of a nonlinear solve. Thus the maximum possible number of acceleration vectors available is one less than the maximum number of AIN iterations, and so specifying a larger number merely wastes memory. If a large number of AIN iterations is being used (probably not advised in itself) then it may be appropriate to use a smaller value for this parameter, otherwise the default value is fine.

Max_Step_Tries

Description: The maximum number of attempts to successfully take a time step before giving up. The step size is reduced between each try.

Type: `integer`

Default: 10

Valid values: ≥ 1

Notes: If other physics is enabled (not currently an option) then this variable is effectively assigned the value 1, overriding the input value. This is required for compatibility with the other physics solvers which currently have no way of recovering from a failed step.

PC_AMG_Cycles

Description: The number of V-cycles to take per preconditioning step of the nonlinear iteration.

Physical Dimension: dimensionless

Type: `integer`

Default: 1

Valid values: ≥ 1

Notes: We use standard $V(1,1)$ cycles. Parameters other than the number of V cycles cannot be controlled by the user.

PC_SSOR_Relax

Description: The relaxation parameter used in the SSOR preconditioning of the nonlinear system.

Physical Dimension: dimensionless

Type: `real`

Default: 1.4

Valid values: $(0, 2)$

Notes: A value less than 1 gives under-relaxation and a value greater than 1 over-relaxation.

PC_SSOR_Sweeps

Description: The number of sweeps used in the SSOR preconditioning of the nonlinear system.

Type: `integer`

Default: 4

Valid values: ≥ 1

Notes: The effectiveness of the SSOR preconditioner (measured by the convergence rate of the nonlinear iteration) improves as the number of sweeps increases, though at increasing cost. For especially large systems where the effectiveness of SSOR deteriorates, a somewhat larger value than the default 4 sweeps may be required. Using fewer than 4 sweeps is generally not recommended.

Rel_Conc_Tol

Description: The tolerance η for the relative error component of the concentration error norm used by the BDF2 integrator. If $\delta\mathbf{c}$ is a concentration field increment with reference concentration field $\mathbf{c}$, then this error norm is

$$|||\delta\mathbf{c}||| \equiv \max_j |\delta c_j|/(\epsilon + \eta|c_j|).$$

The absolute error tolerance ϵ is given by `Abs_Conc_Tol`. This variable is only relevant to diffusion systems that include concentration as a dependent variable.

Physical Dimension: dimensionless

Type: `real`

Default: 0.0

Valid values: $[0, 1)$

Notes: See the notes for `Abs_Conc_Tol`.

Rel_Enthalpy_Tol

Description: The tolerance η for the relative error component of the enthalpy error norm used by the BDF2 integrator. If $\delta\mathbf{c}$ is a enthalpy field increment with reference enthalpy field $\mathbf{H}$, then this error norm is

$$|||\delta\mathbf{H}||| \equiv \max_j |\delta H_j|/(\epsilon + \eta|H_j|).$$

The absolute error tolerance ϵ is given by `Abs_Enthalpy_Tol`. This variable is only relevant to diffusion systems that include enthalpy as a dependent variable.

Physical Dimension: dimensionless

Type: `real`

Default: 0.0

Valid values: $[0, 1)$

Notes: See the notes for `Abs_Enthalpy_Tol`.

Rel_Temp_Tol

Description: The tolerance η for the relative error component of the temperature error norm used by the BDF2 integrator. If $\delta\mathbf{T}$ is a temperature field increment with reference temperature field $\mathbf{T}$, then this error norm is

$$|||\delta\mathbf{T}||| \equiv \max_j |\delta T_j|/(\epsilon + \eta|T_j|).$$

The absolute error tolerance ϵ is given by `Abs_Temp_Tol`. This variable is only relevant to diffusion systems that include temperature as a dependent variable.

Physical Dimension: dimensionless

Type: `real`

Default: 0.0

Valid values: $[0, 1)$

Notes: See the notes for `Abs_Temp_Tol`.

System_Type

Description: A flag that specifies the type of diffusion system to be solved.

Type: string

Default: none

Valid values: The following choices of diffusion system are available:

`"species"` — Diffusion of a scalar field c (species concentration, e.g.),

$$\frac{\partial c}{\partial t} = \nabla \cdot D(c) \nabla c.$$

The diffusivity $D(c)$ is defined through the `SPC_Diffusivity_*` variables of the `MATERIAL` namelists. The boundary conditions on the scalar field c are defined through the `DS_SPECIES_BC` namelists, and the initial conditions are defined through the `Concentration` variable of the `BODY` namelists.

`"thermal"` — Simple heat conduction

$$\frac{\partial H}{\partial t} = \nabla \cdot K(T) \nabla T + q$$

with dependent variables temperature T and enthalpy density H. The enthalpy density is algebraically related to temperature as $H = f(T)$ where $\frac{df}{dT} = \rho\, c_p(T)$ is the volumetric heat capacity. The conductivity $K(T)$, density ρ and specific heat $c_p(T)$ are defined through the `Conductivity_*`, `Density`, and `Cp_*`, variables of the `MATERIAL` namelists. The optional volumetric heat source function q is specified by the `HEAT_SOURCE` namelist. The boundary conditions on temperature T are defined through the `DS_TEMPERATURE_BC` namelists, and the initial temperature conditions are defined through the `Temperature` variable of the `BODY` namelists.

`"thermal+species"` — Coupled heat conduction and scalar diffusion

$$\frac{\partial H}{\partial t} = \nabla \cdot K(T,c) \nabla T + q, \quad \frac{\partial c}{\partial t} = \nabla \cdot D(T,c) \nabla c.$$

This is merely a coupled combination of the `"thermal"` and `"species"` systems where the conductivity K, diffusivity D and specific heat c_p are possibly functions of both temperature T and concentration c.

`Verbose_Stepping`

Description: A flag that enables the output of detailed BDF2 time stepping information. The human-readable information is written to a file with the suffix `.bdf2.out` in the output directory.

Type: `logical`

Default: `.false.`

Chapter 8

DS_SPECIES_BC Namelist

Overview

The variables of a DS_SPECIES_BC namelist define the boundary condition imposed on the concentration variable along a portion of the computational boundary. These conditions pertain only to the diffusion solver and these namelists are only read when the diffusion solver is enabled in the PHYSICS namelist and the diffusion system includes concentration as a dependent variable.

DS_SPECIES_BC Namelist Features

Required/Optional: Required for species diffusion problems

Single/Multiple Instances: Multiple

Components

- BC_Type
- BC_Value
- Face_Set_IDs
- Name

BC_Type

Description: Specifies the type of boundary condition imposed on the concentration variable c.

Type: string

Default: none

Valid values: The following types of conditions are implemented, where Γ denotes a portion of the computational boundary with outward unit normal $\hat{\mathbf{n}}$, and g the boundary data.

"dirichlet" — specifies a Dirichlet condition $c = g$ on Γ.

"neumann" — specifies a flux condition $-D\nabla c \cdot \hat{\mathbf{n}} = g$ on Γ.

"hneumann" — specifies a no-flux condition $-D\nabla c \cdot \hat{\mathbf{n}} = 0$ on Γ.

Notes: The part of the boundary Γ where the condition is applied is specified by `Face_Set_IDs` and the boundary data g by `BC_Value`.

A concentration solution requires that unique concentration boundary conditions be specified for all cell faces on the exterior mesh boundary. Because there is no default boundary condition, the user must ensure that each exterior mesh face is associated with a unique `DS_SPECIES_BC` namelist.

BC_Value

Description: The constant value of the boundary data g for this BC definition. The meaning of the value depends on the type of boundary condition specified by `BC_Type`.

Physical Dimension: varies

Type: `real`

Default: none

Valid values: $(-\infty, \infty)$

Face_Set_IDs

Description: A list of up to 32 face set IDs that define the subset of boundary faces where the boundary condition is applied.

Type: `integer`

Default: none

Valid values: any defined face set ID

Note: The side sets defined within a Genesis/Exodus mesh file are interpreted by TRUCHAS as face sets having the same IDs. Currently this is the only way to define a face set.

Name

Description: A unique name used to identify a particular instance of this namelist.

Type: an arbitrary, case-insensitive string of no more than 32 characters.

Default: none

Chapter 9

DS_TEMPERATURE_BC Namelist

Overview

The variables of a DS_TEMPERATURE_BC namelist define the boundary condition imposed on the temperature variable along a portion of the computational boundary. These conditions pertain *only* to the diffusion solver and these namelists are only read when the diffusion solver is enabled in the PHYSICS namelist and the diffusion system includes temperature as a dependent variable.

DS_TEMPERATURE_BC Namelist Features

Required/Optional: Required for diffusion systems that involve temperature.

Single/Multiple Instances: Multiple

Components

- BC_Type
- BC_Value
- Face_Set_IDs
- Name

BC_Type

Description: Specifies the type of boundary condition imposed on the temperature variable T.

Type: string

Default: none

Valid values: The following types of conditions are implemented, where Γ denotes a portion of the computational boundary with outward unit normal $\hat{\mathbf{n}}$, and g the boundary data.

"dirichlet" — specifies a Dirichlet condition $T = g$ on Γ.

"neumann" — specifies a flux condition $-K\nabla T \cdot \hat{\mathbf{n}} = g$ on Γ.

"hneumann" — specifies a no-flux condition $-K\nabla T \cdot \hat{\mathbf{n}} = 0$ on Γ.

Notes: The part of the boundary Γ where the condition is applied is specified by `Face_Set_IDs` and the boundary data g by `BC_Value`.

A temperature solution requires that unique temperature boundary conditions be specified for all cell faces on the exterior mesh boundary. Because there is no default boundary condition, the user must ensure that each exterior mesh face is associated with a unique `DS_TEMPERATURE_BC` namelist.

BC_Value

Description: The constant value of the boundary data g for this BC definition. The meaning of the value depends on the type of boundary condition specified by `BC_Type`.

Physical Dimension: varies

Type: `real`

Default: none

Valid values: $(-\infty, \infty)$

Face_Set_IDs

Description: A list of up to 32 face set IDs that define the subset of boundary faces where the boundary condition is applied.

Type: `integer`

Default: none

Valid values: any defined face set ID

Note: The side sets defined within a Genesis/Exodus mesh file are interpreted by TRUCHAS as face sets having the same IDs. Currently this is the only way to define a face set.

Name

Description: A unique name used to identify a particular instance of this namelist.

Type: an arbitrary, case-insensitive string of no more than 32 characters.

Default: none

Chapter 10

ELECTROMAGNETICS Namelist

Overview

The ELECTROMAGNETICS namelist sets most of the parameters used by the electromagnetic (EM) solver to calculate the Joule heat used in induction heating simulations. Exceptions are the material electric permittivity, magnetic permeability, and electric conductivity, which are specified in the MATERIAL namelist, and the induction coils that produce an external magnetic source field, which are specified in INDUCTION_COIL namelists. The EM calculations are performed on a tetrahedral mesh specified by the ALTMESH namelist, which is generally different than the main mesh used throughout the rest of TRUCHAS.

A Remark on Units: The EM solver assumes SI units by default. In particular, the result of the Joule heat calculation is a *power density*—W/m^3 in SI units. To use a different system of units, the user must supply appropriate values for the free-space parameters Epsilon_0 and Mu_0. In any case, the user must ensure that a consistent set of units is used throughout TRUCHAS.

ELECTROMAGNETICS Namelist Features

Required / Optional : Optional

Single / Multiple Instances : Single

Components

- CG_Stopping_Tolerance
- EM_Domain_Type
- Epsilon_0
- Graphics_Output
- Material_Change_Threshold
- Maximum_CG_Iterations
- Maximum_Source_Cycles
- Mu_0
- Num_Etasq
- Output_Level
- Source_Frequency
- Source_Times
- SS_Stopping_Tolerance
- Steps_Per_Cycle
- Symmetry_Axis
- Uniform_Source

CG_Stopping_Tolerance

Description : Tolerance used to determine when the conjugate gradient (CG) iteration has converged. The criterion used is that $\|\mathbf{r}\|/\|\mathbf{r}_0\| <$ CG_Stopping_Tolerance. The electromagnetics solver uses its own special preconditioned CG linear solver.

Physical dimension : N/A

Type : real

Default : 10^{-5}

Valid values : $(0, 0.1)$

Notes : The numerical characteristics of the electromagnetic system require that the linear systems be solved to significantly greater accuracy than would otherwise be required. Too loose a tolerance will manifest itself in a significant build-up of noise in the solution of the electric field over the course of the simulation. This input variable should not be greater than 10^{-4}.

EM_Domain_Type

Description : A flag specifying the type of domain geometry that is discretized by the computational mesh.

Physical dimension : N/A

Type : string

Default : none

Valid values : `'Full_Cylinder'`, `'Half_Cylinder'`, `'Quarter_Cylinder'`

Notes : At this time there is not yet a facility for specifying general boundary conditions for the electromagnetic simulation. Consequently, the computational domain Ω is limited to the following special cases when `Symmetry_Axis='z'`:

$$\begin{array}{ll} \texttt{'Full_Cylinder'}: & \Omega = \{(x,y,z) \mid x^2+y^2 \le r^2,\ z_1 \le z \le z_2\} \\ \texttt{'Half_Cylinder'}: & \Omega = \{(x,y,z) \mid x^2+y^2 \le r^2,\ x \ge 0,\ z_1 \le z \le z_2\} \\ \texttt{'Quarter_Cylinder'}: & \Omega = \{(x,y,z) \mid x^2+y^2 \le r^2,\ x,y \ge 0,\ z_1 \le z \le z_2\} \end{array}$$

The values $r > 0$, $z_1 < z_2$ are inferred from the mesh and are not specified directly. Dirichlet source field conditions are imposed on the boundaries $\{x^2 + y^2 = r^2\}$ and $\{z = z_1, z_2\}$, and symmetry conditions on the symmetry planes $\{x = 0\}$ and $\{y = 0\}$ if present. See the User Manual for more details.

The analogous definitions for the other possible symmetry axes, `'x'` and `'y'`, are obtained by the appropriate cyclic permutation of the coordinates.

For the computational mesh used in the EM simulation, see the `ALTMESH` namelist.

Epsilon_0

Description : Free-space electric permittivity

Physical dimension : $\mathrm{L}^{-3}\ \mathrm{M}^{-1}\ \mathrm{t}^{4}\ \mathrm{I}^{2}$

Type : `real`

Default : 8.854188×10^{-12} farad/m (SI)

Valid values : $(0.0, \infty)$

Notes : This input variable is provided in order to allow the use of units other than the default SI units; doing so is *strongly* discouraged, however.

`Graphics_Output`

Description : Controls the graphics output of the electromagnetic solver.

Physical dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.` or `.false.`

Notes : If `.true.`, the electromagnetic solver will generate its own graphics data files in OpenDX format (see `http://www.opendx.org`). The files contain the material parameter fields, the averaged Joule heat field, and the time series of the electromagnetic fields. The files are identified by the suffixes `-EM.dx` and `-EM.bin`. The value of `Graphics_Output` has no impact on the normal graphics output generated by TRUCHAS which is determined elsewhere, and the averaged Joule heat field used in heat transport will be output there in either case.

`Material_Change_Threshold`

Description : Controls, at each step, whether the Joule heat is recalculated in response to temperature-induced changes in the EM material parameter values. The Joule heat is recalculated whenever the difference between the current parameter values and those when the Joule heat was last computed exceeds this threshold value. Otherwise the previously calculated Joule heat is used. The maximum relative change is used as the difference measure.

Physical dimension : N/A

Type : `real`

Default : 0.3

Valid values : $(0.0, \infty)$

Notes : The electric conductivity and magnetic permeability are the only values whose changes are monitored. The electric permittivity only enters the equations through the displacement current term, which is exceedingly small in this quasi-magnetostatic regime and could be dropped entirely. Thus the Joule heat is essentially independent of the permittivity and so any changes in its value are ignored.

For electric conductivity, only the conducting region (where the value is positive) is considered when computing the difference. An underlying assumption is that this region remains fixed throughout the simulation.

Maximum_CG_Iterations

Description : Maximum number of conjugate gradient (CG) iterations allowed. The electromagnetics solver uses its own special preconditioned CG linear solver.

Physical dimension : N/A

Type : `integer`

Default : 500

Valid values : $(0.0, \infty)$

Maximum_Source_Cycles

Description : The electromagnetic field equations are integrated in time toward a periodic steady state. This input variable specifies the time limit, measured in cycles of the sinusoidal source field, for the Joule heat calculation.

Physical dimension : N/A

Type : `integer`

Default : 10

Valid values : $(0.0, \infty)$

Notes : To avoid ringing, the amplitude of the external source field is ramped and is not at full strength until after approximately two cycles have passed. Consequently this input variable should not normally be < 3. Convergence to a periodic steady state is usually attained within 5 cycles; see `SS_Stopping_Tolerance`. If convergence is not attained within the limit allowed by this input variable, the last result is returned and a warning issued, but execution continues with the rest of the physics simulation.

The electromagnetic field equations are solved on an *inner* time distinct from that of the rest of the physics; see `SS_Stopping_Tolerance`.

Mu_0

Description : Free-space magnetic permeability

Physical dimension : $\mathrm{L\ M\ t^{-2}\ I^{-2}}$

Type : `real`

Default : $4\pi \times 10^{-7}$ henry/m (SI)

Valid values : $(0.0, \infty)$

Notes : This input variable is provided in order to allow the use of units other than the default SI units; doing so is *strongly* discouraged, however.

Num_Etasq

Description : This value is used for the displacement current coefficient η^2, in the low-frequency, nondimensional scaling of Maxwell's equations, when its value exceeds the physical value.

Physical dimension : dimensionless

Type : `real`

Default : 0

Valid values : $(0.0, \infty)$

Notes : The quasi-magnetostatic regime is characterized by $\eta^2 \ll 1$. Since this value can become exceedingly small (resulting in a difficult-to-solve, ill-conditioned system), it may be helpful to use a numerical value instead, say $\eta^2 = 10^{-6}$ or 10^{-8}, without having any discernable effect on the solution. However, it is generally safe to ignore this variable and let the solver use the physical value. See the *Truchas Physics and Algorithms* for more details.

Output_Level

Description : Controls the verbosity of the electromagnetic solver

Physical dimension : N/A

Type : `integer`

Default : 1

Valid values : 1, 2, 3 or 4

Notes : At the default level, 1, a status message is output at the end of each source field cycle showing the progress toward steady state. Level 2 adds a summary of the CG iteration for each time step. Level 3 adds the norm of the difference between the solution and extrapolated predictor for each time step. This gives an indication of the (time) truncation error, and if noise is accumulating in the system it will be seen here; see `CG_Stopping_Tolerance`. Level 4 adds convergence info for each CG iterate. Levels 1 and 2 are typical.

Source_Frequency

Description : Frequency f (cycles per unit time) of the sinusoidally-varying magnetic source fields that drive the Joule heat calculation.

Physical dimension : t^{-1}

Type : `real`

Default : none

Valid values : Any single positive value, or any sequence of positive values.

Notes : A sequence of up to 32 values may be assigned to this variable in order to specify a time-dependent frequency; see `Source_Times` and Fig. 10.1.

The source fields are due to induction coils specified through `INDUCTION_COIL` namelists, and a spatially uniform source field specified by `Uniform_Source`. All operate at the common frequency specified by this variable, and a common phase. The phase value is irrelevant due to the time averaging of the calculated Joule heat.

The Joule heat calculation is coupled to the rest of the physics in a manner that assumes the time scale of the electromagnetic fields, f^{-1} is *much smaller* than the time scale of the other physics (as defined by the characteristic time step). Consequently, the frequency f must not be too small.

Source_Times

Description : A sequence of times that define the time partition of the piecewise-constant functional form used in the case of time-dependent source field parameters.

Physical dimension : t

Type : `real`

Default : none

Valid values : Any strictly increasing sequence of values.

Notes : If this variable is not specified, then the source field parameters are assumed to be constant in time, with a *single* value assigned to `Source_Frequency`, `Uniform_Source`, and each `Current` variable in any `INDUCTION_COIL` namelists. Otherwise, if n values are assigned to `Source_Times`, then $n + 1$ values must be assigned to each of those variables. See Fig. 10.1 for a description of the functional form described by these values.

At most 31 values may be specified for this variable.

SS_Stopping_Tolerance

Description : The electromagnetic field equations are integrated in time toward a periodic steady state. Convergence to this steady state is measured by comparing the computed Joule heat field averaged over the last source field cycle, q_{last}, with the result from the previous cycle, q_{prev}. When $\|q_{\text{last}} - q_{\text{prev}}\|_{\max}/\|q_{\text{last}}\|_{\max} <$ `SS_Stopping_Tolerance`, the Joule heat calculation is considered converged and q_{last} returned.

Physical dimension : N/A

Type : `real`

Default : 10^{-2}

Valid values : $(0.0, \infty)$

Notes : Depending on the accuracy of the other physics, 10^{-2} or 10^{-3} are adequate values. If the value is taken too small (approaching machine epsilon) convergence cannot be attained.

It is assumed that the time scale of the electromagnetic fields is *much shorter* than the time scale of the other physics; see `Source_Frequency`. In this case it suffices to solve the electromagnetic field equations to a periodic steady state, while temporarily freezing the other physics, and averaging the rapid temporal variation in the Joule heat field over a cycle. In effect, the electromagnetic field equations are solved over an *inner* time distinct from that of the rest of the physics.

Steps_Per_Cycle

Description : The number of time steps per cycle of the external source field used to integrate the electromagnetic field equations

Physical dimension : N/A

Type : `integer`

Default : 20

Valid values : $(0.0, \infty)$

Notes : Increasing the number of time steps per cycle increases the accuracy of the Joule heat calculation, while generally increasing the execution time. A reasonable range of values is $[10, 40]$; anything less than 10 is *severely* discouraged.

The electromagnetic field equations are solved on an *inner* time distinct from that of the rest of the physics; see `SS_Stopping_Tolerance`.

Symmetry_Axis

Description : A flag that specifies which axis is to be used as the problem symmetry axis for the Joule heat simulation.

Physical dimension : N/A

Type : string

Default : `'z'`

Valid values : { `'x'`, `'y'`, `'z'` }

Notes : The value of this variable determines the orientation of the uniform magnetic source field specified by `Uniform_Source`, and the coils specified by the `INDUCTION_COIL` namelists, if any. It also determines the assumed orientation of the computational domain; see `EM_Domain_Type`.

Uniform_Source

Description : Amplitude of a sinusoidally-varying, uniform magnetic source field that drives the Joule heat computation. The field is directed along the problem symmetry axis as specified by `Symmetry_Axis`.

Physical dimension : $\mathrm{I\,L^{-1}}$

Type : `real`

Default : 0

Valid values : Any single value, or any sequence of values.

Notes : A sequence of up to 32 values may be assigned to this variable in order to specify a time-dependent amplitude; see `Source_Times` and Fig. 10.1. If this variable is not specified, its value or values, as appropriate, are assumed to be zero.

The total external magnetic source field that drives the Joule heat computation will be the superposition of this field and the fields due to the coils specified in the `INDUCTION_COIL` namelists, if any.

For reference, the magnitude of the magnetic field within an infinitely-long, finely-wound coil with current *density* I is simply I. The field magnitude at the center of a circular current loop of radius r with current I is $I/2r$. In both cases the field is directed along the axis of the coil/loop.

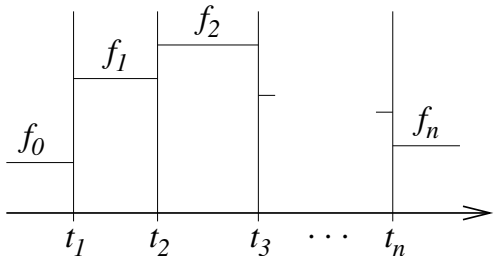

Figure 10.1: Piecewise constant function form showing the constant values $f_0, f_1, \ldots, f_n$ in relation to the time partition $t_1, t_2, \ldots, t_n$.

Chapter 11

HEAT_SOURCE Namelist

Overview

The HEAT_SOURCE namelist sets parameters that define can be used to define two types of heat sources. There can only be one instance of this namelist.

- **Moving Gaussian:**
  The user can define the strength, location, size and motion of a heat source. This heat source is applied over an ellipsoid shaped region of the domain and weighted by a Guassian distribution centered at the center of the ellipsoid. The code currently supports only one such heat source in the domain. To define such a moving Gaussian, the user must set heat_source_type to 'moving gaussian' and define the moving Gaussian heat source by setting the namelist parameters heat_source_center, heat_source_constant, heat_source_radius, heat_source_time, and travel_speed.

- **Piecewise Constant:**
  Alternatively, the user can specify a piecewise constant heat source over the domain. To define such a heat source, the user must set heat_source_type to 'piecewise constant, and specify the heat source by setting the namelist parameters volumetric_heat_source and, optionally, cell_set_id_list. The user can specify three different types of piecewise constant heat source.

  1. *Constant over the entire domain:*
     Set volumetric_heat_source to the constant and do not set cell_set_id_list.
  2. *The same constant over one or more cell sets, and zero elsewhere:*
     Set volumetric_heat_source to the constant and set cell_set_id_list to a list of one or more cell set IDs.

3. *A different constant for each cell set and zero elsewhere:*
   Set `volumetric_heat_source` to a list of constants, and set `cell_set_id_list` to a list of the same length of cell set IDs of the corresponding cell sets.

Note that cases 2. and 3. are only possible with an Exodus mesh file that contains the appropriate cell blocks/IDs. Case 1. can be used with any mesh.

Note that it is not allowed to specify both types of heat source ('`moving gaussian`' and '`piecewise constant`') at the same time. As a result the code will stop with an error message, if `heat_source_type` is set to '`moving gaussian`' and any of the namelist parameters that are used to define a piecewise constant heat source appear in the namelist. Similarly, the code will stop with an error message, if `heat_source_type` is set to '`piecewise constant`' and any of the namelist parameters that are used to define a moving Gaussian appear in the namelist.

`HEAT_SOURCE` Namelist Features

Required / Optional : Optional

Single / Multiple Instances : Single

Limitations : A truly '`piecewise contant`' volumetric heat source can only be specified on an Exodus mesh.

Components

- `cell_set_id_list`
- `heat_source_type`
- `heat_source_center`
- `heat_source_constant`
- `heat_source_radius`
- `heat_source_time`
- `travel_speed`
- `volumetric_heat_source`

cell_set_id_list

Description : Specifies a list of cell set IDs that correspond to cell sets that are specified in an Exodus mesh file. These cell sets are used to specify a piecewise constant volumetric heat source. (See `volumetric_heat_source` for more details.)

Pysical Dimension : N/A

Type : `integer`

Default : none

Valid Values : The user can input a list of any integers. However, all the integers should conicide with a cell set ID, that is specified in the Exodus mesh. If the list of cell set IDs contains an ID that does not exist in the Exodus mesh file, then this cell set will be treated like the empty set. The code will not generate an error, and there will be no cell in the mesh that is associated to the volumetric heat source constant that corresponds to this cell set ID.

heat_source_type

Description : Defines the type of heat source that is defined in the namelist. The two allowed types are mutually exclusive.

Pysical Dimension : N/A

Type : string

Default : none

Valid Values : `'moving gaussian'`, `'piecewise constant'`

If "`moving gaussian`" is specified, the namelist must not contain either of the parameters

and

.

heat_source_center

Description : Defines the coordinates (xc_1, xc_2, xc_3) of the centroid of the heat source.

Together with the `heat_source_radius` these two parameters determine the portion of the domain over which the heat source is applied at initial conditions.

Physical Dimension : L

Type : `real` triplet

Default : 0.0, 0.0, 0.0

Valid values : $(-\infty, \infty)$

heat_source_constant

Description : Defines the total power being added to the domain. Negative values represent a heat sink. A non-zero value of this input variable triggers the routine which specifies a volumetric heat source to be called.

Physical Dimension : Energy / t

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

heat_source_radius

Description : Defines the size of the heat source (r_1, r_2, r_3). The heat source can have an ellipsoidal shape of the type:

$$\frac{(x_1)^2}{r_1^2} + \frac{(x_2)^2}{r_2^2} + \frac{(x_3)^2}{r_3^2} = 1 \tag{11.1}$$

Where (x_1, x_2, x_3) are distances from the center of a cell to the the centroid of the heat source.

NOTE : See the Physics and Algorithm Manual for the complete heat source equation.

Physical Dimension : L

Type : `real` triplet

Default : 0.0, 0.0, 0.0

Valid values : $(-\infty, \infty)$

heat_source_time

Description : The time interval during which the heat source is applied to the domain

Physical Dimension : t

Type : `real`

Default : 0.0, 0.0

Valid values : two increasing real numbers in range $(-\infty, \infty)$

`travel_speed`

Description : Defines the velocity of the center of the heat source
Together with the `heat_source_radius` and `heat_source_center` these three parameters determine the portion of the domain over which the heat source is applied throughout the problem.

Physical Dimension : L/t

Type : `real` triplet

Default : 0.0, 0.0, 0.0

Valid values : $(-\infty, \infty)$

`volumetric_heat_source`

Description : A heat source (power per unit volume) that is either applied uniformly to all cells in the domain, or uniformly to cells that are specified by a list of cell sets `cell_set_id_list` and is zero elsewhere, or, when a list of constants is specified each is applied uniformly to the corresponding cell set that is specified in `cell_set_id_list`.

Physical Dimension : M/Lt^3

Type : `real`

Default : none

Valid values : $(-\infty, \infty)$

Chapter 12

INDUCTION_COIL Namelist

Overview

The variables in an INDUCTION_COIL namelist specify the physical characteristics of an induction coil that is to produce an external magnetic field to drive the electromagnetic Joule heat calculation. Figure 12.1 shows the idealized model of a coil that is used to analytically evaluate the driving field. The coil axis is assumed to be oriented with the problem symmetry axis as defined in the ELECTROMAGNETICS namelist. Multiple coils may be specified; the net driving field is the superposition of the fields due to the individual coils, and a spatially uniform field that can be specified in the ELECTROMAGNETICS namelist.

The coils carry a sinusoidally-varying current with a common frequency and phase. The frequency is specified in the ELECTROMAGNETICS namelist, while the phase value is irrelevant due to the time averaging of the calculated Joule heat. Each coil, however, has an independent current amplitude which is specified here. In addition, the current and frequency may be piecewise constant functions of time.

INDUCTION_COIL Namelist Features

Required / Optional : Optional

Single / Multiple Instances : Multiple

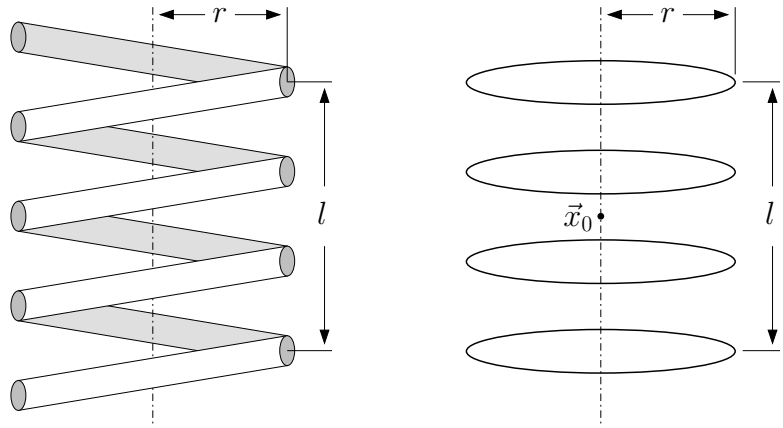

Figure 12.1: Physical 4-turn helical coil with extended wire cross section (left), and its idealized model as a stacked array of circular current loops (right).

Components

- `Center`
- `Current`
- `Length`
- `NTurns`
- `Radius`

`Center`

Description : A 3-vector $\vec{x}_0$ giving the position of the center of the coil; cf. Figure 12.1.

Physical dimension : L

Type : `real`

Default : $(0, 0, 0)$

Valid values : any 3-vector

Current

Description : Amplitude of the sinusoidally-varying current in the coil.

Physical dimension : I

Type : `real`

Default : none

Valid values : Any single value, or any sequence of values.

Notes : A sequence of up to 32 values may be assigned to this variable in order to specify a time-dependent current amplitude; see `Source_Times` in the `ELECTROMAGNETICS` namelist and Fig. 10.1.

Length

Description : Length l of the coil; cf. Figure 12.1.

Physical dimension : L

Type : `real`

Default : none

Valid values : $(0, \infty)$

Notes : `Length` is not required, nor meaningful, if `NTurns` is 1.

NTurns

Description : Number of turns of the coil; cf. Figure 12.1.

Physical dimension : N/A

Type : `integer`

Default : none

Valid values : Any positive integer.

Radius

Description : Radius r of the coil; cf. Figure 12.1.

Physical dimension : L

Type : `real`

Default : none

Valid values : $(0, \infty)$

Chapter 13

INTERFACES Namelist

Overview

The INTERFACES namelist sets parameters used to evaluate the initial material distribution in cells, as specified by BODY namelists.

INTERFACES Namelist Features

Required / Optional : Optional

Single / Multiple Instances : Single

Components

- Vof_Method
- Int_Particles
- Vof_Particles
- Vof_Tolerance
- Vof_Max_Recursion

Vof_Method

Description : `Vof_Method` determines which algorithm to use to compute initial volume fractions of the various materials in each cell. The current methods both use sampling points, and differ in how the sampling points are determined. The 'points' method uses a two-pass algorithm where Int_Particles points are used to determine which cells contain interfaces and Vof_Particles points are used to calculate the volume fractions in those cells. The 'divide' method uses a recursive divide and conquer algorithm to select the sampling points. The accuracy of the divide and conquer scheme is controlled by Vof_Tolerance, the size of a cell subdivision at which the algorithm terminates, and Vof_Max_Recursion, the maximum number of times a cell can be subdivided.

Physical Dimension : N/A

Type : string

Default : `'points'`

Valid values : `'points', 'divide'`

Int_Particles

Description : `Int_Particles` is the number of particles used to identify cells that initially contain more than one material. The higher this number, the greater the likelihood that multi-material cells will be identified, at the expense of a longer runtime for initialization.

Physical Dimension : N/A

Type : `integer`

Default : 5

Valid values : $(0, \infty)$

Vof_Particles

Description : `Vof_Particles` is the number of particles used to calculate the initial volume fractions of materials in each cell identified by `Int_Particles`. The higher this number, the more accurate the initial material distribution, at the expense of a longer runtime for initialization.

Physical Dimension : N/A

Type : `integer`

Default : 5

Valid values : $(0, \infty)$

Vof_Tolerance

Description : `Vof_Tolerance` is the size of a cell subdivision at which the recursive divide and conquer algorithm no longer subdivides cells. At this size a subcell will be assumed to wholly contain the material at the subcell centroid.

Physical Dimension : L^3

Type : `real`

Default : 0.001*smallest cell

Valid values : $(0.0, \infty)$

Vof_Max_Recursion

Description : `Vof_Max_Recursion` is the maximum number of times a cell can be divided in the divide and conquer algorithm. Large numbers require more stack memory and computation time, but allow for greater accuracy. This parameter is provided mostly as a safety to control stack usage, the accuracy of the algorithm is more easily controlled with Vof_Tolerance. A value of 0 means to not subdivide cells at all.

Physical Dimension : N/A

Type : `integer`

Default : 100

Valid values : $[0, 1000]$

Chapter 14

LINEAR_SOLVER Namelist

Overview

The LINEAR_SOLVER namelist sets parameters used in the solution of linear systems of equations as defined in the NUMERICS namelist.

LINEAR_SOLVER Namelist Features

Required / Optional : Optional

Single / Multiple Instances : Multiple

Components

- Convergence_Criterion
- Krylov_Vectors
- Maximum_Iterations
- Method
- Name
- Output_Mode
- Preconditioning_Method

- `Preconditioning_Preconditioner`
- `Preconditioning_Scope`
- `Preconditioning_Steps`
- `Relaxation_Parameter`
- `Status_Frequency`
- `Stopping_Criterion`

In the sections that follow we refer to the linear system being solved as $Ax = b$, where:

- A is the coefficient matrix,
- x is an estimate of the solution vector,
- b is the source vector,
- N is the number of unknowns,
- r is an estimate of the residual, $r = b - Ax$,
- x_0 is the initial guess for the solution vector, and
- r_0 is the initial estimate of the residual, $r_0 = b - Ax_0$.

`Convergence_Criterion`

Description : Value of error estimate used to determine when convergence has been reached. Meaning depends on `Stopping_Criterion`.

Physical Dimension : N/A

Type : `real`

Default : 10^{-8}

Valid values : $(0, 0.1)$

`Krylov_Vectors`

Description : Number of vectors used by GMRES and FGMRES in orthogonalization. Also determines frequency of restart in GMRES and FGMRES.

Physical Dimension : N/A

Type : `integer`

Default : max(10, min(100,N_{cells}))

Valid Values : $(0, N_{cells})$

Notes :

- relevant only when `Method` = `'gmres'` or `Method` = `'fgmres'`
- In general, as the size of `Krylov_Vectors` increases, the likelihood of successful convergence increases, but so does cost/iteration and memory usage. Start with the default value and increase only if convergence difficulties are encountered.
- To insist that GMRES and FGMRES never restart, set `Krylov_Vectors` to a value greater than or equal to `Maximum_Iterations`.

Maximum_Iterations

Description : Maximum number of iterations allowed.

Physical Dimension : N/A

Type : `integer`

Default : max(20, min(1000, $2 * \sqrt{N_{cells}}$))

Valid values : > 0

Notes : If a solution is not found by the time this number of iterations has been performed, execution stops, and an message is written to the `.err` file. In addition, a file containing the residuals at the last successful iteration is written. The filename will be of the form `prob.name.00000`, where `prob` is the basename of the problem input file and `name` is the name of the linear solver that failed, i.e. `Name`. The format of the file is appropriate for use with the visualization tool GMV.

When comparing values found in the graphics dump of the residuals and the values reported in the `.err` file keep in mind that the values in the error file are norms (and scaled norms) of residuals rather than the residuals themselves.

Method

Description : Algorithm used for solution of linear systems.

Physical Dimension : N/A

Type : string

Default : `'fgmres'`

Valid values :

'cg' - preconditioned Conjugate Gradients (CG)

In the absence of roundoff, CG is guaranteed to converge in N iterations for symmetric positive definite N by N systems. Although the guarantee is lost in the real world, with effective preconditioning it can require significantly fewer than N iterations. Also, it can successfully solve slightly non-symmetric systems.

'gmres' - left-preconditioned Generalized Minimal Residuals (GMRES)

Effective on non-symmetric systems, but more expensive and somewhat more difficult to use due to dependence on the Krylov subspace size, `Krylov_Vectors`.

'fgmres' - preconditioned Flexible GMRES (FGMRES)

Extension of GMRES that implicitly applies right-preconditioning and allows the preconditioner to vary at each iteration. As with GMRES, convergence is dependent on choice of `Krylov_Vectors`.

'tfqmr' - preconditioned Transpose-Free Quasi-minimal Residuals (TFQMR)

An alternative to GMRES for non-symmetric systems. Can converge more quickly than GMRES and requires no additional parameters, but requires two matrix-vector products and two applications of the preconditioner per iteration and can break down (i.e. fail).

'bcgstab' - preconditioned stabilized Bi-Conjugate Gradients (Bi-CGSTAB)

Another alternative to GMRES for non-symmetric systems, with the same advantages and disadvantages over GMRES as TFQMR but different convergence behavior.

'none' - only call preconditioner

In this case, only the preconditioner is called—there is no linear solution which calls the preconditioner, and no testing for convergence. In this case, we ignore the following `LINEAR_SOLVER` namelist entries: `Convergence_Criterion`, `Krylov_Vectors`, `Maximum_Iterations`, and `Stopping_Criterion`. Choosing 'none' is suitable for the case when the `LINEAR_SOLVER` namelist is pointed to by a `NONLINEAR_SOLVER` namelist with `Method` `'ain'`.

Name

Description : Arbitrary but unique string to identify a set of linear solver settings

Physical Dimension : N/A

Type : string

Default : `'default'`

Valid values : arbitrary string

Notes : If `Projection_Linear_Solution` in the `NUMERICS` namelist, or `Linear_Solver_Name` in the `NONLINEAR_SOLVER` namelist, is set to a string other than `'default'`, it should match the `Name` string in one and only one `LINEAR_SOLVER` namelist.

Output_Mode

Description : Controls verbosity of linear solver.

Physical Dimension : N/A

Type : string

Default : `'none'`

Valid values :

- **`'none'`** - quiet
- **`'errors'`** - errors only
- **`'errors+warnings'`** - errors and warnings
- **`'warnings+errors'`** - same as above
- **`'summary'`** - errors, warnings, plus a one-line summary consisting of the iteration number, the norms of the calculated residual and error estimate, and the norms of the true residual and error estimate (if calculated) each time convergence is checked
- **`'iterates'`** - same as `'summary'`, plus the coefficient, preconditioner (if there is one), source, initial guess and converged solution, plus the current iterate and true residual (if computed) at each iteration
- **`'full'`** - same as `'iterates'`, plus more intermediate values computed during the course of each iteration (note that this can generate large amounts of output, and is primarily for debugging)

Preconditioning_Method

Description : Algorithm used to precondition the Krylov iteration

Physical Dimension : N/A

Type : string

Default : `'none'`

Valid values for `Projection_Linear_Solution` :

- **`'none'`** - no preconditioning
- **`'diagonal'`** - multistep weighted diagonal
- **`'jacobi'`** - multistep weighted Jacobi
- **`'ssor'`** - multistep weighted Symmetric Gauss-Seidel, a.k.a. Symmetric Successive Over-Relaxation
- **`'ilu0'`** - incomplete LU factorization (ILU) with no fill-in
- **`'lu'`** - LU decomposition

'2level' - 2-level "multigrid"

Valid values for Viscous_Linear_Solution :

'none' - no preconditioning
'diagonal' - multistep weighted diagonal

Note : relevant when Method = 'cg', 'fgmres', 'gmres', 'tfqmr', or 'bcgstab'

Preconditioning_Preconditioner

Description : If the preconditioner is itself a Krylov subspace method, this variable specifies the algorithm to be used as its preconditioner.

Physical Dimension : N/A

Type : string

Default : 'none'

Valid values :

- when Preconditioning_Method = 'cg':

  'none' - no preconditioning
  'jacobi' - multistep weighted Jacobi
  'ssor' - multistep weighted Gauss-Seidel, a.k.a. Symmetric Successive Over-Relaxation
  'ilu0' - incomplete LU factorization (ILU) with no fill-in
  'lu' - LU decomposition

- when Preconditioning_Method = '2level':

  'none' - no preconditioning
  'ssor' - multistep weighted Gauss-Seidel, a.k.a. Symmetric Successive Over-Relaxation

Note : relevant only if Preconditioning_Method is '2level'.

Preconditioning_Steps

Description : Number of passes to be performed in Jacobi or SSOR preconditioning

Physical Dimension : N/A

Type : integer

Default : 1

Valid values : > 0

Note : valid only when:

- `Preconditioning_Method` is set to `'jacobi'` or `'ssor'` (in which case `Preconditioning_Preconditioner` is not required), or when
- `Preconditioning_Method` is set to `'2level'` *and* `Preconditioning_Preconditioner` is set to `'jacobi'` or `'ssor'`.

Preconditioning_Scope

Description : For parallel runs, determines whether preconditioner is global (requiring communication each time the preconditioner is applied) or local (requiring no communication).

Physical Dimension : N/A

Type : string

Default : `'global'`

Valid values :

`'local'`
`'global'`

Relaxation_Parameter

Description : Relaxation parameter used in Jacobi and SSOR preconditioning

Physical Dimension : N/A

Type : `real`

Default : 0.90

Valid values : $(0, 2)$

Note : valid only when:

- `Preconditioning_Method` is set to `'jacobi'` or `'ssor'` (in which case `Preconditioning_Preconditioner` is not required), or when
- `Preconditioning_Method` is set to `'2level'` *and* `Preconditioning_Preconditioner` is set to `'jacobi'` or `'ssor'`

Status_Frequency

Description : Frequency of linear solver status reports to the tty

Physical Dimension : N/A

Type : `integer`

Default : 0

Valid values : ≥ 0

Note : If set to a positive integer value n, a one-line status report will be sent to the tty every n iterations containing the iteration number as well as $\|r\|$, the relative change in $\|r\|$, and the current error estimate for that iteration.

Note that this output will occur each cycle (but only for the linear solver for which `Status_Frequency` is set).

Stopping_Criterion

Description : Test used to estimate error and determine when convergence has been reached. Value is set by `Convergence_Criterion`.

Physical Dimension : N/A

Type : string

Default : `'||r||'`

Valid values :

- **`'||r||/||b||'`** - good for most situations, but can lead to difficulties when $\|A\|\|x\| >> \|b\|$ - in that case, use the following criterion
- **`'||r||/(||A||*||x||+||b||)'`** - useful in cases mentioned above, but note that it requires $\|A\|$ (or some estimate), which may not be available - in addition, it increases the cost of GMRES somewhat
- **`'||r||/||r0||'`** - useful, but suffers from dependence on the initial guess
- **`'||r||/||x||'`** - useful, but increases the cost of GMRES and FGMRES significantly and isn't dimensionless
- **`'||r||'`** - note that this test isn't scaled by some other characteristic "size" of the system
- **`'||x-xold||/||x||'`** - inappropriate for nonstationary methods such as CG and GMRES - NOT RECOMMENDED

Notes : Unfortunately, in the current version of the code it is difficult to automatically set different default stopping criteria for different physics. Hence the user must set the stopping criterion explicitly. Recommended criteria for the various physics / numerics are shown in the following table:

Physics / Numerics	`LINEAR_SOLVER` namelist	**Criterion**
Pressure Poisson	`Projection_Linear_Solution`	$\|r\|$
Viscous Stress	`Viscous_Linear_Solution`	$\|r\|/\|b\|$
Heat Transfer / Phase Change	`Linear_Solver_Name`	$\|r\|/\|r0\|$

For the Pressure Poisson equation, used in the projection-step of the flow solution algorithm, the choice of $\|r\|$ makes it possible to relate the convergence criterion to the divergence error measured at each time step.

The divergence error is computed in terms of the fluxing (face) velocities as

$$\epsilon_{div} = \| \sum_f \mathbf{u}_f \cdot \mathbf{n}_f A_f \; \delta t / V \|$$

where $\mathbf{u}_f$ is the face-velocity, $\mathbf{n}_f$ is the face-normal, A_f is the face-area, δt is the time-step, and V is the cell volume.

Internally, the pressure Poisson problem is scaled in a non-dimensional way so that $\|r\| \approx \epsilon_{div}$. Thus, choosing $\|r\|$ as the convergence criteria ensures that the L_2 norm of the divergence error will be bounded by the specified convergence criteria for the pressure Poisson solve.

For the viscous stress terms, the default preconditioning method is chosen to be `diagonal`. This is adequate for most advectively dominated flow problems, but may not be sufficient in the Stokes' flow limit. This default is only active when `Inviscid = .false` in the `PHYSICS` namelist.

Chapter 15

MATERIAL Namelist

Overview

The MATERIAL namelist identifies materials and specifies their properties. Materials are refered to in the BODY, BC and PHASE_CHANGE_PROPERTIES namelists. Each material to be used in the calculation (including materials to be added in later restart segments), must be specified by its own MATERIAL namelist.

It is important to note that in the current version of the code, all materials connected to one another by a phase change (as described in PHASE_CHANGE_PROPERTIES namelists) must have the same value for Density.

Many material properties can depend on the material temperature and/or species concentration in TRUCHAS. Most materials use the general polynomial form:

$$\Phi\left(\chi_i, T_i\right) = \sum_k c_{T,k}\left(T_i - T_{ref}\right)^{e_{T,k}} + \sum_m c_{\chi,m}\left(\chi_i - \chi_{ref}\right)^{e_{\chi,m}} \tag{15.1}$$

to specify this dependence. T_{ref} is the reference temperature for the material, and χ_{ref} the reference solute concentration for the material.

Material density, however, uses the special form:

$$\rho\left(\chi_i, T_i\right) = \rho_{ref} \cdot \left[1 + \sum_k c_{T,k}\left(T_i - T_{ref}\right)^{e_{T,k}} + \sum_m c_{\chi,m}\left(\chi_i - \chi_{ref}\right)^{e_{\chi,m}}\right] \tag{15.2}$$

in order to be consistent with the usual formulation of the Boussinesq approximation.

The form used for any specific property is specified by a "relation" input variable, which may be any of the following:

- `'constant'` - The value of the property is independent of the local material state
- `'const'` - same as `'constant'`
- `'temperature polynomial'` - The value of the property is calculated from a polynomial in the difference between the local temperature and the reference temperature
- `'temperature_polynomial'` - same as `'temperature polynomial'`
- `'temp polynomial'` - same as `'temperature polynomial'`
- `'concentration polynomial'` - The value of the property is calculated from a polynomial in the difference between the local concentration and the reference concentration
- `'concentration_polynomial'` - same as `'concentration polynomial'`
- `'conc polynomial'` - same as `'concentration polynomial'`
- `'temperature_and_concentration_polynomial'` - The value of the property is calculated from a polynomial in both the difference between the local temperature and reference temperature and the difference between the local concentration and the reference concentration
- `'temp_and_conc_polynomial'` - same as `'temperature_and_concentration_polynomial'`

`MATERIAL` Namelist Features

Required / Optional : Required

Single / Multiple Instances : Multiple. One for each material referenced by other namelists.

Components

- CTE_Conc_Bounds
- CTE_Constants
- CTE_Exponents
- CTE_Relation
- CTE_Temp_Bounds
- Conductivity_Conc_Bounds
- Conductivity_Constants
- Conductivity_Exponents
- Conductivity_Relation
- Conductivity_Temp_Bounds
- Cp_Conc_Bounds
- Cp_Constants
- Cp_Exponents
- Cp_Relation
- Cp_Temp_Bounds
- Density_Change_Coefficients_C
- Density_Change_Coefficients_T
- Density_Change_Exponents_C
- Density_Change_Exponents_T
- Density_Change_Relation
- Density_Conc_Bounds
- Density_Temp_Bounds
- Density
- EM_Conductivity_Conc_Bounds
- EM_Conductivity_Constants
- EM_Conductivity_Exponents
- EM_Conductivity_Relation
- EM_Conductivity_Temp_Bounds
- EM_Permeability_Conc_Bounds
- EM_Permeability_Constants

- EM_Permeability_Exponents
- EM_Permeability_Relation
- EM_Permeability_Temp_Bounds
- EM_Permittivity_Conc_Bounds
- EM_Permittivity_Constants
- EM_Permittivity_Exponents
- EM_Permittivity_Relation
- EM_Permittivity_Temp_Bounds
- Immobile
- Lame1_Conc_Bounds
- Lame1_Constants
- Lame1_Exponents
- Lame1_Relation
- Lame1_Temp_Bounds
- Lame2_Conc_Bounds
- Lame2_Constants
- Lame2_Exponents
- Lame2_Relation
- Lame2_Temp_Bounds
- MTS_b
- MTS_d
- MTS_edot_0i
- MTS_g_0i
- MTS_k
- MTS_mu_0
- MTS_p_i
- MTS_q_i
- MTS_sig_a
- MTS_sig_i
- MTS_temp_0
- Material_Feature
- Material_Name

- Material_Number
- Permeability_Constant
- Priority
- Pwr_Law_A
- Pwr_Law_N
- Pwr_Law_Q
- Pwr_Law_R
- Reference_Concentration
- Reference_Temperature
- SPC_Diffusivity_Constants
- SPC_Diffusivity_Exponents
- SPC_Diffusivity_Relation
- Sound_Speed
- Stress_Reference_Temperature
- Surf10_Conc_Bounds
- Surf10_Constants
- Surf10_Exponents
- Surf10_Material_Number
- Surf10_Relation
- Surf10_Temp_Bounds
- Transmits_Radiation
- Viscoplastic_Model
- Viscosity_Conc_Bounds
- Viscosity_Constants
- Viscosity_Exponents
- Viscosity_Relation
- Viscosity_Temp_Bounds
- Void_Temperature

Every material that may appear in the calculation should be specified by the occurance of a MATERIAL namelist with the same Material_Number.

CTE_Conc_Bounds

Description : An array of two values specifying lower and upper bounds, respectively, for solute concentration in the polynomial expansion of the material coefficient of thermal expansion.

Physical Dimension : N/A

Type : `real`

Default : none

Valid values : $(0.0 - 1.0)$

CTE_Constants

Description : An array of up to 10 coefficients appearing in the polynomial expansion of the material coefficient of thermal expansion. See Equation 15.1 above. If CTE depends on both temperature and concentration, then the odd elements are temperature coefficients and the even elements are concentration coefficients.

Physical Dimension : L/T

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

CTE_Exponents

Description : An array of up to 10 exponents appearing in the polynomial expansion of the material coefficient of thermal expansion. See Equation 15.1 above. If CTE depends on both temperature and concentration, then the odd elements are temperature exponents and the even elements are concentration exponents.

Physical Dimension : N/A

Type : `integer`

Default : 0.0

Valid values : $(-\infty, \infty)$

CTE_Relation

Description : Form of the relation for evaluating the material coefficient of thermal expansion

Physical Dimension : N/A

Type : string

Default : `'constant'`

Valid values :

- `'constant'`
- `'temperature_polynomial'`
- `'temperature_and_concentration_polynomial'`

Notes :

- `'concentration_polynomial'` is not allowed
- see also the Overview for definitions of alternate values
- polynomial expressions also require:
  - `Reference_Temperature` and
  - `Reference_Concentration`

CTE_Temp_Bounds

Description : An array of two values specifying lower and upper bounds, respectively, for temperature in the polynomial expansion of the material coefficient of thermal expansion.

Physical Dimension : T

Type : `real`

Default : none

Valid values : $(-\infty, \infty)$

Conductivity_Conc_Bounds

Description : An array of two values specifying lower and upper bounds, respectively, for solute concentration in the polynomial expansion of the material thermal conductivity.

Physical Dimension : N/A

Type : `real`

Default : none

Valid values : $(0.0 - 1.0)$

`Conductivity_Constants`

Description : An array of up to 10 coefficients appearing in the polynomial expansion of the material thermal conductivity. See Equation 15.1 above. If the conductivity depends on both temperature and concentration, then the odd elements are temperature coefficients and the even elements are concentration coefficients.

Physical Dimension : E/(t*L*T)

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

`Conductivity_Exponents`

Description : An array of up to 10 exponents appearing in the polynomial expansion of the material thermal conductivity. See Equation 15.1 above. If the conductivity depends on both temperature and concentration, then the odd elements are temperature exponents and the even elements are concentration exponents.

Physical Dimension : N/A

Type : `integer`

Default : 0.0

Valid values : $(-\infty, \infty)$

`Conductivity_Relation`

Description : Form of the relation for evaluating the material conductivity

Physical Dimension : N/A

Type : string

Default : `'constant'`

Valid values :

- `'constant'`
- `'temperature_polynomial'`
- `'temperature_and_concentration_polynomial'`

Notes :

- `'concentration_polynomial'` is not allowed
- see also the Overview for definitions of alternate values
- polynomial expressions also require:
  - `Reference_Temperature` and
  - `Reference_Concentration`

`Conductivity_Temp_Bounds`

Description : An array of two values specifying lower and upper bounds, respectively, for temperature in the polynomial expansion of the material thermal conductivity.

Physical Dimension : T

Type : `real`

Default : none

Valid values : $(-\infty, \infty)$

`Cp_Conc_Bounds`

Description : An array of two values specifying lower and upper bounds, respectively, for solute concentration in the polynomial expansion of the specific heat.

Physical Dimension : N/A

Type : `real`

Default : none

Valid values : $(0.0 - 1.0)$

Cp_Constants

Description : An array of up to 10 coefficients appearing in the polynomial expansion of the material specific heat. See Equation 15.1 above. If the specific heat depends on both temperature and concentration, then the odd elements are temperature coefficients and the even elements are concentration coefficients.

Physical Dimension : E/(M*T)

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Cp_Exponents

Description : An array of up to 10 exponents appearing in the polynomial expansion of the material specific heat. See Equation 15.1 above. If the specific heat depends on both temperature and concentration, then the odd elements are temperature exponents and the even elements are concentration coefficients.

Physical Dimension : N/A

Type : `integer`

Default : 0.0

Valid values : $(-\infty, \infty)$

Cp_Relation

Description : Form of the relation for evaluating the material specific heat

Physical Dimension : N/A

Type : string

Default : `'constant'`

Valid values :

- `'constant'`
- `'temperature_polynomial'`
- `'temperature_and_concentration_polynomial'`

Notes :

- `'concentration_polynomial'` is not allowed
- see also the Overview for definitions of alternate values
- polynomial expressions also require:
  - `Reference_Temperature` and
  - `Reference_Concentration`

`Cp_Temp_Bounds`

Description : An array of two values specifying lower and upper bounds, respectively, for temperature in the polynomial expansion of the specific heat.

Physical Dimension : T

Type : `real`

Default : none

Valid values : $(-\infty, \infty)$

`Density`

Description : Density of the material at the reference temperature and concentration. See Equation 15.2 above.

It is important to note that all materials connected to one another by a phase change (as described in `PHASE_CHANGE_PROPERTIES` namelists) must have the same value for Density.

Physical Dimension : M/L^3

Type : `real`

Default : `(none)`

Valid values : $(0, \infty)$

`Density_Change_Coefficients_C`

Description : An array of up to 10 coefficients of species-concentration powers appearing in the polynomial expansion of the material density. See Equation 15.2 above.

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Density_Change_Coefficients_T

Description : An array of up to 10 coefficients of temperature powers appearing in the polynomial expansion of the material density. See Equation 15.2 above.

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Density_Change_Exponents_C

Description : An array of up to 10 exponents of species concentration appearing in the polynomial expansion of the material density. See Equation 15.2 above.

Physical Dimension : N/A

Type : `integer`

Default : 0.0

Valid values : $(-\infty, \infty)$

Notes : Values corresponding to non-zero Density_Change_Constants_C must not be zero.

Density_Change_Exponents_T

Description : An array of up to 10 exponents of temperature appearing in the polynomial expansion of the material density. See Equation 15.2 above.

Physical Dimension : N/A

Type : `integer`

Default : 0.0

Valid values : $(-\infty, \infty)$

Notes : Values corresponding to non-zero Density_Change_Constants_T must not be zero.

`Density_Change_Relation`

Description : Form of the relation for evaluating the material density

Physical Dimension : N/A

Type : string

Default : `'constant'`

Valid values :

- `'constant'`
- `'temperature_polynomial'`
- `'concentration_polynomial'`
- `'temperature_and_concentration_polynomial'`

Notes :

- see also the Overview for definitions of alternate values
- polynomial expressions also require:
  - `Reference_Temperature` and
  - `Reference_Concentration`

`Density_Conc_Bounds`

Description : An array of two values specifying lower and upper bounds, respectively, for solute concentration in the polynomial expansion of the density.

Physical Dimension : N/A

Type : `real`

Default : none

Valid values : $(0.0 - 1.0)$

Density_Temp_Bounds

Description : An array of two values specifying lower and upper bounds, respectively, for temperature in the polynomial expansion of the density.

Physical Dimension : T

Type : `real`

Default : none

Valid values : $(-\infty, \infty)$

EM_Conductivity_Conc_Bounds

Description : An array of two values specifying lower and upper bounds, respectively, for solute concentration in the polynomial expansion of the material electomagnetic conductivity.

Physical Dimension : N/A

Type : `real`

Default : none

Valid values : $(0.0 - 1.0)$

EM_Conductivity_Constants

Description : An array of up to 10 coefficients appearing in the polynomial expansion of the material electric conductivity. See Equation 15.1 above. If the conductivity depends on both temperature and concentration, then the odd elements are temperature coefficients and the even elements are concentration coefficients.

Physical Dimension : t I^2 / (E L)

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Note : The default units (SI) for electric conductivity are ohm^{-1} m^{-1}; to use other units, see the discussion on units in the chapter on the `ELECTROMAGNETICS` namelist.

EM_Conductivity_Exponents

Description : An array of up to 10 exponents appearing in the polynomial expansion of the material electric conductivity. See Equation 15.1 above. If the conductivity depends on both temperature and concentration, then the odd elements are temperature exponents and the even elements are concentration coefficients.

Physical Dimension : N/A

Type : `integer`

Default : 0.0

Valid values : $(-\infty, \infty)$

EM_Conductivity_Relation

Description : Form of the relation for evaluating the material electric conductivity

Physical Dimension : N/A

Type : string

Default : `'constant'`

Valid values :

- `'constant'`
- `'temperature_polynomial'`
- `'temperature_and_concentration_polynomial'`

Notes :

- `'concentration_polynomial'` is not allowed
- see also the Overview for definitions of alternate values
- polynomial expressions also require:
  - `Reference_Temperature` and
  - `Reference_Concentration`

EM_Conductivity_Temp_Bounds

Description : An array of two values specifying lower and upper bounds, respectively, for temperature in the polynomial expansion of the material electomagnetic conductivity.

Physical Dimension : T

Type : `real`

Default : none

Valid values : $(-\infty, \infty)$

EM_Permeability_Conc_Bounds

Description : An array of two values specifying lower and upper bounds, respectively, for solute concentration in the polynomial expansion of the material electomagnetic permeability.

Physical Dimension : N/A

Type : `real`

Default : none

Valid values : $(0.0 - 1.0)$

EM_Permeability_Constants

Description : An array of up to 10 coefficients appearing in the polynomial expansion of the material *relative* magnetic permeability. See Equation 15.1 above. If the permeability depends on both temperature and concentration, then the odd elements are temperature coefficients and the even elements are concentration coefficients.

Physical Dimension : dimensionless

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

EM_Permeability_Exponents

Description : An array of up to 10 exponents appearing in the polynomial expansion of the material *relative* magnetic permeability. See Equation 15.1 above. If the permeability depends on both temperature and concentration, then the odd elements are temperature exponents and the even elements are concentration coefficients.

Physical Dimension : N/A

Type : `integer`

Default : 0.0

Valid values : $(-\infty, \infty)$

EM_Permeability_Relation

Description : Form of the relation for evaluating the material *relative* magnetic permeability

Physical Dimension : N/A

Type : string

Default : `'constant'`

Valid values :

- `'constant'`
- `'temperature_polynomial'`
- `'temperature_and_concentration_polynomial'`

Notes :

- `'concentration_polynomial'` is not allowed
- see also the Overview for definitions of alternate values
- polynomial expressions also require:
  - `Reference_Temperature` and
  - `Reference_Concentration`

EM_Permeability_Temp_Bounds

Description : An array of two values specifying lower and upper bounds, respectively, for temperature in the polynomial expansion of the material electomagnetic permeability.

Physical Dimension : T

Type : `real`

Default : none

Valid values : $(-\infty, \infty)$

EM_Permittivity_Conc_Bounds

Description : An array of two values specifying lower and upper bounds, respectively, for solute concentration in the polynomial expansion of the material electomagnetic permittivity.

Physical Dimension : N/A

Type : `real`

Default : none

Valid values : $(0.0 - 1.0)$

EM_Permittivity_Constants

Description : An array of up to 10 coefficients appearing in the polynomial expansion of the material *relative* electric permittivity. See Equation 15.1 above. If the permittivity depends on both temperature and concentration, then the odd elements are temperature coefficients and the even elements are concentration coefficients.

Physical Dimension : dimensionless

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

EM_Permittivity_Exponents

Description : An array of up to 10 exponents appearing in the polynomial expansion of the material *relative* electric permittivity. See Equation 15.1 above. If the permittivity depends on both temperature and concentration, then the odd elements are temperature exponents and the even elements are concentration coefficients.

Physical Dimension : N/A

Type : `integer`

Default : 0.0

Valid values : $(-\infty, \infty)$

EM_Permittivity_Relation

Description : Form of the relation for evaluating the material *relative* electric permittivity

Physical Dimension : N/A

Type : string

Default : `'constant'`

Valid values :

- `'constant'`
- `'temperature_polynomial'`
- `'temperature_and_concentration_polynomial'`

Notes :

- `'concentration_polynomial'` is not allowed
- see also the Overview for definitions of alternate values
- polynomial expressions also require:
  - `Reference_Temperature` and
  - `Reference_Concentration`

EM_Permittivity_Temp_Bounds

Description : An array of two values specifying lower and upper bounds, respectively, for temperature in the polynomial expansion of the material electomagnetic permittivity.

Physical Dimension : T

Type : `real`

Default : none

Valid values : $(-\infty, \infty)$

Immobile

Description : Used to specify those materials that are solid, and so do not flow.

Physical Dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.` or `.false.`

`Lame1_Conc_Bounds`

Description : An array of two values specifying lower and upper bounds, respectively, for solute concentration in the polynomial expansion of the first Lame´ constant, λ, in the elastic constitutive equation (Equation 15.3).

Physical Dimension : N/A

Type : `real`

Default : none

Valid values : $(0.0 - 1.0)$

`Lame1_Constants`

Description : An array of up to 10 coefficients appearing in the polynomial expansion of the material first Lame´ constant, λ, in the elastic constitutive equation (Equation 15.3). See Equation 15.1 above concerning the polynomial expansion. If the first elastic constant depends on both temperature and concentration, then the odd elements are temperature coefficients and the even elements are concentration coefficients.

Physical Dimension : M/($t^2 * L$)

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Notes :

- The Lame´ constants λ and G apply to the following thermo-elastic constitutive law (see the *Truchas Physics and Algorithms*):

$$\sigma_{ij} = \lambda e_{kk}\delta_{ij} + 2Ge_{ij} - (3\lambda + 2G)\alpha\delta_{ij}\theta \tag{15.3}$$

- The Lame´ constants are related to Young's modulus E and Poisson's ratio ν as follows:

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} \qquad G = \frac{E}{2(1+\nu)} \tag{15.4}$$

Lame1_Exponents

Description : An array of up to 10 exponents appearing in the polynomial expansion of the material first Lame´ elastic constant (Equation 15.3). See Equation 15.1 above concerning the polynomial expansion. If the first elastic constant depends on both temperature and concentration, then the odd elements are temperature exponents and the even elements are concentration exponents.

Physical Dimension : N/A

Type : `integer`

Default : 0.0

Valid values : $(-\infty, \infty)$

Lame1_Relation

Description : Form of the relation for evaluating the material first elastic constant

Physical Dimension : N/A

Type : string

Default : `'constant'`

Valid values :

- `'constant'`
- `'temperature_polynomial'`
- `'temperature_and_concentration_polynomial'`

Notes :

- `'concentration_polynomial'` is not allowed
- see also the Overview for definitions of alternate values
- polynomial expressions also require:
  - `Reference_Temperature` and
  - `Reference_Concentration`

Lame1_Temp_Bounds

Description : An array of two values specifying lower and upper bounds, respectively, for temperature

in the polynomial expansion of the first Lame´ constant, λ, in the elastic constitutive equation (Equation 15.3).

Physical Dimension : T

Type : `real`

Default : none

Valid values : $(-\infty, \infty)$

`Lame2_Conc_Bounds`

Description : An array of two values specifying lower and upper bounds, respectively, for solute concentration in the polynomial expansion of the second Lame´ constant, λ, in the elastic constitutive equation (Equation 15.3).

Physical Dimension : N/A

Type : `real`

Default : none

Valid values : $(0.0 - 1.0)$

`Lame2_Constants`

Description : An array of up to 10 coefficients appearing in the polynomial expansion of the material second Lame´ constant, G, in the elastic constitutive equation (Equation 15.3). See Equation 15.1 above concerning the polynomial expansion. If the second elastic constant depends on both temperature and concentration, then the odd elements are temperature coefficients and the even elements are concentration coefficients.

Physical Dimension : M/($t^2 * L$)

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

`Lame2_Exponents`

Description : An array of up to 10 exponents appearing in the polynomial expansion of the material second

Lame´ constant (Equation 15.3). See Equation 15.1 above concerning the polynomial expansion. If the second elastic constant depends on both temperature and concentration, then the odd elements are temperature exponents and the even elements are concentration exponents.

Physical Dimension : N/A

Type : `integer`

Default : 0.0

Valid values : $(-\infty, \infty)$

`Lame2_Relation`

Description : Form of the relation for evaluating the material second Lame´ elastic constant

Physical Dimension : N/A

Type : string

Default : `'constant'`

Valid values :

- `'constant'`
- `'temperature_polynomial'`
- `'temperature_and_concentration_polynomial'`

Notes :

- `'concentration_polynomial'` is not allowed
- see also the Overview for definitions of alternate values
- polynomial expressions also require:
  - `Reference_Temperature` and
  - `Reference_Concentration`

`Lame2_Temp_Bounds`

Description : An array of two values specifying lower and upper bounds, respectively, for temperature in the polynomial expansion of the second Lame´ constant, λ, in the elastic constitutive equation (Equation 15.3).

Physical Dimension : T

Type : `real`

Default : none

Valid values : $(-\infty, \infty)$

Material_Feature

Description : The input file must specify one and only one material as 'background'. Truchas uses the background material in its setup phase to fill in any portion of the mesh that is not explicitly defined in the 'Body' namelists. If in some Body namelist the 'Surface_Name' variable is set equal to 'background', then the 'Material_Feature' of the material specified in that body must also be set equal to 'background'.

Physical Dimension : N/A

Type : string

Default : `(none)`

Valid values : 'background'

Material_Name

Description : Descriptive name of the material.

Physical Dimension : N/A

Type : string

Default : `(none)`

Valid values : any character string (written within single quotes)

Material_Number

Description : Identifying number for material.

Physical Dimension : N/A

Type : `integer`

Default : `(none)`

Valid values : $[1, \infty)$

MTS_b

Description : Burger's vector length b in the MTS viscoplastic model

Physical Dimension : L

Type : `real`

Default : `(none)`

Valid values : $[0, \infty)$

Note : See the *Truchas Physics and Algorithms* for the description of the MTS model as used in TRUCHAS.

MTS_d

Description : Constant D used in the temperature dependent shear modulus equation in the MTS viscoplastic model

Physical Dimension : M/($t^2 * L$)

Type : `real`

Default : `(none)`

Valid values : $(-\infty, \infty)$

Note : See the *Truchas Physics and Algorithms* for the description of the MTS model as used in TRUCHAS.

MTS_edot_0i

Description : Reference strain rate $\dot{\epsilon}_{0i}$ used in the MTS viscoplastic model

Physical Dimension : t^{-1}

Type : `real`

Default : `(none)`

Valid values : $[0, \infty)$

Note : See the *Truchas Physics and Algorithms* for the description of the MTS model as used in TRUCHAS.

MTS_g_0i

Description : Material constant g_{0i} used in the MTS viscoplastic model

Physical Dimension : N/A

Type : `real`

Default : `(none)`

Valid values : $[0, \infty)$

Note : See the *Truchas Physics and Algorithms* for the description of the MTS model as used in TRUCHAS.

MTS_k

Description : Boltzmann's constant k for use in the MTS viscoplastic model

Physical Dimension : E/T

Type : `real`

Default : `(none)`

Valid values : any real value is accepted (should be 1.38e-23 J/K or the equivalent in the appropriate units for the simulation)

Note : See the *Truchas Physics and Algorithms* for the description of the MTS model as used in TRUCHAS.

MTS_mu_0

Description : Reference value μ_0 for the temperature dependent shear modulus used in the MTS viscoplastic model

Physical Dimension : M/($t^2 * L$)

Type : `real`

Default : `(none)`

Valid values : $(-\infty, \infty)$

Note : See the *Truchas Physics and Algorithms* for the description of the MTS model as used in TRUCHAS.

MTS_p_i

Description : Exponent term p_i used the MTS viscoplastic model

Physical Dimension : N/A

Type : `real`

Default : `(none)`

Valid values : $[0, \infty)$

Note : See the *Truchas Physics and Algorithms* for the description of the MTS model as used in TRUCHAS.

MTS_q_i

Description : Exponent term q_i used the MTS viscoplastic model

Physical Dimension : N/A

Type : `real`

Default : `(none)`

Valid values : $[0, \infty)$

Note : See the *Truchas Physics and Algorithms* for the description of the MTS model as used in TRUCHAS.

MTS_sig_a

Description : The athermal stress term σ_a in the MTS viscoplastic model

Physical Dimension : M/($t^2 * L$)

Type : `real`

Default : `(none)`

Valid values : $[0, \infty)$

Note : See the *Truchas Physics and Algorithms* for the description of the MTS model as used in TRUCHAS.

MTS_sig_i

Description : A stress term σ_i related to obstacles to dislocation motion in the MTS viscoplastic model

Physical Dimension : M/($t^2 * L$)

Type : `real`

Default : `(none)`

Valid values : $[0, \infty)$

Note : See the *Truchas Physics and Algorithms* for the description of the MTS model as used in TRUCHAS.

MTS_temp_0

Description : Constant T_0 used in the temperature dependent shear modulus equation in the MTS viscoplastic model

Physical Dimension : T

Type : `real`

Default : `(none)`

Valid values : $[0, \infty)$

Note : See the *Truchas Physics and Algorithms* for the description of the MTS model as used in TRUCHAS.

Permeability_Constant

Description : An array of up to 3x10 material directional flow permeabilities. One value is specified for each coordinate axis. The flow permeabilities are the coefficients in the Carman-Koseny porous media drag correlation for flow in the mushy zone of solidifying alloys. These are only used for immobile materials.

Physical Dimension : M/($L^3 * t$)

Type : `real`

Default : 0.0

Valid values : $[0, \infty)$

Priority

Description : Material flow priority. The value is only used if the material is fluid. Lower priority fluids are moved first in the volume tracking method.

Physical Dimension : N/A

Type : `integer`

Default : `(none)`

Valid values : $[1, \infty)$

Pwr_Law_A

Description : Constant term A in the power law viscoplastic model

Physical Dimension : M/(t^2 $*$ L)

Type : `real`

Default : `(none)`

Valid values : $[0, \infty)$

Note : The power law viscoplastic model assumes a relation between the effective strain rate $\dot{\epsilon}$ and effective stress $\bar{\sigma}$ to be:

$$\dot{\epsilon} = A\bar{\sigma}^n \exp \frac{-Q}{RT} \tag{15.5}$$

where A, n, and Q are material properties and R is the gas constant in units consistent with the those of Q and T.

Pwr_Law_N

Description : Stress exponent term n in the power law viscoplastic model

Physical Dimension : (none)

Type : `real`

Default : `(none)`

Valid values : $[0, \infty)$

Note : The power law viscoplastic model is given in Equation 15.5

Pwr_Law_Q

Description : Activation energy Q in the power law viscoplastic model

Physical Dimension : E/mole

Type : `real`

Default : `(none)`

Valid values : $[0, \infty)$

Note : The power law viscoplastic model is given in Equation 15.5

Pwr_Law_R

Description : Gas constant R in the power law viscoplastic model

Physical Dimension : E/(mole * T)

Type : `real`

Default : `(none)`

Valid values : $[0, \infty)$

Note : The power law viscoplastic model is given in Equation 15.5

Reference_Temperature

Description : Reference temperature T_{ref} in Equation 15.1

Physical Dimension : T

Type : `real`

Default : `(none)`

Valid values : $[0, \infty)$

Note : If a temperature dependent property relationship is specified for any material property, then this variable must be explicitly set.

Reference_Concentration

Description : Reference concentration χ_{ref} in Equation 15.1

Physical Dimension : N/A

Type : `real`

Default : `(none)`

Valid values : $[0, \infty)$

Note : If a concentration dependent property relationship is specified for any material property, then this variable must be explicitly set.

Sound_Speed

Description : This variable is only used for void (zero density) fluid materials. In this case, it is the adiabatic sound speed that is used in computing the compressibility of each cell containing the material. Note that this is not a real sound speed, but a numerical artifice used to permit collapse of small void bubbles.

Physical Dimension : L/t

Type : `real`

Default : 0.0

Valid values : $(0.0, \infty)$

SPC_Diffusivity_Constants

Description : An array of up to 10 coefficients appearing in the polynomial expansion of the species diffusion diffusivity. See Equation 15.1 above. This diffusivity is allowed to either be constant or depend on concentration.

Physical Dimension : L^2/t

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

SPC_Diffusivity_Exponents

Description : An array of up to 10 exponents appearing in the polynomial expansion of the material species diffusion diffusivity. See Equation 15.1 above. This diffusivity is allowed to either be constant (in which case the value of `SPC_Diffusivity_Exponents` is disregarded) or dependent on concentration.

Physical Dimension : N/A

Type : `integer`

Default : 0.0

Valid values : $(-\infty, \infty)$

`SPC_Diffusivity_Relation`

Description : Form of the relation for evaluating the Species Diffusion diffusivity.

Physical Dimension : N/A

Type : string

Default : `'constant'`

Valid values :

- `'constant'`
- `'concentration_polynomial'`

Notes :

- `'temperature_polynomial'` is not allowed.
- `'temperature_and_concentration_polynomial'` is not allowed.
- A reference concentration of zero is assumed ($\chi_{ref} = 0$ in Equation 15.1).

`Stress_Reference_Temperature`

Description : Stress-free reference temperature for the thermo-elastic solid mechanics calculation.

Physical Dimension : T

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Surf10_Conc_Bounds

Description : An array of two values specifying lower and upper bounds, respectively, for solute concentration in the polynomial expansion of the material surface tension.

Physical Dimension : N/A

Type : `real`

Default : none

Valid values : $(0.0 - 1.0)$

Surf10_Constants

Description : An array of up to 10 coefficients appearing in the polynomial expansion of the interfacial surface tension. See Equation 15.1 above. If the surface tension coefficient depends on both temperature and concentration, then the odd elements are temperature coefficients and the even elements are concentration coefficients. Note that the current surface-tension model pertains to a surface between two materials only. The user must define the same model (i.e. set of `Surf10_Constants`, `Surf10_Exponents`, and `Surf10_Relation` variables) within the `MATERIAL` Namelist of each material.

Physical Dimension : M/t^2

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Surf10_Exponents

Description : An array of up to 10 exponents appearing in the polynomial expansion of the interfacial surface tension. See Equation 15.1 above. If the surface tension coefficient depends on both temperature and concentration, then the odd elements are temperature exponents and the even elements are concentration exponents. The user must define the same model (i.e. set of `Surf10_Constants`, `Surf10_Exponents`, and `Surf10_Relation` variables) within the `MATERIAL` Namelist of each material.

Physical Dimension : N/A

Type : `integer`

Default : 0.0

Valid values : $(-\infty, \infty)$

Surf10_Material_Number

Description : Material identifying number for interfacial surface tension. Surface tension force is applied at the interface between the material described by the current MATERIAL namelist and the material specified by this variable.

Physical Dimension : N/A

Type : integer

Default : 0

Valid values : $(-\infty, \infty)$

Surf10_Relation

Description : Form of the relation for evaluating the interfacial surface tension. Note that the current surface-tension model pertains to a surface between two materials only. The user must define the same model (i.e. set of Surf10_Constants, Surf10_Exponents, and Surf10_Relation variables) within the MATERIAL Namelist of each material.

Physical Dimension : N/A

Type : string

Default : 'constant'

Valid values :

- 'constant'
- 'temperature_polynomial'
- 'temperature_and_concentration_polynomial'

Notes :

- 'concentration_polynomial' is not allowed
- see also the Overview for definitions of alternate values
- polynomial expressions also require:
  - Reference_Temperature and
  - Reference_Concentration

Surf10_Temp_Bounds

Description : An array of two values specifying lower and upper bounds, respectively, for temperature in the polynomial expansion of the material surface tension.

Physical Dimension : T

Type : `real`

Default : none

Valid values : $(-\infty, \infty)$

Transmits_Radiation

Description : A logical flag that informs truchas whether this material transmits radiation. Useful only for boundary conditions that use the enclosure radiation model.

Physical Dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.`, `.false.`

Viscoplastic_Model

Description : Model used for viscoplastic stress - strain rate relationship in the solid mechanics calculation

Physical Dimension : N/A

Type : string

Default : `'elastic_only'`

Valid values : 'elastic_only', 'mts', 'power_law'

Note : The parameters for the mechanical threshold stress (MTS) model are defined in the physics and algorithms manual.

Viscosity_Conc_Bounds

Description : An array of two values specifying lower and upper bounds, respectively, for solute concen-

tration in the polynomial expansion of the material viscosity.

Physical Dimension : N/A

Type : `real`

Default : none

Valid values : $(0.0 - 1.0)$

`Viscosity_Constants`

Description : An array of up to 10 coefficients appearing in the polynomial expansion of the material dynamic viscosity. See Equation 15.1 above. If the viscosity depends on both temperature and concentration, then the odd elements are temperature coefficients and the even elements are concentration coefficients.

Physical Dimension : M/($L * t$)

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

`Viscosity_Exponents`

Description : An array of up to 10 exponents appearing in the polynomial expansion of the material dynamic viscosity. See Equation 15.1 above. If the viscosity depends on both temperature and concentration, then the odd elements are temperature exponents and the even elements are concentration exponents.

Physical Dimension : N/A

Type : `integer`

Default : 0.0

Valid values : $(-\infty, \infty)$

`Viscosity_Relation`

Description : Form of the relation for evaluating the material dynamic viscosity.

Physical Dimension : N/A

Type : string

Default : `'constant'`

Valid values :

- `'constant'`
- `'temperature_polynomial'`
- `'temperature_and_concentration_polynomial'`

Notes :

- `'concentration_polynomial'` is not allowed
- see also the Overview for definitions of alternate values
- polynomial expressions also require:
  - `Reference_Temperature` and
  - `Reference_Concentration`

`Viscosity_Temp_Bounds`

Description : An array of two values specifying lower and upper bounds, respectively, for temperature in the polynomial expansion of the material viscosity.

Physical Dimension : T

Type : `real`

Default : none

Valid values : $(-\infty, \infty)$

`Void_Temperature`

Description : The temperature to be assigned to all cells that contain only material of zero density. The temperature of such cells cannot be calculated because both the enthalpy and the specific heat are zero. The value of this input variable does not affect the result of the simulation, but it does impact graphical images because of interpolation that takes place between cells, and because it may set the overall scale of contour plots.

Physical Dimension : T

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Chapter 16

MESH Namelist

Overview

The MESH namelist describes the primary ("main") mesh, whether it is to be read from a file or generated on the fly, and other mesh attributes. A secondary ("alternative") mesh can be employed for TRUCHAS calculations involving electromagnetics. This alternative mesh is specified using the ALTMESH namelist.

MESH Namelist Features

Required / Optional : Required

Single / Multiple Instances : Single

Components

- NCell
- Coord
- Ratio
- Fuzz
- Gap_Element_Blocks
- Heps

- `Mesh_File`
- `Mesh_File_Format`
- `Coordinate_Scale_Factor`
- `Partitions_Total`
- `Partitions_Per_Process`

NCell

Description : The total number of cells along each axis in an internally generated Cartesian mesh block. An array of three integers, each ≥ 1. The total number of cells in the mesh will be their product. If the mesh is to be read from a file, this namelist variable is ignored; the total number of cells (as well as the total number of nodes) is read from the file.

Physical Dimension : N/A

Type : `integer`

Default : 1

Valid values : $[1, \infty)$

Coord

Description : Used for internally generated block meshes. An array of six real numbers. The first three are the xyz coordinates of the starting point of the mesh. The second three are the xyz coordinates of the corner point diagonally across from the starting point. The second x coordinate must be greater than the first, and similarly for the pairs of y and z coordinates.

Physical Dimension : Length

Type : `real` (rank 1) array

Default : none

Valid values : $(-\infty, \infty)$

Ratio

Description : Used for internally generated meshes. If used, three values are required, one for each dimension. Values are the ratios of the size of a cell to the size of the next cell along that dimension.

If Ratio is 0 or 1, the cells will be all the same size. Operative (allowed) values of Ratio are in the range $[0.8, 1.2]$. The cell sizes in a dimension are computed by:

$$h_n = W_s * \frac{1 - \texttt{Ratio}}{1 - (\texttt{Ratio})^{\mathrm{ncell}_s}} * (\texttt{Ratio})^{n-1}, \;\; 1 \leq n \leq \mathrm{ncell}_s \; ,$$

where ncell_s is the number of cells in that dimension s, W_s is the total width of the segment, and h_n is the width of the n'th cell in the segment. If `Ratio` is outside the allowed range, a fatal error is generated.

Physical Dimension : N/A

Type : `real` (rank 1) array

Default : none

Valid values : $[0.8, 1.2]$, or 0

Fuzz

Description : Used for internally generated meshes. Array of size `ndim`. Used to move the vertices slightly away from the positions computed from a Cartesian model. Useful for generating a (slightly) non-orthogonal mesh with the internal mesh generator. Only one of Fuzz and Heps can be nonzero.

Physical Dimension : N/A

Type : `real` array

Default : 0.0

Valid values : $[0.0, 0.95]$, although it is generally best to keep `Fuzz` smaller than about 0.3.

Gap_Element_Blocks

Description : A list of block IDs from an Exodus or Genesis mesh that are to be treated as gap elements.

Physical Dimension : N/A

Type : `integer`

Default : 0

Valid values : $[0, \infty)$

Notes : Any block ID in the mesh file can be specified, but elements that are not connected such that they can function as gap elements or not consistent with side set definitions will almost certainly result in incorrect behavior. The code does not check for these inconsistencies.

Heps

Description : Used for internally generated meshes. Defines how much vertices initially computed from a Cartesian model should be moved in a smooth manner away from computed position. The single value is used for all dimensions. Useful for generating a (smooth) non-orthogonal mesh with the internal mesh generator. Only one of Fuzz and Heps can be nonzero.

Physical Dimension : N/A

Type : `real`

Default : 0.0

Valid values : $[0.0, 1.0]$.

Mesh_File

Description : Specifies the name of the file to read the mesh data from. If this variable is set to other than `'none'` then the mesh is read from that file and the internal mesh generator is not used. Since file names are case sensitive, this input string is case sensitive.

Physical Dimension : N/A

Type : string

Default : `'none'`

Valid values : Either relative or absolute path name.

Mesh_File_Format

Description : Specifies the format of the mesh file to be read.

Physical Dimension : N/A

Type : string

Default : `'none'`

Valid values : `'avs'`, `'avs-ucd'`, `'ucd'`, `'AVS'`, `'ngp'`, `'NGP'`, `'ngp-hex'`, `'NGP-HEX'`, `'ideas'`, `'IDEAS'`, `'i-deas'`, `'I-DEAS'`, `'patran'`, `'PATRAN'`, `'patran-n'`, `'PATRAN-N'`, `'ansys'`, `'ANSYS'`, `'FIDAP'`, `'fidap'`, `'FDI'`, `'fdi'`, `'topaz'`, `'topaz2d'`, `'topaz3d'`, `'TOPAZ'`, `'TOPAZ2D'`, `'TOPAZ3D'`, `'topaz-2d'`, `'topaz-3d'`, `'EXODUS'`, `'GENESIS'`, `'exodus'`, `'genesis'`, `'Exodus'`, `'Genesis'`, `'ExodusII'`

Notes : Use `'ExodusII'` to specify genuine Exodus format files. These are sometimes known as Genesis files when the file contains only the mesh description and no results data. The files usually have a `.exo` or `.gen` suffix, and are platform-independent netCDF files (binary). Use `'Exodus'` or `'Genesis'` to specify the ASCII dump of an Exodus file produced by the utility program `exotxt`.

Coordinate_Scale_Factor

Description : Used for internally generated meshes as well as for meshes read from a file. Scales all the vertex coordinates by a factor that must be positive.

Physical Dimension : Length.

Type : `real`

Default : 1.0

Valid values : $(0.0, \infty)$

Partitions_Total

Description : Specifies the total number of partitions to use. Useful when requesting 2-level preconditioning. Must be greater than the number of processors. If `Partitions_Per_Process` is set, then `Partitions_Total` must not be set.

Physical Dimension : N/A

Type : `integer`

Default : -1836547290

Valid values : $[1, \infty)$

Partitions_Per_Process

Description : Specifies the number of partitions to use on each processor. Useful when requesting 2-level preconditioning. If `Partitions_Total` is set, then `Partitions_Per_Process` must not be set.

Physical Dimension : N/A

Type : `integer`

Default : -1836547290

Valid values : $[1, \infty)$

Chapter 17

NONLINEAR_SOLVER Namelist

Overview

The NONLINEAR_SOLVER namelist sets parameters used in the solution of nonlinear systems of equations as defined in the NUMERICS namelist.

NONLINEAR_SOLVER Namelist Features

Required / Optional : Optional

Single / Multiple Instances : Multiple

Components

- AIN_Max_Vectors
- AIN_Vector_Tolerance
- Convergence_Criterion
- Damper_Parameters
- Linear_Solver_Name
- Maximum_Iterations
- Method

- `Name`
- `Perturbation_Parameter`

In the sections that follow we use the following notation.

- N is the number of unknowns
- $F(x) = 0$ is the nonlinear system being solved
- x_0 is the initial estimate of the solution vector
- x_k is the current solution vector
- x_{k+1} is an estimate of the future solution vector
- δx is the change in the solution, $x_{k+1} - x_k$
- $\|\delta x\|_\infty$ is the infinity, or max, norm of the solution delta
- $\|F(x)\|_\infty$ is the infinity, or max, norm of the nonlinear residual
- $\|F(x)\|_2$ is the 2-norm of the nonlinear residual
- $J(x_k)\delta x = -F(x_k)$ is the linear system solved at each nonlinear iteration
- $J(x_k)$ is the Jacobian matrix, $J_{ij} = \frac{\partial F_i(x)}{\partial x_j}$

AIN_Max_Vectors

Description : For the AIN method, the maximum number of acceleration vectors to be used.

Physical Dimension : N/A

Type : `integer`

Default : 20

Valid values : $[0, \infty)$

AIN_Vector_Tolerance

Description : For the AIN method, the cosine tolerance for determining if a vector is so close to the span of subsequent vectors that it should be dropped.

Physical Dimension : N/A

Type : `real`

Default : 0.01

Valid values : $(0, 1)$

`Convergence_Criterion`

Description : Value used to determine when nonlinear convergence has been reached

Physical Dimension : N/A

Type : `real`

Default : 10^{-5}

Valid values : $(0, 0.1)$

Note : We refer to the input value of `Convergence_Criterion` as ϵ.

The nonlinear iteration is stopped when *either* of the following two conditions are met:

- reduction in the 2-norm of the nonlinear residual meets the criterion, i.e.**:**

$$\frac{\|F(x_{k+1})\|_2}{\|F(x_0)\|_2} < \gamma$$

- the relative change in the max-norm of the solution meets the criterion, i.e.**:**

$$\frac{\|\delta x\|_\infty}{\|x_{k+1}\|_\infty} < \gamma$$

where γ is the input desired tolerance, modified using an estimate of the convergence rate, i.e.:

$$\gamma = (1 - \rho)\epsilon$$

and:

$$\rho = \frac{\|x_{k+1} - x_k\|_\infty / \|x_{k+1}\|_\infty}{\|x_k - x_{k-1}\|_\infty / \|x_k\|_\infty}$$

This is an attempt to prevent false convergence if the solution stagnates, but allow iteration to stop if the solution is acceptable.

Damper_Parameters

Description : Parameters used to improve stability of the nonlinear iteration in some cases (α_L and α_H).

Physical Dimension : N/A

Type : `real`

Default : 1.0, 1.0

Valid values : $[0, \infty)$

Note : `Damper_Parameters` is only relevant when `Use_Damper` is set to `.true.` (note that the default value of `Use_Damper` is `.true.`).

The damper is applied as follows. Referring to the two input parameters as α_L and α_U, for lower and upper:

$$\alpha = \min_{1 \le i \le N} \begin{cases} -(1-\alpha_L)\frac{x_{k+1}(i)}{\delta x(i)} & \text{if} \quad x_{k+1}(i) + \delta x(i) < \alpha_L x_{k+1}(i) \\ -(\alpha_H - 1)\frac{x_{k+1}(i)}{\delta x(i)} & \text{if} \quad x_{k+1}(i) + \delta x(i) > \alpha_H x_{k+1}(i) \end{cases}$$

then:

$$\alpha = \max(0.1, \min(1.0, \alpha))$$

and finally:

$$x_{k+1} = x_k + \alpha\, \delta x$$

If convergence problems are encountered during nonlinear solves, try adjusting `Damper_Parameters` or setting `Use_Damper` to `.false.`.

Linear_Solver_Name

Description : String matching the `Name` setting for one `LINEAR_SOLVER` namelist

Physical Dimension : N/A

Type : string

Default : `'default'`

Valid values : arbitrary string

Maximum_Iterations

Description : Maximum number of iterations allowed

Physical Dimension : N/A

Type : `integer`

Default : 30

Valid values : $[0, \infty)$

Method

Description : Algorithm used for solution of nonlinear systems

Physical Dimension : N/A

Type : string

Default : `'nk'`

Valid values :

'default'
'nk'
'ain'

Note : If 'ain' is specified, we ignore the following `NONLINEAR_SOLVER` namelist entries: `Damper_Parameters`, `Perturbation_Parameter`, `Use_Damper`.

Name

Description : Arbitrary but unique string to identify a set of nonlinear solver settings

Physical Dimension : N/A

Type : string

Default : `'default'`

Valid values : arbitrary string

Perturbation_Parameter

Description : Differencing parameter used in the Taylor series approximation of the product of the Jacobian and a vector

Physical Dimension : N/A

Type : `real`

Default : 10^{-6}

Valid values : $(0.0, 1.0)$

Note : In Truchas a Jacobian-Free implementation of Newton's method is used for the nonlinear solution. This algorithm uses a Krylov subspace method to solve the linear systems at each iteration of Newton's method. Since Krylov subspace methods require the coefficient only for matrix-vector products, we use a first-order Taylor series expansion to approximate the Jacobian times a vector:

$$Jv \approx \frac{F(x + \epsilon v) - F(x)}{\epsilon}$$

The accuracy of this approximation is obviously affected by the differencing parameter, ϵ, which in Truchas is computed as:

$$\epsilon = \frac{b\|x_{k+1}\|_\infty}{N\|v\|_2} + b$$

where b is the input variable `Perturbation_Parameter`.

The default value is usually appropriate, so this parameter should be changed only with extreme caution.

Use_Damper

Description : Whether or not to apply damping during the nonlinear iteration

Physical Dimension : N/A

Type : `logical`

Default : `.true.`

Valid values : `.true.` or `.false.`

Note : See `Damper_Parameters` for a description of the damping algorithm.

If convergence problems are encountered during nonlinear solves, try adjusting `Damper_Parameters` or setting `Use_Damper` to `.false.`.

Chapter 18

NUMERICS Namelist

Overview

Most numerical algorithm options and parameters are specified within the NUMERICS namelist, including integration time step constraints and selection mechanisms. The exceptions are the linear and nonlinear solution algorithms options and parameters which are specified in separate LINEAR_SOLVER and NONLINEAR_SOLVER namelists and the electromagnetics solution algorithm options which are specified in the ELECTROMAGNETICS namelist. The NUMERICS namelist allows the user to steer various aspects of the flow, heat transfer, phase change, and thermo-mechanical solution methods through liberal choices of algorithm options and parameters.

NUMERICS Namelist Features

Required / Optional : Required

Single / Multiple Instances : Single

Components

- advection_order_energy
- advection_order_momentum
- Alittle

- Body_Force_Face_Method
- Body_Force_Implicitness
- CSF_Mollify_Passes
- Conduction_Implicitness
- Contact_Distance
- Contact_Norm_Trac
- Contact_Penalty
- Courant_Number
- Cutvof
- Cycle_Max
- Cycle_Number
- Discrete_Ops_Type
- Displacement_Nonlinear_Solution
- Dt_Constant
- Dt_Grow
- Dt_Init
- Dt_Max
- Dt_Min
- Enclosure_Flux_Convergence
- Enclosure_Implicit_Solution
- Enclosure_Max_Iterations
- Enclosure_Use_Cell_Temps
- Energy_Nonlinear_Solution
- Energy_Sensitivity_Solution
- Energy_Sensitivity_Pseudo_Load
- FF_Discrete_Ops_Type
- Flux_Vol_Iter_Max
- Fourier_Number
- HT_Discrete_Ops_Type
- Interface_Area
- Interface_Geometry
- Interface_Smoothing_Length

- Interface_Topology_Model
- Limiter_Type
- Mass_Limiter
- Mass_Limiter_Cutoff
- Mechanical_Energy_Bound
- MinFaceFraction
- Momentum_Solidify_Implicitness
- Porous_Implicitness
- Projection_Linear_Solution
- Sfront_Number
- Stress_Reduced_Integration
- Strain_Limit
- Surften_Number
- t
- Temp_Init_Iter_Max
- Viscous_Implicitness
- Viscous_Linear_Solution
- Viscous_Number
- Volume_Track_Brents_Method
- Volume_Track_Interfaces
- Volume_Track_Iter_Max
- Volume_Track_Iter_Tol
- Volume_Track_Subcycles

advection_order_energy

Description : Order of accuracy of the algorithm for the advection of enthalpy between mesh cells. The first order approximation uses a donor cell method. The second order approximation uses the flux limited method.

Physical Dimension : (none)

Type : `integer`

Default : 1

Valid values : (1,2)

advection_order_momentum

Description : Order of accuracy of the algorithm for the advection of momentum between mesh cells. The first order approximation uses a donor cell method. The second order approximation uses the flux limited method.

Physical Dimension : (none)

Type : `integer`

Default : 1

Valid values : (1,2)

Alittle

Description : A small, positive real number (relative to unity) used to avoid division by zero or to compare against other numbers to deduce relative significance.

Physical Dimension : (none)

Type : `real`

Default : `EPSILON`(x), where x is of type `real`. If the precision of x is double, `EPSILON`(x) returns 1.0^{-16} for most combinations of software (Fortran 90 compiler) and hardware platforms tested.

Valid values : (0.0, 0.001]

Body_Force_Face_Method

Description : Activate the use of a face-centered body-force method. This method is required to achieve a steady-state solution for certain multi-fluid situations that should remain stable. By default, a cell-centered formulation for the body-force is used.

Type : `logical`

Default : `.false.`

Valid values : `.true.`, `.false.`

Body_Force_Implicitness

Description : Degree of time implicitness used for the body-forces in the discrete form of the momentum equations. If θ is the body-force implicitness, then the body-force at time level θ is given by $f^{\theta} = (1-\theta)f^{n} + \theta f^{n+1}$. When θ is zero, a fully explicit treatment is specified, and when θ is unity a fully implicit treatment results.

Physical Dimension : (none)

Type : `real`

Default : 0.5

Valid values : [0.0,1.0]

Notes : The selection of an explicit treatment of the body force does not affect the stability of the flow algorithm, but does reduce the order of accuracy in time. For problems involving fluid and void, the explicit treatment may result in noisy velocities near the interface.

CSF_Mollify_Passes

Description : Number of times to sweep the mesh while smoothing the cell-centered interface "color" function prior to computing mean interface curvature. The smoothed (mollified) color function, also located at cell centers, is derived from an interpolation scheme that reduces to tri-linear interpolation for orthogonal, hexahedral cells.

Physical Dimension : (none)

Type : `integer`

Default : 1

Valid values : [0, 10]

Notes : The color function prior to smoothing is initialized as the volume fraction of the appropriate material. The mollified color function is only used for curvature estimates, *not* for actual interface tracking (mass advection). Too many smoothing passes will broaden the mollified color function interface to many cell widths, resulting in curvature estimates that can be highly erroneous. One to two passes are usually sufficient for curvature estimates.

Conduction_Implicitness

Description : Degree of time implicitness used for the temperature field in the thermal diffusion term for the average mixture enthalpy conservation equation. If θ is the conduction implicitness, then the

temperature T at time level θ is given by $T^\theta = (1-\theta)T^n + \theta T^{n+1}$. When θ is zero, a fully explicit treatment is specified, and when θ is unity a fully implicit treatment results.

Physical Dimension : (none)

Type : `real`

Default : 1.0

Valid values : [0.0,1.0]

Notes : In the absence of heat sources, `Conduction_Implicitness` can be set to any number within the range $\theta = [0.0, 1.0]$. When heat sources are employed, `Conduction_Implicitness` should be set to unity as heat sources are currently evaluated at time level $n+1$.

Contact_Distance

Description : A length scale parameter β for the contact function

$$\lambda = \lambda_s * \lambda_\tau$$

where

$$\lambda_s = \begin{cases} 1 & \text{if } s < 0 \\ 0 & \text{if } s > \beta \\ 2(\frac{s}{\beta} - 1)^3 + 3(\frac{s}{\beta} - 1)^2 & \text{if } 0 < s < \beta \end{cases}$$

and

$$\lambda_\tau = \begin{cases} 1 & \text{if } \tau_n < 0 \\ 0 & \text{if } \tau_n > \tau^* \\ 2(\frac{\tau_n}{\tau^*} - 1)^3 + 3(\frac{\tau_n}{\tau^*} - 1)^2 & \text{if } 0 < \tau_n < \tau^* \end{cases}$$

$s = \hat{n} \cdot (u_k - u_j)$

Physical Dimension : L

Type : `real`

Default : 1.0e-7

Valid values : $(0.0, \infty]$

Notes : The default value is usually a good value for mesh cell sizes in the 1 - 10 mm size range.

Contact_Norm_Trac

Description : A parameter τ^* for the contact function

$$\lambda = \lambda_s * \lambda_\tau$$

where

$$\lambda_s = \begin{cases} 1 & \text{if } s < 0 \\ 0 & \text{if } s > \beta \\ 2(\frac{s}{\beta} - 1)^3 + 3(\frac{s}{\beta} - 1)^2 & \text{if } 0 < s < \beta \end{cases}$$

and

$$\lambda_\tau = \begin{cases} 1 & \text{if } \tau_n < 0 \\ 0 & \text{if } \tau_n > \tau^* \\ 2(\frac{\tau_n}{\tau^*} - 1)^3 + 3(\frac{\tau_n}{\tau^*} - 1)^2 & \text{if } 0 < \tau_n < \tau^* \end{cases}$$

τ_n is the normal traction at the interface where a positive value corresponds to a tensile force normal to the surface.

Physical Dimension : M/($t^2 * L$)

Type : `real`

Default : 1.0e4

Valid values : $[0, \infty]$

Notes : The default value is probably appropriate for materials with elastic constants in the range 10^9 - 10^{11}. This parameter should probably be scaled proportionately for elastic constants that differ from this range.

Contact_Penalty

Description : A penalty factor for the penetration constraint in the contact algorithm. Changing this is probably not a good idea in the current version.

Physical Dimension : (none)

Type : `real`

Default : 1.0e3

Valid values : $[0, \infty]$

Notes :

Courant_Number

Description : The *Courant Number* C is defined as $V\delta t/h$, where δt is the time step, h is some measure of the cell size, and V is the fluid velocity magnitude. Given the current explicit treatment of advection terms in all conservation equations, linear stability theory states that time integration of these terms is stable only when $C \leq 1.0$. Given C, V, and h, a stable timestep δt_c results when $\delta t_c < Ch/V$. A specification of C is equivalent to specifying the maximum allowed advection time step δt_c. Smaller values of C result in a smaller allowed δt_c.

Physical Dimension : (none)

Type : `real`

Default : 0.5

Valid values : (0.0, 1.0]

Notes : For general hexahedral cells (which can be degenerate), estimation of h and V can be somewhat tricky. Currently h is computed as the distance between opposite face centroids, giving two values in 2D and three in 3D. The velocity magnitude V is derived from face-centered fluxing velocities, computed as the sum of opposite face outward-directed velocities (inward velocities are assumed zero in this estimate). If the δt_c estimate gives total outward flux volumes that exceed the given cell volume, then δt_c is further scaled back to ensure outward flux volumes do not exceed cell volumes.

Cutvof

Description : The value of a material cell volume fraction below which that material is ignored. If any material has a cell volume fraction less than `Cutvof`, then that material is deleted from that cell, with all other materials present receiving a proportional increase in volume fraction. The only exception is the "background" material, which if present receives the entire allocation (equal to the volume fraction deleted).

Physical Dimension : (none)

Type : `real`

Default : 10^{-8}

Valid values : (0.0, 1.0)

Notes : Relative to most other volume-fraction-based algorithms, `Cutvof` is defaulted and used at a *much* lower value. If a prototypical value of `Cutvof` equal to 10^{-4} were used, as is the case in most commercial software, local and global mass conservation would suffer, and lack of algorithmic robustness would be masked. The default 10^{-8} value for `Cutvof` yields good results, hence setting it higher is generally not necessary.

Cycle_Max

Description : The maximum cycle number allowed in the given simulation, where one "cycle" corresponds to one integration time step over all physical model equations. The simulation will terminate gracefully when the cycle number reaches `Cycle_Max`.

Physical Dimension : (none)

Type : integer

Default : 1000000

Valid values : $(0, \infty)$

Notes : A simulation will also terminate gracefully if the last entry in the `Output_T` input variable (`real`) array (in the `OUTPUTS` namelist) is exceeded by the current simulation time `t`.

Cycle_Number

Description : The cycle number to be used just prior to the first actual cycle (time step) taken in the given simulation. The first simulation cycle number is then taken to be `cycle_number` $+ 1$.

Physical Dimension : (none)

Type : integer

Default : 0

Valid values : $[1, \infty)$

Notes : The default value of 0 results in the first computational cycle taken to be 1. This input variable is most useful when starting simulations from a restart file that already contains a restart cycle number > 0.

Discrete_Ops_Type

Description : Flag for choosing the numerical reconstruction method used in estimating face-centered (located at cell face centroids) spatial gradients and values of discrete cell-centered data. Specifically, face pressure gradients, face velocity values, face velocity gradients, and face temperature gradient estimations are all controlled by this flag.

Physical Dimension : (none)

Type : string

Default : `'default'`

Valid values : `'default', 'ortho', 'nonortho'`

Notes : Face-centered pressure gradients are needed for cell-centered estimates of the pressure Laplacian (used in the pressure Poisson solve) and for enforcement of solenoidal face-centered velocities. Face-centered velocity gradients are needed for cell-centered estimates of the stress tensor, and face-centered velocity values are needed for estimation of fluxing velocities. Face-centered temperature gradients are needed for cell-centered estimates of the thermal diffusion (heat conduction) term in the average mixture enthalpy solution. If this variable does not appear in the `NUMERICS` namelist, or if it appears and is set to `'default'`, the type of discrete operators to use is chosen by testing the orthogonality of the mesh cells. When *all* mesh cells are found to be orthogonal (to roundoff error) and thus hexahedral, then `Discrete_Ops_Type` defaults to an `'ortho'` method. Otherwise, `Discrete_Ops_Type` defaults to a `'nonortho'` method, namely the Least Squares Linear Reconstruction (LSLR) method. For a detailed discussion of discrete operators in TRUCHAS, consult the *Truchas Physics and Algorithms*. If `'ortho'` or `'nonortho'` is set, the chosen operator type is used, whatever the mesh.

The user can set an overall discrete operator type as above and choose alternative discrete operator types for conductive heat transfer (HT) and/or the projection step in fluid flow (FF). To selectively override an overall default or an explicit overall setting, include in the namelist a different setting for one or both of the variables `FF_Discrete_Ops_Type`, `HT_Discrete_Ops_Type`. Allowed values for these namelist variables include the same as those for `Discrete_Ops_Type`, and add the possibility of Support Operators (SO), a higher order support operator discretization. Please see the *Truchas Physics and Algorithms*. Setting one or both of these variables leaves the overall setting `Discrete_Ops_Type` unmodified for all remaining parts of the code.

`Displacement_Nonlinear_Solution`

Description : A character string pointer to the nonlinear solution algorithm parameters to be used in a Newton-Krylov solution of the nonlinear thermo-elastic viscoplastic equations. This string "points" to a particular `NONLINEAR_SOLVER` namelist if it matches the `Name` input variable string in the `NONLINEAR_SOLVER` namelist.

Physical Dimension : (none)

Type : string

Default : `'default'`

Valid values : arbitrary string

Notes : If this string does not match a `Name` input variable string specified in a `NONLINEAR_SOLVER` namelist, then the default set of nonlinear solution algorithm parameters is used for the thermo-elastic viscoplastic equations.

Dt_Constant

Description : A *constant* integration time step value to be used for all time steps in the simulation

Physical Dimension : t

Type : `real`

Default : (none)

Valid values : $(0.0, \infty)$

Notes : This `Dt_Constant` input variable should be used with *extreme caution*, as its specification overrules all other time step choices that are controlled by linear stability and accuracy considerations. In particular, `NUMERICS` namelist input variables `Dt_Grow`, `Dt_Init`, `Dt_Max`, and `Dt_Min` are all ignored if `Dt_Constant` is specified. Use of `Dt_Constant` is therefore advised only for controlled numerical algorithm experiments, and not for simulations intended for applications analysis and validation.

Dt_Grow

Description : A factor to multiply the current integration time step (δt) for the purpose of estimating the time step used during the next cycle ($\delta t_g = \texttt{Dt_Grow} * \delta t$). This candidate time step (δt_g) is chosen only if all other currently active time step restrictions have not already limited its value below δt_g.

Physical Dimension : (none)

Type : `real`

Default : 1.05

Valid values : $[1.0, 1.2]$

Notes : `Dt_Grow` is restricted to allow at most a 20% increase in the time step from one cycle to the next because higher percentage growth can inhibit temporal accuracy and rapidly lead to large time steps in need of reduction due to stability considerations. `Dt_Grow` is *ignored* if `Dt_Constant` is specified.

Dt_Init

Description : Integration time step value used for the first computational cycle

Physical Dimension : t

Type : `real`

Default : 10^{-6}

Valid values : $(0.0, \infty)$

Notes : The default value of `Dt_Init` is completely arbitrary, as it is *always* problem dependent. Unless a constant time step is desired (via specification of `Dt_Constant`), then, a value for `Dt_Init` should be specified (instead of relying on the default) that is consistent with the time scales of interest for the simulation at hand. A general rule of thumb is to set `Dt_Init` smaller than the time step ultimately desired, thereby allowing the time step to grow gradually and steadily (say, over 10-20 cycles) to the desired time step value.

For restart calculations, the initial time step size is extracted from the restart file and used instead of `Dt_Init`, unless `Ignore_Dt` in the `RESTART` namelist has been set to true.

`Dt_Init` is *ignored* if `Dt_Constant` is specified.

Dt_Max

Description : Maximum allowable value for the time step

Physical Dimension : t

Type : `real`

Default : 10.0

Valid values : $(0.0, \infty)$

Notes : The time step is *not* allowed to exceed this value, even if other accuracy- or stability-based criteria would permit it. The default value of `Dt_Max` is completely arbitrary, as it is *always* problem dependent. Unless a constant time step is desired (via specification of `Dt_Constant`), then, a value for `Dt_Max` should be specified (instead of relying on the default) that is consistent with the time scales of interest for the simulation at hand.

`Dt_Max` is *ignored* if `Dt_Constant` is specified.

Dt_Min

Description : Minimum allowable value for the time step

Physical Dimension : t

Type : `real`

Default : 10^{-6}

Valid values : $(0.0, \infty)$

Notes : If the time step falls below this value, the user is informed as such and the simulation terminates gracefully. Termination occurs at this condition because is it very probable that either numerical algorithm or physical model problems have occurred and the simulation is unable to recover. The default value of `Dt_Min` is completely arbitrary, as it is *always* problem dependent. Unless a constant time step is desired (via specification of `Dt_Constant`), then, a value for `Dt_Min` should be specified (instead of relying on the default) that is consistent with the time scales of interest for the simulation at hand. A good rule of thumb to use for `Dt_Min` is to use a value several orders of magnitude below the time scale of interest.

`Dt_Min` is *ignored* if `Dt_Constant` is specified.

Enclosure_Flux_Convergence

Description : specifies the convergence criterion for calculating radiative fluxes when using the enclosure (viewfactor-based) radiation model.

Physical Dimension : (none)

Type : `real`

Default : 10^{-5}

Valid values : > 0.0

Enclosure_Implicit_Solution

Description : Logical flag that determines whether the flux at enclosure radiation boundary conditions will be calculated using an implicit method during the heat transfer solve or will be calculated explicitly using the previous timestep cell temperatures. Setting this to `.true.` will increase the fidelity of the solution with a corresponding increase in computational cost.

Physical Dimension : (none)

Type : `logical`

Default : `.false.`

Valid values : `.true.`, `.false.`

Enclosure_Max_Iterations

Description : Specifies the maximum number of iterations for calculating flux due to radiation when using the enclosure (viewfactor-based) radiation model.

Physical Dimension : (none)

Type : `integer`

Default : 200

Valid values : > 0

Enclosure_Use_Cell_Temps

Description : Logical flag that controls whether we approximate the face temperatures at the cell boundary using the cell gradient normal to the face, or whether we use the cell temperatures as the face temperatures. Setting this to `.false.` will decrease the speed of the boundary flux calculation but increase accuracy.

Physical Dimension : (none)

Type : `logical`

Default : `.true.`

Valid values : `.true.`, `.false.`

Energy_Nonlinear_Solution

Description : A character string pointer to the nonlinear solution algorithm parameters to be used in a Newton-Krylov solution of the nonlinear average enthalpy conservation equation. This string "points" to a particular `NONLINEAR_SOLVER` namelist if it matches the `Name` input variable string in that namelist.

Physical Dimension : (none)

Type : string

Default : `'default'`

Valid values : arbitrary string

Notes : If this string does not match a `Name` input variable string specified in a `NONLINEAR_SOLVER` namelist, then the default set of nonlinear solution algorithm parameters is used for the average enthalpy conservation equation.

Energy_Sensitivity_Solution

Description : A character string pointer to the linear solution algorithm parameters to be used in the solution of the linear pseudo analysis for the average enthalpy conservation sensitivity equation. This string "points" to a particular LINEAR_SOLVER namelist if it matches the Name input variable string in that namelist.

Physical Dimension : (none)

Type : string

Default : `'default'`

Valid values : arbitrary string

Notes : If this string does not match a Name input variable string specified in a LINEAR_SOLVER namelist, then the default set of linear solution algorithm parameters is used to resolve the average enthalpy conservation sensitivity equation.

Energy_Sensitivity_Pseudo_Load

Description : Logical flag that determines whether the pseudo loads will be calculated using a central difference or forward difference approximation. Setting this to `.false.`(to perform the forward difference approximation rather then the central difference approximation) will decrease the computational cost and accuracy of the solution.

Physical Dimension : (none)

Type : `logical`

Default : `.true.`

Valid values : `.true.`, `.false.`

FF_Discrete_Ops_Type

Description : Flag for choosing the numerical reconstruction method used in estimating face-centered (located at cell face centroids) spatial gradients from values of discrete cell-centered data *specifically* for the projection step in the fluid flow module. Face pressure gradient estimations (or flux estimates) are controlled by this flag.

Physical Dimension : (none)

Type : string

Default : `'default'`

Valid values : `'default', 'ortho', 'nonortho', 'SO'`

Notes : Setting `FF_Discrete_Ops_Type` to be any value other than `'default'` overrides, for fluid flow, the value defined by `Discrete_Ops_Type`. A default value for this flag ensures that the choice made by `Discrete_Ops_Type` is used for fliud flow, if `FF_Discrete_Ops_Type` is not set. The difference between the two flags is that here one can choose the support operators (`'SO'`) approach to explicitly compute the pressure gradient in the fluid flow. For additional information, refer to the notes on `Discrete_Ops_Type` and see the detailed discussion of the discrete operators in the Physics and Algorithm Manual.

Flux_Vol_Iter_Max

Description : Maximum number of iterations allowed in the iterative search for interior face flux volume coordinates in the volume tracking algorithm

Physical Dimension : (none)

Type : `integer`

Default : 10

Valid values : $(0, \infty)$

Notes : For the volume tracking algorithm to compute the flux volumes of each material crossing each cell face, it must compute the volume of truncation formed by the material interface plane cutting through the total face flux volume. The total face flux volume is bounded on one side by the respective face and on the opposite by a mirror image of that face set back (into the cell) a distance roughly equal to the product of the normal face flux velocity and the time step. An iteration is required to locate the points of intersection of the mirror face with the four appropriate cell edges. Input variable `Flux_Vol_Iter_Max` is the maximum allowed number of iterations used in the process of iterative search for these vertices. See the Physics and Algorithm Manual for further details.

Fourier_Number

Description : The upper bound on the mesh Fourier number, Fo $= \alpha \delta t / h^2$, where δt is the time step, h is some measure of the cell size, and α is the thermal diffusivity ($k/\rho C_p$). This parameter serves to impose an upper bound on the time step in connection with the heat transfer physics.

Physical Dimension : (none)

Type : `real`

Default : 10.0

Valid values : $(0.0, \infty)$

Notes : The heat transfer physics uses the θ-method for its time discretization, with the parameter θ given by `Conduction_Implicitness`. In the idealized case of a completely-regular, orthogonal grid with constant diffusion coefficient and standard 7-point finite difference discretization of the Laplacian, the θ-method is unconditionally stable for $\frac{1}{2} \leq \theta \leq 1$. For $0 \leq \theta < \frac{1}{2}$ the method is stable when Fo $\leq (1/6)/(1-2\theta)$. The exact stability condition for the mesh and discretization actually employed is impractical, and the idealized bound is used as guidance instead. So for $0 \leq \theta < \frac{1}{2}$, `Fourier_Number` is restricted to being $\leq (1/6)/(1-2\theta)$, and unrestricted otherwise.

For general hexahedral cells, which can be degenerate, the definition of the mesh Fourier number is imprecise. Currently h^2 is computed as the squared distance between opposite face centroids, giving three possible values for Fo on a cell. The largest of these over all cells in the mesh is taken as Fo for the mesh, and the condition Fo $\leq$ `Fourier_Number` defines an upper bound for the time step.

`HT_Discrete_Ops_Type`

Description : Flag for choosing the numerical reconstruction method used in estimating face-centered (located at cell face centroids) spatial gradients from values of discrete cell-centered data *specifically* for the heat transfer module. Face temperature gradient estimations (or flux estimates) are controlled by this flag.

Physical Dimension : (none)

Type : string

Default : `'default'`

Valid values : `'default', 'ortho', 'nonortho', 'SO'`

Notes : Setting `HT_Discrete_Ops_Type` to be any value other than `'default'` overrides, for heat transfer only, the value defined by `Discrete_Ops_Type`. A default value for this flag ensures that the choice made by `Discrete_Ops_Type` is used for heat transfer, if `HT_Discrete_Ops_Type` is not set. The difference between the two flags is that here one can choose the support operators (`'SO'`) approach to explicitly compute the temperature flux in heat transfer. For additional information, refer to the notes on `Discrete_Ops_Type` and see the detailed discussion of the discrete operators in the Physics and Algorithm Manual.

Interface_Area

Description : Flag for activating the computation of interface areas in the volume tracking algorithm, taken to be the polygonal area of the planar interfaces in each interface cell

Physical Dimension : (none)

Type : `logical`

Default : `.false.`

Valid values : `.true.` or `.false.`

Notes : Input variable `Interface_Area` is not activated by default because it can involve an expensive set of computations, e.g., sorting and ordering points, computing dot products, etc. When computed, however, the result is an accurate estimate of the actual interfacial surface area which is written out to the interface output (`.int`) file. This information can also be used in other interfacial models such as those for surface tension, mushy zone flow, and surface delta functions.

Interface_Geometry

Description : Flag for choosing the interface geometry approximation in the volume tracking algorithm. Interfaces are approximated as a piecewise collection of planes, one per interface cell, with the interface plane equation in each cell given by $\hat{\mathbf{n}} \cdot \mathbf{x} - \rho = 0$. This flag allows the user to specify either a first order (piecewise constant) or second order (piecewise linear) computation of the interface normal $\hat{\mathbf{n}}$ used in each interface plane equation.

Physical Dimension : (none)

Type : string

Default : `'piecewise linear'`

Valid values : `'piecewise linear'`, `'linear'`, `'plic'`, `'piecewise constant'`, `'constant'`, `'slic'`

Notes : A piecewise linear approximation for the interface results in a standard gradient computation for interface normal: $\hat{\mathbf{n}} = \nabla f_k$, where f_k is the volume fraction of material k. A truly second order implementation would be "planarity preserving", i.e., an interface plane translated or rotated through any mesh should remain planar. This algorithm is planarity preserving only in a limited set of circumstances, hence it is not in general second order. See the Physics and Algorithm Manual for details on how the discrete ∇ operator is estimated.

A piecewise constant interface approximation results in a first order approximation for the interface normal, which starts with the piecewise linear approximation outlined above. This interface normal $\hat{\mathbf{n}}$, however, is further restricted by the following rule: if $\hat{n}_x > \hat{n}_y, \hat{n}_z$, then $\hat{n}_y, \hat{n}_z = 0.0$ (and

similarly for other directions). The result is an interface plane that is orthogonal to one of the coordinate axes, depending upon the relative magnitudes of the original second order estimates for normal $\hat{\mathbf{n}}$. This approximation is *always* less accurate than the piecewise linear approximation, hence is only recommended for illustrative and comparative purposes. Commercial software typically employ the piecewise constant approximation, so two identical simulations, each with different choices of `Interface_Geometry`, can illustrate the effect and impact of this algorithmic choice.

`Interface_Smoothing_Length`

Description : Radius of support of the convolution kernel used in estimating the interface normal $\hat{\mathbf{n}}$ by convolving the kernel gradient with the volume fraction field. This normal $\hat{\mathbf{n}}$ is used in the cell interface plane equation $\hat{\mathbf{n}} \cdot \mathbf{x} - \rho = 0$ needed by the volume tracking algorithm.

Physical Dimension : L

Type : `real`

Default : 0.15

Valid values : $(0.0, \infty)$

Notes : The default value of `Interface_Smoothing_Length` is completely arbitrary, as it is *always* problem dependent. The user should therefore select a value appropriate for the simulation. A good rule of thumb for the radius of support is a length that at least encompasses cells that are nearest neighbors to the reference cell. For many cases, with a proper selection of `Interface_Smoothing_Length`, the interface normal resulting from this algorithm often yields higher accuracy (is more planarity preserving) than the `'piecewise linear'` choice for the `Interface_Geometry` input variable. See the Physics and Algorithm Manual for further details.

`Interface_Topology_Model`

Description : Flag for choosing the method used in estimating the interface normal $\hat{\mathbf{n}}$ in the cell interface plane equation $\hat{\mathbf{n}} \cdot \mathbf{x} - \rho = 0$ required by the volume tracking algorithm. Either a convolution method or a more traditional discrete gradient computation for $\hat{\mathbf{n}} = \nabla f_k$ (f_k is the material k volume fraction) can be used.

Physical Dimension : (none)

Type : string

Default : `'least squares model'`

Valid values : `'least squares model'`, `'convolution model'`, `'convolve fast'`

Notes : See the Physics and Algorithm Manual for further details on the algorithms used for estimating interface topology.

Limiter_Type

Description : Second (and higher) order discretizations of conservation law advection terms typically require a method for enforcing monotonicity of the advected data. One method for preserving monotonicity is a "slope limiter", of which there are numerous types. This input variable is a flag for choosing the manner by which the slope limiter is computed.

Physical Dimension : (none)

Type : string

Default : `'Barth'`

Valid values : `'Barth', 'Venkat'`

Notes : Advected data ϕ is defined as *monotonic* if ϕ does not possess any *new* extrema after it has been integrated forward in time with a standard advection equation $\partial\phi/\partial t + \nabla \cdot (\mathbf{u}\phi) = 0$. The advection discretization is weakly monotonic if the new time level data (ϕ^{n+1}) is monotonic relative to the old data (ϕ^n). A certain class of upwind, slope-limiting numerical schemes have been found useful for enforcing weak monotonicity. Two such examples for finite volume, unstructured mesh algorithms are those devised by V. Venkatakrishnan [1] and T. Barth [2]. The Barth slope limiter belongs to a general class of van Leer limiters [3, 4], while the Venkatakrishnan limiter allows a minor amount of oscillatory behavior and therefore may not enforce monotonicity in some cases.

See the Physics and Algorithm Manual for further details on the algorithms used for higher-order discretization of the conservation equation advection terms. *This input variable is currently not active owing to the fact that second order discretization of all advection terms has been disabled.*

Mass_Limiter

Description : Activate the use of mass limiter to avoid the generation of excessively large velocities in partially-filled fluid-void cells where the fluid mass approaches the cutoff volume-fraction, `cutvof`. This option may be required in multiphysics flow-solidification problems when heat transfer rates are high.

Type : `logical`

Default : `.false.`

Valid values : `.true.`, `.false.`

Mass_Limiter_Cutoff

Description : The mass limiter relies on the value of `Cutvof` to specify the the value of a material cell volume fraction below which mass is ignored. The `Mass_Limiter` applies an exponential filter over the range of volume fractions from `Cutvof` to the value of the `Mass_Limiter_Cutoff`. The mass limiter has a maximum damping effect at a volume fraction corresponding to `Cutvof`, and no effect at `Mass_Limiter_Cutoff`.

Physical Dimension : (none)

Type : `real`

Default : 10^{-5}

Valid values : (`Cutvof`, 1.0)

Notes : The use of the `Mass_Limiter` is recommended for filling problems with heat-transfer and phase-change where solidification is occuring and the heat-transfer rates are high. Excessively large values of `Mass_Limiter_Cutoff` can lead to overly damped numerical solutions.

If the input value of `Cutvof` is larger than the default or input value of `Mass_Limiter_Cutoff`, then the value of `Mass_Limiter_Cutoff` is set to a value of 10^{-3} smaller than `Cutvof`. Similarly, if `Cutvof` is larger than 10^{-3}, then `Mass_Limiter_Cutoff` is set to a value of 1.0.

Mechanical_Energy_Bound

Description : Control option for applying the body force at cell faces during the velocity correction step. If the sum of the kinetic energy (per unit mass) and the potential energy (also per unit mass) exceeds `Mechanical_Energy_Bound` the body force is not added at the face. The proper value of this parameter (if the model is to be introduced) is the sum of the kinetic and potential energy (per unit mass) at the inlet to the problem, increased by perhaps 10% to allow for uncertainties in the solution. This model is used to limit the acceleration of isolated droplets of material that occur as a result of volume tracking errors.

Physical Dimension : (none)

Type : real

Default : ∞

Valid values : $[-\infty, \infty]$

MinFaceFraction

Description : This option sets the minimum value of the fluid density associated with any face for the projection solution of the fluid flow equations. It is specified as a fraction of the minimum fluid density (excluding void) of any fluid material specified in the input for the calculation.

Physical Dimension : (none)

Type : real

Default : 10^{-3}

Valid values : $[0.0, \infty]$

Momentum_Solidify_Implicitness

Description : Degree of time implicitness used for the velocity field in the treatment of the momentum deposition associated with solidification. If θ is the time-weighting for the momentum solidification implicitness, then the velocity $\mathbf{u}$ at time level θ is given by $\mathbf{u}^\theta = (1-\theta)\mathbf{u}^n + \theta\mathbf{u}^{n+1}$. When θ is zero, a fully explicit treatment is specified, and when θ is unity a fully implicit treatment results.

Physical Dimension : (none)

Type : `real`

Default : 0.0

Valid values : $[0.0, 1.0]$

Porous_Implicitness

Description : Degree of time implicitness used for the velocity field in the treatment of the porous drag term in the fluid momentum conservation equation. If θ is the porous implicitness, then the velocity $\mathbf{u}$ at time level θ is given by $\mathbf{u}^\theta = (1-\theta)\mathbf{u}^n + \theta\mathbf{u}^{n+1}$. When θ is zero, a fully explicit treatment is specified, and when θ is unity a fully implicit treatment results.

Physical Dimension : (none)

Type : `real`

Default : 0.0

Valid values : $[0.0, 1.0]$

Projection_Linear_Solution

Description : A character string pointer to the linear solution algorithm parameters to be used in a Krylov solution of the linear pressure Poisson equation for the projection phase of the incompressible flow algorithm. This string "points" to a particular LINEAR_SOLVER namelist if the string matches the Name input variable string in the LINEAR_SOLVER namelist.

Physical Dimension : (none)

Type : string

Default : `'default'`

Valid values : arbitrary string

Notes : If this string does not match a Name input variable string specified in a LINEAR_SOLVER namelist, then the default set of linear solution algorithm parameters is used for the pressure Poisson equation.

Sfront_Number

Description : The *Solidification Front Number* S_{f} is defined as $V_{\mathrm{f}}\delta t/h$, where δt is the time step, h is some measure of the cell size, and V_{f} is an estimate of the maximum solidification (or melting) front speed in the system. Given an implicit treatment of solid/liquid phase change in the average mixture enthalpy conservation equations, linear stability theory states that time integration of these terms is unconditionally stable for all phase change time steps δt_{pc} (and hence all values of S_{f}). A specification of S_{f} is equivalent to specifying the maximum allowed time step $\delta t_{\mathrm{pc}} < S_{\mathrm{f}}h/V_{\mathrm{f}}$, hence restricts the time step despite unconditional stability. Smaller values of S_{f} result in a smaller allowed δt_{pc}.

Physical Dimension : (none)

Type : `real`

Default : 0.5

Valid values : $(0.0, \infty)$

Notes : In the presence of solid/liquid phase change, the value of Sfront_Number can be roughly interpreted as the number of cells a solidification (or melting) front is allowed to traverse in a given time step. A value of 0.5, therefore, is equivalent to restricting the time step to ensure that any phase front in the system does not travel more than one-half of a mesh width during that time step. This time step restriction helps to retain temporal accuracy of the enthalpy method relative to much smaller or much larger values of Sfront_Number (and hence δt). The front velocity V_{f} is estimated each time step using the novel technique of Rider and Knoll [5], wherein the specific enthalpy h is postulated to obey a hyperbolic wave equation of the form $\partial h/\partial t + \mathbf{u} \cdot \nabla h = 0$. The front velocity V_{f} is then estimated from a scalar norm of this equation, given as $||\partial h_i/\partial t||_1 + V_{\mathrm{f}}||\nabla h_i||_1 = 0$, where $|| \cdot ||_1$ is

the L_1 norm and h_i is the specific enthalpy of cell i. This scalar approximation is only an estimate, as the phase front speed in general depends on local thermodynamic conditions that can be functions of both space and time. The cell size h is approximated as the norm $||V_i^{1/3}||_1$, where V_i is the volume of cell i.

When a value of `Sfront_Number` of order unity is specified, the result is a powerful, automated technique for time step adjustment that is cognizant of temporal accuracy. This time step restriction only takes effect when solid/liquid phase change is occurring.

Strain_Limit

Description : This parameter controls the use of the ODE integrator in the plastic strain calculation. It should be set to the minimum significant value of the plastic strain increment for a time step. If convergence seems poor when a viscoplastic material model is used, it may help to reduce the value.

Physical Dimension : L/L

Type : real

Default : 1.0e-10

Valid values : $[0.0, \infty)$

Notes : This parameter can not be currently used to control the time step. It may be used for such purposes in future releases.

Surften_Number

Description : The *Surface Tension Number* S is defined as $\delta t(2\pi\sigma/\rho h^3)^{1/2}$, where δt is the time step, h is some measure of the cell size, σ is the surface tension coefficient and ρ is the fluid density. It is used to restrict the time step to avoid capillary instability. Given σ, ρ, and h, a stable time step δt_σ results when $\delta t_\sigma < S(\rho h^3/2\pi\sigma)^{1/2}$. Smaller values of S result in a smaller allowed δt_σ.

Physical Dimension : (none)

Type : `real`

Default : 1.0

Valid values : (0.0, 1.0]

Notes : For general hexahedral cells (which can be degenerate), estimation of h can be somewhat tricky. Currently h is computed as the distance between opposite face centroids, giving two values in 2D and three in 3D. The fluid density ρ is computed from averaging the fluid density in the cell.

t

Description : The simulation time to be used at the beginning of the first computational cycle.

Physical Dimension : t

Type : `real`

Default : 0.0

Valid values : $[0.0, \infty)$

Notes : Although `t` currently has non-negative restrictions, there is no reason in principal why a negative value could not be allowed.

Temp_Init_Iter_Max

Description : Maximum number of iterations allowed in the iterative initialization of cell-centered equilibrium temperatures. This initialization, required just prior to the beginning of the first computational cycle, is based upon user-specified initial temperatures for each body in the computational domain.

Physical Dimension : (none)

Type : `integer`

Default : 100

Valid values : $(0, \infty)$

Notes : One assumption inherent in the average mixture enthalpy model is that a single equilibrium temperature exists in every computational cell. If multiple (≥ 2) materials (termed *bodies* on input) having different temperatures are specified at the beginning of a simulation, then a single equilibrium temperature must be computed in those cells containing intersecting regions of different bodies having different specified temperatures. To do this, an enthalpy-conserving Newton iteration is performed within each of these cells, whereby the total cell enthalpy (defined as the sum of material enthalpies at their specified temperatures) is expressed as the sum of material enthalpies defined at some equilibrium temperature. The iterative algorithm finds this equilibrium temperature, which is then taken to be the average cell temperature prior to the first computational cycle.

The current iterative algorithm does *not* take into account any material phase changes that might occur within a cell during the equilibrium temperature iteration. Since material temperatures might change relative to their specified initial value as a result of this iteration, phase changes *could* occur as well. *This equilibrium temperature initialization scheme must therefore be used with caution until this restriction is relaxed and generalized to include any phase changes that have been specified.* This initialization is *not* used or needed if the simulation is instructed to begin from a restart file. See the Physics and Algorithm Manual for further details on the algorithm used for temperature initialization.

Viscous_Implicitness

Description : Degree of time implicitness used for the velocity field in the treatment of the Newtonian viscous stress term in the fluid momentum conservation equation. If θ is the viscous implicitness, then the velocity $\mathbf{u}$ at time level θ is given by $\mathbf{u}^\theta = (1-\theta)\mathbf{u}^n + \theta\mathbf{u}^{n+1}$. When θ is zero, a fully explicit treatment is specified, and when θ is unity a fully implicit treatment results.

Physical Dimension : (none)

Type : `real`

Default : 0.0

Valid values : $[0.0, 1.0]$

Viscous_Linear_Solution

Description : A character string pointer to the linear solution algorithm parameters to be used in a Krylov solution of the linear velocity equation to incorporate the implicit formulation of the viscous stress tensor. This string "points" to a particular `LINEAR_SOLVER` namelist if it matches the `Name` input variable string in `LINEAR_SOLVER` namelist.

Physical Dimension : (none)

Type : string

Default : `'default'`

Valid values : arbitrary string

Notes : If this string does not match a `Name` input variable string specified in a `LINEAR_SOLVER` namelist, then the default set of linear solution algorithm parameters is used for this equation.

Viscous_Number

Description : The *Viscous Number* V_μ is defined as $\nu\delta t/h^2$, where δt is the time step, h is some measure of the cell size, and ν is the average fluid dynamic viscosity (μ/ρ, where μ is the kinematic viscosity). Given an implicit treatment of the Newtonian stress tensor in the fluid momentum conservation equation, linear stability theory states that the time integration of this term is unconditionally stable for all diffusion time steps δt_μ (and hence all values of V_μ). For an explicit treatment, however, stable timestep δt_μ results only when $\delta t_\mu < V_\mu h^2/\nu$, where the V_μ depends upon the problem dimension

and cell geometry (e.g., V_μ is 1/2, 1/4, and 1/6 in 1-D, 2-D, and 3-D, respectively, for uniform rectilinear cells and constant μ). A specification of V_μ is equivalent to specifying the maximum allowed viscous diffusion time step δt_μ. Smaller values of V_μ result in a smaller allowed δt_μ.

To promote a stable solution, the permitted value of Viscous_Number is limited to the theoretical stability limit (described in the Default section below). That is, values above the stability limit are reduced to it during input processing.

Physical Dimension : (none)

Type : `real`

Default : depends upon the value of Viscous_Implicitness. Set θ to be Viscous_Implicitness. With $\theta = 0$ (explicit simulations) *Viscous Number*=$1/2, 1/4, 1/6$ for 1-D, 2-D and 3-D respectively. For $0.0 < \theta < 1.0$, *Viscous Number* = $\frac{1/2}{1-\theta}, \frac{1/4}{1-\theta}, \frac{1/6}{1-\theta}$ for 1-D, 2-D and 3-D respectively.

Valid values : $[0.0, \infty)$

Notes : *Viscous Number = 0* is treated as a special case indicating that no limitation should be placed upon the time step due to viscous effects.

For general hexahedral cells (which can be degenerate), estimation of h^2 can be somewhat tricky. Currently h^2 is computed as the squared distance between opposite face centroids, giving two values in 2-D and three in 3-D. Multiple candidate values of $\delta t_\mu < V_\mu h^2/\nu$ are then computed (one for each pair of cell faces), with the final time step constraint being taken as the minimum of the candidate values δt_μ.

`Volume_Track_Brents_Method`

Description : Logical flag for choosing the iterative method used in finding the constant ρ in the interface plane equation $\hat{\mathbf{n}} \cdot \mathbf{x} - \rho = 0$ required by the volume tracking algorithm. For each interface cell, a separate plane equation describes the interface, with the constant ρ being needed to properly locate the plane within the cell so as to ensure volume (mass) conservation. A value of `.true.` for this flag selects Brent's method [6] for the nonlinear iterative algorithm, otherwise a standard Newton method is used.

Physical Dimension : (none)

Type : `logical`

Default : `.true.`

Valid values : `.true.` or `.false.`

Notes : This flag defaults to `.true.` because Brent's method, while requiring 1-2 more iterations to converge relative to Newton's method, generally requires less overall CPU time because it requires one less function evaluation per iteration than Newton's method.

Volume_Track_Interfaces

Description : Logical flag for selecting a volume tracking method to represent the kinematics (movement) of *fluid* interfaces through the computational mesh. This option *must* be selected as `.true.` when multiple fluid materials having different densities are defined in a problem, as a volume tracking algorithm is currently the *only* method by which mass advection is approximated.

Physical Dimension : (none)

Type : `logical`

Default : `.false.` if nmat=1, `.true.` otherwise

Valid values : `.true.` or `.false.`

Notes : In future versions of the code the input variable `Volume_Track_Interfaces` is likely to be replaced with a more general mass advection flag that will allow the user to select from numerous algorithms for approximating mass advection. Examples are interface tracking schemes (of which volume tracking is one of many) as well as standard continuum advection schemes.

Volume_Track_Iter_Max

Description : Maximum number of iterations allowed in the nonlinear iterative algorithm used to find the constant ρ in the interface plane equation $\hat{\mathbf{n}} \cdot \mathbf{x} - \rho = 0$ required by the volume tracking algorithm. For each interface cell, a separate plane equation describes the interface, with the constant ρ being needed to properly locate the plane within the cell so as to ensure volume (mass) conservation. This input variable sets a limit on the maximum number of iterations used in finding ρ.

Physical Dimension : (none)

Type : `integer`

Default : 20

Valid values : $(0, 100]$

Notes : Whether Brent's or Newton's method is used for the plane constant (ρ) iteration, convergence rarely requires more than $4 - 6$ iterations, hence the default of 20 should virtually never be reached in the iteration loop. Something else is likely to have gone wrong if this is the case.

Volume_Track_Iter_Tol

Description : Volume fraction tolerance used in declaring convergence in the nonlinear iterative algorithm employed to find the constant ρ in the interface plane equation $\hat{\mathbf{n}} \cdot \mathbf{x} - \rho = 0$ required by the volume

tracking algorithm. For each interface cell, a separate plane equation describes the interface, with the constant ρ being needed to properly locate the plane within the cell so as to ensure volume (mass) conservation. Convergence is attained in each interface cell if the volume fraction truncated by the interface plane is within `Volume_Track_Iter_Tol` of the actual cell volume fraction.

Physical Dimension : (none)

Type : `real`

Default : 10^{-8}

Valid values : $(0.0, 10^{-3}]$

Notes : The default value of `Volume_Track_Iter_Tol` is much smaller than values often used in most other volume tracking algorithms, and much smaller than perhaps necessary to ensure good local and global mass conservation.

`Volume_Track_Subcycles`

Description : Volume tracking time step (δt_{vt}) specification, expressed as a multiplicative factor relative to the time step δt. If `Volume_Track_Subcycles` is 4, then $\delta t_{\mathrm{vt}} = \delta t/4$, or 4 volume tracking time steps are required before the full δt integration is achieved.

Physical Dimension : (none)

Type : `integer`

Default : 2

Valid values : $(0, 20]$

Notes : Given the current naive unsplit advection algorithm [7] used for time integration, adequate "corner coupling" of the volume flux terms in the volume tracking algorithm can only be achieved by subcycling the volume tracking time integration method. In this scheme, time level n volume fractions f^n are updated after one subcycle to time level $*$ volume fractions f^*, then to f^{**} the next subcycle and so on. This process of using updated volume fractions for the next subcycle helps to ensure that volume fractions are adequately "corner coupled". A value for `Volume_Track_Subcycles` of at least 4 is recommended.

Chapter 19

OUTPUTS Namelist

Overview

The OUTPUTS namelist defines the problem end time and various output options.

OUTPUTS Namelist Features

Required / Optional : Required

Single / Multiple Instances : Single

Components

- Enclosure_Diagnostics
- Int_Output_Dt_Multiplier
- Long_Edit_Bounding_Coords
- Long_Output_Dt_Multiplier
- Output_Dt
- Output_T
- Precise_Output

- Probe_Output_Cycle_Multiplier
- Retain_Last_Step
- Short_Output_Dt_Multiplier
- User_Output_Dt_Multiplier
- XML_Output_Dt_Multiplier

Enclosure_Diagnostics

Description : Flag to activate writing a graphics dump to show the enclosures used in the enclosure-based radiation model. The flag will cause TRUCHASto write a face-based GMV file in the output directory tagged with the name ¡*problemroot*¿`.vfdiag.`¡*nprocs*¿`.gmv`. This file will have a face set defined for each enclosure. The face-based variables that are currently output are: the sum of viewfactors to other faces in the enclosure, the viewfactor with the ambient environment, and the sum of all viewfactors for all faces.

Physical Dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.`or `.false.`

Int_Output_Dt_Multiplier

Description : Factor multiplying `Output_Dt` for time interval to write interface output data.

Physical Dimension : N/A

Type : `integer` array

Default : 0

Valid values : ≥ 0

Long_Edit_Bounding_Coords

Description : Minimum and maximum coordinates in each dimension of the volume within which long edits for each cell are written, $x_{min}, x_{max}, y_{min}, y_{max}$, etc.

Physical Dimension : L

Type : `integer` triplet

Default : (none)

Valid values : min $\leq$ max for each dimension

`Long_Output_Dt_Multiplier`

Description : Factor multiplying `Output_Dt` for time interval to write long edits

Physical Dimension : N/A

Type : `integer` array

Default : 0

Valid values : $[0, \infty)$

`Output_Dt`

Description : Output time interval for each output time span

Physical Dimension : t

Type : `real` array

Default : (none)

Valid values : $(0, \infty)$

`Output_T`

Description : A sequence of time values for defining time spans that have distinct output time intervals. The last time is the problem end time.

Physical Dimension : t

Type : `real` array

Default : (none)

Valid values : at least two monotonously increasing numbers ≥ 0

Precise_Output

Description : Flag to force outputs at the exact times determined by the output intervals

Physical Dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.` or `.false.`

Probe_Output_Cycle_Multiplier

Description : Factor multiplying truchas cycle to determine frequency of writing probe output.

Physical Dimension : N/A

Type : `integer`

Default : 1

Valid values : $[1, \infty)$

Retain_Last_Step

Description : Boolean to allow the developer to always output the last successful timestep.

Physical Dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.` or `.false.`

Short_Output_Dt_Multiplier

Description : Factor multiplying `Output_Dt` for time interval to write short edits

Physical Dimension : N/A

Type : `integer` array

Default : 0

Valid values : $[0, \infty)$

User_Output_Dt_Multiplier

Description : Factor multiplying `Output_Dt` for time interval to write user (auxiliary) outputs.

Physical Dimension : N/A

Type : `integer` array

Default : 0

Valid values : $[0, \infty)$

XML_Output_Dt_Multiplier

Description : Factor multiplying `Output_Dt` for time interval to write XML output.

Physical Dimension : N/A

Type : `integer` array

Default : 0

Valid values : $[0, \infty)$

Chapter 20

PARALLEL_PARAMETERS Namelist

Overview

The PARALLEL_PARAMETERS namelist defines the partitioner to be used for dividing a mesh into computational subdomains. This namelist is optional.

If this namelist does not exist in an input file, the code will choose the most appropriate partitioning strategy. If more than one instance of this namelist exists in an input file, the first instance will be used.

Several partitioning methods are available. Chaco is a general-purpose partitioning algorithm that can be used to divide any mesh into subdomains. The resulting subdomains will generally have minimum surface area to volume ratios and cut a minimum number of edges in the mesh.

Orthogonal structured meshes created by TRUCHAS can also be partitioned in a simple cartesian fashion using the Cartesian partitioner, which will block the mesh into structured brick-shaped subdomains.

In the Cartesian case, the namelist also controls the number of processors to be used and how the mesh is to be distributed across those processors. For example, a 6x6x8 cell mesh partitioned with the cartesian partitioner and a processor array of 3x3x2 will result in 18 subdomains, each 2x2x4 cells in size.

PARALLEL_PARAMETERS Namelist Features

Required / Optional : Optional

Single / Multiple Instances : Single

Components

- `Partitioner`
- `Processor_Array`

`Partitioner`

Description : Specifies the partitioner to use for dividing the mesh into subdomains. This is used both for dividing a mesh across processors and for dividing the mesh for use by the 2-level preconditioners. "Chaco" is the best choice for unstructured meshes. "Cartesian" may be used for internally generated meshes when you want to be explicit about the blocking (see "Processor_Array" below). "None" is only used for serial computations without the 2-level preconditioners. "Automatic" lets TRUCHAS decide what to do. In this case, if there is only one partition requested, TRUCHAS won't subdivide the mesh. If the mesh was generated by TRUCHAS (hence is also structured and orthogonal, and the product of the 3 integers in the processor_array equals the number of processes being run in a parallel job, the "cartesian" decomposition algorithm will be selected. Otherwise "Chaco" (the most general algorithm) will be invoked.

If "Cartesian" is selected, the product of the 3 integers in the processor_array must equal the number of processes being used to run the job.

Physical Dimension : N/A

Type : string

Default : 'automatic'

Valid values : 'none', 'cartesian', 'chaco', 'automatic'

`Processor_Array`

Description : Define the cartesian layout of the processors. This is useful only when using "Cartesian" partitioning of an internally generated mesh.

Physical Dimension : N/A

Type : `integer` array

Default : 0

Valid values : If the value is not 0, then the product of the dimensions must equal the total number of partitions. Each dimension of the mesh will be divided into the corresponding number of parts.

Chapter 21

PHASE_CHANGE_PROPERTIES Namelist

Overview

The PHASE_CHANGE_PROPERTIES namelist identifies the phases (materials) for each phase change system, with definitions of the relevant thermodynamic properties. For multiple phase changes in a problem, more than one of these namelist groups will be needed. For example, if solidification followed by a solid-solid phase transformation is to be simulated, i.e, two phase changes are to be simulated, than two PHASE_CHANGE_PROPERTIES namelists are required. The material ID for the first solid phase will appear as the Lo_Temp_Phase_Material_ID in the namelist specifying liquid to solid phase transition and as the Hi_Temp_Phase_Material_ID in the remaining namelist specifying solid to solid phase transition.

PHASE_CHANGE_PROPERTIES Namelist Features

Required / Optional : Required

Single / Multiple Instances : Multiple.

Components

- a2_coeff
- a3_coeff
- deltempnucl

- deltempsigma
- ndmaxnucl
- full_coupling_flag
- Forward_Diffusion_Coefficient
- alphaCnK
- Back_Diffusion_Coefficient
- Eutectic_Concentration
- Eutectic_Temperature
- Hi_Temp_Phase_Material_ID
- Latent_Heat
- Liquidus_Slope
- Liquidus_Temp
- Lo_Temp_Phase_Material_ID
- Melting_Temperature_Solvent
- Melting_Temperature
- Partition_Coefficient_Constants
- Partition_Coefficient_Exponents
- Partition_Coefficient_Relation
- Phase_Change_Type
- Phase_Change_Active
- Phase_Change_Model
- Secondary_Half_Arm_Spacing
- Solidification_Time

a2_coeff

Description : a quadratic coefficient for the dendrite tip cubic growth polynomial; used when Phase_Change_Type is set to either 'grain_growth_rnt', or 'grain_growth_bw'.

Physical Dimension : $m/s/K^2$

Type : real

Default : N/A

Valid values : $[0.0, \infty)$

a3_coeff

Description : a cubic coefficient for the dendrite tip cubic growth polynomial; Used when Phase_Change_Type is set to either 'grain_growth_rnt', or 'grain_growth_bw'.

Physical Dimension : $m/s/K^3$

Type : real

Default : N/A

Valid values : $[0.0, \infty)$

deltempnucl

Description : Mean of the nucleation law Gaussian. The nucleation law specifies the number of nuclei growing at a given undercooling. Used when Phase_Change_Type is set to either 'grain_growth_rnt', or 'grain_growth_bw'

Physical Dimension : K

Type : real

Default : N/A

Valid values : $[0.0, \infty)$

deltempsigma

Description : Standard deviation of the nucleation law Gaussian. The nucleation law specifies the number of nuclei growing at a given undercooling. Used when Phase_Change_Type is set to either 'grain_growth_rnt', or 'grain_growth_bw'

Physical Dimension : K

Type : real

Default : N/A

Valid values : $[0.0, \infty)$

ndmaxnucl

Description : The amplitude of the nucleation law Gaussian. The nucleation law specifies the number of nuclei growing at a given undercooling. Used when Phase_Change_Type is set to either 'grain_growth_rnt', or 'grain_growth_bw'

Physical Dimension : $1/m^3$

Type : real

Default : N/A

Valid values : $[0.0, \infty)$

full_coupling_flag

Description : An experimental feature which allows for a full iterative coupling of the recalsecence/grain growth model with macroscopic heat transfer algorithm. The default value is 0. Full coupling is set with the value of 1. The computation is generally much slower in full coupling. Used when Phase_Change_Type is set to 'grain_growth_bw'

Physical Dimension : 1

Type : integer

Default : 0

Valid values : $0, 1$

alphaCnK

Description : A back diffusion parameter used only if Phase_Change_Model is set to 'clyne_and_kurz'. The microsegregation model of Clyne and Kurz is described in detail in [8]. The parameter can be set directly, or will be computed if Back_Diffusion_Coefficient ($D_{\beta B}$), Secondary_Half_Arm_Spacing (λ) and Solidification_Tim ($t_{solidif}$) are all specified. $alphaCnK = \frac{D_{\beta B}}{\lambda^2/t_{solidif}}$

Physical Dimension : 1

Type : real

Default : N/A

Valid values : $[0.0, \infty)$

Back_Diffusion_Coefficient

Description : Mass diffusion coefficient of the alloy species in the solid phase, which is used in the solute back diffusion model of Clyne and Kurz [8].

Physical Dimension : L^2/t

Type : real

Default : 0.0

Valid values : $[0.0, \infty)$

Eutectic_Concentration

Description : The concentration of the binary eutectic point

Physical Dimension : (none)

Type : real

Default : 0.0

Valid values : $[0.0, 1.0]$

Eutectic_Temperature

Description : The temperature of the binary eutectic point

Physical Dimension : T

Type : real

Default : 0.0

Valid values : $(-\infty, \infty)$

Forward_Diffusion_Coefficient

Description : The coefficient of forward diffusion.

Physical Dimension : L^2/t

Type : real

Default : 0.0

Valid values : non-negative real

Hi_Temp_Phase_Material_ID

Description : The identification number of the material that is part of the phase-change and is stable at temperatures higher than the phase-change temperature interval.

Physical Dimension : (none)

Type : integer

Default : 'none'

Valid values : $[1, \infty)$

Latent_Heat

Description : The change in the specific enthalpy during the phase-change

Physical Dimension : E/M

Type : real

Default : 0.0

Valid values : $(0.0, \infty)$

Liquidus_Slope

Description : The slope of the liquidus line in the phase diagram. As discussed in the *Truchas Physics and Algorithms*, it is assumed that the liquidus can be approximated as a straight line over the range of interest.

Physical Dimension : T

Type : real

Default : 0.0

Valid values : $(-\infty, \infty)$

Liquidus_Temp

Description : The temperature corresponding to the liquidus line in the phase diagram, at the `CONCENTRATION` defined in the `BODY` namelist.

Physical Dimension : T

Type : real

Default : 0.0

Valid values : $(-\infty, \infty)$

Lo_Temp_Phase_Material_ID

Description : Identification number of the material that is part of the phase-change and is stable at temperatures lower than the phase-change temperature interval.

Physical Dimension : (none)

Type : integer

Default : 'none'

Valid values : any material number $[1, \infty)$

Melting_Temperature_Solvent

Description : Transformation temperature of the pure solvent of the alloy, required for `Phase_Change_Type` = `'solid_liquid_alloy'` or `'solid_solid_alloy'` or `'isothermal'`.

Physical Dimension : T

Type : real

Default : 0.0

Valid values : $(-\infty, \infty)$

Melting_Temperature

Description : Transformation temperature of a pure material, needed for `Phase_Change_Type` = `'solid\protect\_` or 'solid_solid_isothermal' or 'isothermal' or 'non_isothermal'. (This is the solidus temperature for the

'non_isothermal' case.)

Physical Dimension : T

Type : real

Default : 0.0

Valid values : $(-\infty, \infty)$

Partition_Coefficient_Constants

Description : The constants involved in the polynomial description of the partition coefficient (see the *Truchas Physics and Algorithms*).

Physical Dimension : (none)

Type : real

Default : 0.0

Valid values : $(0.0, \infty)$

Partition_Coefficient_Exponents

Description : The exponents involved in the polynomial description of the partition coefficient (see the *Truchas Physics and Algorithms*).

Physical Dimension : (none)

Type : real

Default : 0.0

Valid values : $(-\infty, \infty)$

Partition_Coefficient_Relation

Description : Type of mathematical representation to be used for the partition coefficient (see the *Truchas Physics and Algorithms*).

Physical Dimension : (none)

Type : `character(LEN=256)`

Default : `'none'`

Valid values : `'constant'` or `'temperature_polynomial'`

Phase_Change_Active

Description : Activates or deactivates the namelist. This variable will allow you to selectively turn on and off different phase change lists in your input deck.

Physical Dimension : (none)

Type : `logical`

Default : `.true.`

Valid values : `.true.` or `.false.`

Phase_Change_Type

Description : Type of phase change.

Physical Dimension : (none)

Type : `character(LEN=256)`

Default : `'none'`

Valid values :

- `'none'`,
- `'isothermal'`,
- `'non_isothermal'`,
- `'alloy'`

Phase_Change_Model

Description : Type of solidification model to be used with = 'alloy'.

Physical Dimension : (none)

Type : `character(LEN=256)`

Default : `'none'`

Valid values : `'lever'`, `'scheil'`, `'clyne_and_kurz'`, `'grain_growth_rnt'`, or `'grain_growth_bw'`

Secondary_Half_Arm_Spacing

Description : A half-arm spacing between secondary branches of dendrites forming the microstrucutre of a cast specimen. The parameter is used to approximately evaluate the back diffusion parameter for microsegration model of $clyne_and_kurz$. Used only if `Phase_Change_Model` is set to $clyne_and_kurz$.

Physical Dimension : m

Type : real

Default : N/A

Valid values : $(0.0, \infty)$

Solidification_Time

Description : Local solidification time used to approximately evaluate the back diffusion parameter for microsegration model of $clyne_and_kurz$. Used only if `Phase_Change_Model` is set to $clyne_and_kurz$.

Physical Dimension : sec.

Type : real

Default : N/A

Valid values : $(0.0, \infty)$

Chapter 22

PHYSICS Namelist

Overview

The PHYSICS namelist is used to specify the physical models to be enabled in a simulation.

PHYSICS Namelist Features

Required / Optional : Required

Single / Multiple Instances : Single

Components

- Absolute_Zero
- Body_Force
- CSF_Normal
- CSF_Tangential
- Curvature_Model
- Fluid_Flow
- Diffusion_Solver

- Electromagnetics
- Heat_Conduction
- Inviscid
- Phase_Change
- Porous_Flow
- Solid_Mechanics
- Solid_Mechanics_Phase_Change
- Stefan_Boltzmann
- Stokes
- Surface_Tension_Model
- Turbulence_CMU
- Turbulence_KE_Fraction
- Turbulence_Length
- Turbulence_Model
- Void_Pressure

Absolute_Zero

Description :
Value for Absolute Zero in the current system of units. The default value is for Kelvin. Note that if you use Centigrade, the correct value is -273.15. In other words, this is the value in the current temperature scale that corresponds to the absolute zero temperature. This value is constrained to be negative since there are no common temperature scales that have a positive value at 0 Kelvin.

Physical Dimension : T

Type : `real`

Default : 0.0

Valid values : $(-\infty, 0)$

Body_Force

Description : Not actually a force, but rather the acceleration imposed on any fluid in the domain

Physical Dimension : L/t^2

Type : `real` triplet

Default : (0.0, 0.0, 0.0)

Valid values : $(-\infty, \infty)$

CSF_Normal

Description : Enables the calculation of the component of the surface tension force normal to an interface. If `CSF_Normal = .true.` surface tension forces will automatically be calculated using the CSF model.

Physical Dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.` or `.false.`

CSF_Tangential

Description : Enables the calculation of the component of the surface tension force tangential to an interface. If `CSF_Tangential = .true.` surface tension forces will automatically be calculated using the CSF model.

Physical Dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.` or `.false.`

Curvature_Model

Description : Enables curvature computations for problem involving surface tension. A choice of two convolution models is proposed. The first one is based on Rudman's kernel and the second one is based on Williams Kernel.

Physical Dimension : N/A

Type : string

Default : 'off'

Valid values : 'off', 'rudman_kernel', 'williams_kernel'

Diffusion_Solver

Description : Enables the diffusion solver. The type of diffusion system to be solved is determined by the value of the `DIFFUSION_SOLVER` namelist variable `System_Type`.

Type : `logical`

Default : `.false.`

Note : The diffusion solver is not currently co-operable with the other physics solvers (a limitation to be relaxed in future releases). To enable this option `Fluid_Flow`, `Heat_Conduction`, `Solid_Mechanics`, and `Electromagnetics` must be disabled.

Electromagnetics

Description : Enables electromagnetics calculations for problems involving induction heating.

Type : `logical`

Default : `.false.`

Fluid_Flow

Description : Enables the calculation of fluid flow

Physical Dimension : N/A

Type : `logical`

Default : `.true.`

Valid values : `.true.` or `.false.`

Heat_Conduction

Description : Enables the calculation of heat transfer.

Physical Dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.` or `.false.`

Inviscid

Description : Disables the calculation of the viscous term of the flow equations. The inviscid option is incompatible with the Stokes option.

Physical Dimension : N/A

Type : `logical`

Default : `.true.`

Valid values : `.true.` or `.false.`

Phase_Change

Description : Controls whether or not phase change calculations will be done. Note that this is a global on/off switch. If this is set to `.false.`, no phase change calculations will be done. However if this is set to `.true.`, the phase change calculations that will be done are controlled by `Phase_Change_Active` within the Phase_Change_Properties namelist for that phase change. Requires `Heat_Conduction` = `.true.`.

Physical Dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.` or `.false.`

Porous_Flow

Description : Enables the model of flow through porous media, according to the prescription of Voller & Prakash (*Int. J. Heat Mass Transfer 30(8), 1709-1719 (1987)*). Requires `Fluid_Flow = .true.`

Physical Dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.` or `.false.`

Solid_Mechanics

Description : Enables the solid mechanics capability

Physical Dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.` or `.false.`

Solid_Mechanics_Phase_Change

Description : Allows the volume changes associated with solid-solid phase changes to affect the solid mechanics calculation. This requires `Phase_Change` = `.true.` and a difference in density between the two solid phases.

Physical Dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.` or `.false.`

Stefan_Boltzmann

Description :
Stefan-Boltzmann constant for thermal radiation.

Physical Dimension : E/(tL^{2}T^4)

Type : `real`

Default : 5.67×10^{-8}

Valid values : $(0.0, \infty)$

Note : The default value is in Watt/m^{2}K^4.

Stokes

Description : Disables the calculation of momentum advection. Forces `inviscid = .false.`

Physical Dimension : N/A

Type : `logical`

Default : `.false.`

Valid values : `.true.` or `.false.`

Surface_Tension_Model

Description : Enables the calculation of surface tension effects, by specifying the model to be used. `Surface_Tension_Model` will be set to `'csf model'` if either `CSF_Normal` or `CSF_Tangential` is `.true.`. `Surface_Tension_Model` will be set to `'convolution model'` if both `CSF_Normal` and `CSF_Tangential` are `.false.`, unless `Surface_Tension_Model` is 'off'.

Physical Dimension : N/A

Type : string

Default : 'off'

Valid values : 'off', 'csf model', 'convolution model'

Turbulence_CMU

Description : Proportionality constant between the turbulent diffusivity and the product of the turbulent length and velocity scales in the algebraic model (default value applies in most situations). `Turbulence_CMU` is only used if `Turbulence_Model = 'algebraic'`.

Physical Dimension : none

Type : `real`

Default : 0.05

Valid values : $(0.0, \infty)$

Turbulence_KE_Fraction

Description : Fraction of the mean flow kinetic energy that is assumed in the algebraic model to give rise to the turbulent kinetic energy (default value should be good for many situations). `Turbulence_KE_Fraction` is only used if `Turbulence_Model = 'algebraic'`.

Physical Dimension : none

Type : `real`

Default : 0.1

Valid values : $(0.0, 1.0)$

Turbulence_Length

Description : Turbulence length scale used in the algebraic model, which should be chosen to represent the size of the turbulent eddies, for example a third of the radius in a turbulent pipe flow. `Turbulence_Length` is only used if `Turbulence_Model = 'algebraic'`.

Physical Dimension : L

Type : `real`

Default : 0.0254

Valid values : $(0.0, \infty)$

Turbulence_Model

Description : Activates the algebraic turbulence model, with several model parameters, of which the most important (problem-specific) is the turbulence length scale. Requires `Fluid_Flow = .true.`

Physical Dimension : N/A

Type : string

Default : 'none'

Valid values : 'none', 'algebraic'

`Void_Pressure`

Description : The uniform pressure of all void (zero density) regions.

Physical Dimension : M/Lt^2

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

Chapter 23

PROBE Namelist

Overview

The `Probe` namelist defines the name and location of a probe used to diagnose TRUCHAS fields at the probe location. The values of TRUCHAS cell fields at the probe are the fields' values at the nearest cell to the probe. Similarly, the values of the TRUCHAS nodal fields at the probe are the fields' values at the nearest node to the probe. Multiple probes can be defined in the input file. All TRUCHAS fields are outputted at the probe location as lookaside binary files, to be postprocessed in the TRUCHAS postprocessor. Refer to the TRUCHAS User manual for instructions on using the postprocessor.

`Probe` Namelist Features

Required / Optional : Optional for diagnostic studies.

Single / Multiple Instances : Multiple

Components

- `Probe_Name`
- `Probe_Description`
- `Probe_Coords`
- `Probe_Coords_Scale`

Probe_Name

Description : The name of this `Probe`.

Physical Dimension : N/A

Type : string

Default : `Unnamed`

Valid values : Any string

Probe_Description

Description : (Optional) Description of this `Probe`.

Physical Dimension : N/A

Type : string

Default : `None`

Valid values : Any string

Probe_Coords

Description : Used to define the location of this `Probe`. Array of size `ndim`.

Physical Dimension : N/A

Type : `real` array

Default : 0.0

Valid values : $(-\infty, \infty)$

Probe_Coords_Scale

Description : Scales the coordinates used to define the probe location. Note that if the `MESH` `Coordinate_Scale_Facto` variable is specified then this scaling factor must also be specified to ensure clarity.

Physical Dimension : N/A

Type : `real`

Default : 1.0

Valid values : $(-\infty, \infty)$

Chapter 24

RADIATION_ENCLOSURE Namelist

Overview

The RADIATION_ENCLOSURE namelist sets parameters used to calculate view factors for surfaces that have an 'enclosure_radiation' BC_Type.

Typically the user will have one entry for each radiation enclosure defined in the problem. If there is no namelist specified for an enclosure, then default values are used.

It is crucial to pick good values for chaparral_Amax, as well as for parameter 3 in chaparral_rParam and parameter 9 in chaparral_iParam to get an accurate view factor matrix. Please refer to the discussion of these parameters belew. Optimally, there should not be any warning messages from Chaparral when running TRUCHAS. By tuning the three parameters mentioned above one can often get the view factor matrix calculation to run without warning messages.

RADIATION_ENCLOSURE Namelist Features

Required / Optional : Optional

Single / Multiple Instances : Multiple

Components

- EnclosureID
- Method
- inFile
- inFileType
- outFile
- outFileType
- Comment
- Partial
- LinearSolver
- chaparral_Amax
- chaparral_iParam
- chaparral_rParam
- chaparral_symmetry

EnclosureID

Description : This entry identifies which enclosures this particular set of namelist parameters is intended for. The enclosure ID is the first entry specified in the BC_Value array for any boundary condition. More than one BC can share the same enclosure ID allowing an enclosure to be made up of the union of a large number of surfaces, each of which is specified in a BC namelist.

Physical Dimension : N/A

Type : integer

Default : N/A

Valid values : $(0, \infty)$

Method

Description :

Specifies how viewfactors are to be obtained. The choices are 'file' and 'chaparral'. 'file' means read the viewfactors from a file created by some program, and 'chaparral' means use chaparral to calculate the viewfactors from the surfaces in this enclosure.

Note that chaparral is available only to LANL users. Non-LANL users will have to get an independent license for chaparral and install it. For LANL users, you can obtain a copy of the chaparral tarball from the download section of the source forge project of the same name. See the prerequisites seciton of the installation guide for details on how to obtain the tarball.

Physical Dimension : N/A

Type : string

Default : 'chaparral'

Valid values : 'file', 'chaparral'

inFile

Description :

Contains the filename from which to read the view factors when `Method` is set to 'file' The format of the file is as follows:

Record-1**:** 512 characters**:** comment, 512 bytes long

Record-2**:** 2 Integers**:** Number_of_faces Number_of_VF_entries

Record-3**:** NFaces Integers**:** Cell IDs of the nFaces

Record-4**:** NFaces Integers**:** Face-of-cell IDs of the nFaces (goes from 1 to 6)

Record-5**:** NFaces Integers**:** Entry Counts the nFaces

Record-6**:** NVFEntry Integers**:** IDs of corresponding View Factor faces

Record-7**:** NVFEntry Reals**:** Values of corresponding View Factors

The CellID and FaceID for the ambient face is set to 0 and is the last entry in the file. Note that binary files are 'FORTRAN' binary files i.e. they have record lengths embedded in them unlike 'C' binary files. Integers are 4 bytes long, and reals are 8 bytes long.

Physical Dimension : N/A

Type : string

Default : N/A

Valid values : N/A

inFileType

Description : Format of the input file. Can be 'ascii' or 'binary'

Physical Dimension : N/A

Type : string

Default : 'binary'

Valid values : 'ascii', 'binary'

outFile

Description : Contains the filename to which we want to write the view factors. This can be used for example to save viewfactors calculated during one run for use in future runs. See `inFile` for a description of the output file entries.

Physical Dimension : N/A

Type : string

Default : N/A

Valid values : N/A

outFileType

Description : Format of the output file. Can be 'ascii' or 'binary'

Physical Dimension : N/A

Type : string

Default : 'binary'

Valid values : 'ascii', 'binary'

Comment

Description : A comment to be placed on top of the output file

Physical Dimension : N/A

Type : string

Default :

Valid values :

Partial

Description : Is this a partial or complete enclosure. This parameter overrides that in the `chaparral_iParam` entry below.

Physical Dimension : N/A

Type : `logical`

Default : `.true.`

Valid values : `.true.`, `.false.`

LinearSolver

Description : Then name of the linear solver namelist that will be used for solving the flux equatins.

Physical Dimension : none

Type : string

Default :

Valid values :

chaparral_Amax

Description : In the case of partial enclosures, one needs to specify the area of the face at ambient temperature in order to enforce the reciprocity relationship ($A_iF_{ij} = A_jF_{ji}$). This overrides the value in `chaparral_rParam` entry below. To make an educated guess, first estimate the area of the opening of the partial enclosure and use that value. Then observe the output from Chaparral, it will generate a warning about the size of the parial enclosure area, if this value was too small and suggest a minimal value. Use a value that is perhaps 5% larger than the one suggested by Chapparral as a the value for `chaparral_Amax`. If in a subsequent run with this new value for `chaparral_Amax` Chaparral does not print a warning about the minimal size of the partial enclosure area, and your guess for `chaparral_Amax` is close but slightly larger than the minimum partial enclosure area value that

is reported by Chaparral, you can consider your guess reasonable. Note, however, that grossly over estimating amax will cause Chaparral to compute incorrect view factors just as much as grossly under estimating amax.

Physical Dimension : L^2

Type : `real`

Default : twice the size of largest face in enclosure

Valid values : $(0, \infty)$

`chaparral_iParam`

Description : Integer parameters to be passed to Chaparral during viewfactor calculations. This is an array of 21 entries which are defined in the table below.

Physical Dimension : none

Type : `integer`

Default : 1, 3, 0, 1, 0, 0, 10, 25, 10, 100, 100, 1, 1000, 3, 2, 100, 3, 1, 0, -1, 1

Valid values : See Table below

Index	Value Description	Default	Valid Values
1	Method for computing viewfactors	1	0=pairwise, 1=hemicube
2	2D / 3D Geometry	3	2=2D, 3=3D
3	Blocking / non-blocking surfaces	0	0=Blocking, 1=NonBlocking
4	Partial Enclosure? **IGNORED**	-	Ignored. See `Partial`
5	Number of nodes, local **IGNORED**	-	Ignored, calculated using the mesh
6	Number of nodes, global **IGNORED**	-	Ignored, calculated using the mesh
7	Maximum bsp depth	10	See chaparral documentation
8	Mimimum list length per BSP node	25	See Chaparral documentation
9	Maximum Number of Surface Divisions	20	Used with hemicube method
10	Hemicube resolution	200	Resolution of hemicube
11	Number of Samples for visibility Check	16	See Chaparral documentation
12	MC Surface Sampling Method	1	0=random, 1=uniform, 2=jitter, 3=halton, 4=sobol
13	Number of Samples for MC Accuracy	1000	See Chaparral documentation
14	MC Sampling Method	3	0=random, ... See Chaparral documentation
15	Viewfactor Smoothing Method	2	0=none, 1=simple, 2=add, 3=average
16	Max Smoothing Iterations	100	See Chaparral documentation
17	Symmetry method for smoothing	3	1=subtract, 2=add, 3=average
18	Smoothing Output Flag	1	0=no output, higher is more verbose
19	Output exodus file on rowsum error?	0	Not applicable since we do not have exodus libraries
20	Serial / Parallel calculation? **IGNORED**	-	**IGNORED**, set internally by TRUCHAS
21	Output level flag	1	0=no output, higher=more verbose

`chaparral_rParam`

Description : Real parameters to be passed to Chaparral during viewfactor calculations. This is an array of 8 entries which are defined in the table below.

Physical Dimension : none

Type : `real`

Default : 0.0, 10^{-6}, 7.5, , 10^{-6}, 10^{-6}, 2.0, 10^{-8}, 0.0

Valid values : See Table below

Index	Value Description	Default	Valid Values
1	Area of phantom surface **IGNORED**	-	See `chaparral_Amax`
2	spatial tolerance	10^{-6}	See Chaparral documentation
3	Minimum separation for hemicube	7.5	See below
4	MC Convergence Tolerance 1	10^{-6}	See Chaparral documentation
5	MC Convergence Tolerance 2	10^{-6}	See Chaparral documentation
6	Least Squares smoothing weighting power	2.0	See Chaparral documentation
7	Smoothing Error Tolerance	10^{-8}	See Chaparral documentation
8	Rowsum error tolerance for exodus output	-	**IGNORED**

Increasing parameter 3 in `chaparral_rParam`, the minimum distance for the hemicube calculation, will cause more aggressive face refinement in the hemicube method. This should result in a more accurate view factor matrix, albeit at increased computational cost. The refinement of surfaces can be controlled by parameter 9 in `chaparral_iParam`, the maximum number of surface subdivisions. Now run TRUCHAS and observe the output from Chaparral under the VIEWFACTOR CALCULATION heading, to see how many surface subdivisions were maximally desired versus the number of actual surface subdivisions, and increase either parameter 9 in `chaparral_iParam` to increase accuracy, or decrease parameter 3 in `chaparral_rParam` to decrease accuracy of the view factor matrix. The goal is to have the maximally desired surface subdivisions to be smaller or equal to the actual surface subdivisions.

Please refer to the file `SAND95-2049.pdf` in the Chaparral subdirectory `truchas/packages/chaparral-LGPL` for a more in depth discussion of the hemi-cube method (sections 2.2.1 and 2.2.2 in that file). Note also that this report refers to an early version of Chaparral, so software related discussion there is very out of date.

`chaparral_symmetry`

Description : This namelist entry modifies the viewfactor matrix generated by Chaparral to account for symmetric enclosures. Consequently, you can ask for viewfactors for a quarter symmetry mesh and the corresponding faces will be duplicated before sending them to Chaparral for calculation of viewfactors.

Multiple symmetry can be specified on one line as strings separated by commas. Even though symmetry operations are commutative, and so the order does not matter, they are applied in the order in which they are present in the array.

Physical Dimension : none

Type : string array

Default : '– – – – – –'

Valid values :

Value	Value Description
'x'	Reflect across x=0 plane i.e. add face at (-x, y, z) for every face at (x,y,z)
'y'	Reflect across y=0 plane i.e. add face at (x,-y, z) for every face at (x,y,z)
'z'	Reflect across z=0 plane i.e. add face at (x, y, -z) for every face at (x,y,z)
'$< newx >< newy >< newz >$'	Allow general swapping of x, y, and z axes with negations. Each of $< newx >$, $< newy >$, and $< newz >$ provide us information on the new x, new y, and new z axes and can be one of 'x', 'y', 'z', '-x', '-y', '-z'. The minimum length of this argument is 3 (no negations) and the maximum length of the argument is 6 (all axes negated). So, a value of 'x-yz' will provide us an additional face at (x, -y, z) for every face at (x,y,z), or in other words, a reflection in the y axis. Similarly, a value of (y,x,-z) will provide us an additional face at (y,x,-z) for every face at (x,y,z), i.e. a swapping of the x and y axes and a reflection in the z-axis. Note the lack of spaces between the new axis specifications.

Chapter 25

RESTART Namelist

Overview

TRUCHAS is able to use data from a previous calculation to initialize a new calculation. Such a *restart* calculation is invoked by using the '`-r`' commandline argument to the executable with the path name of the restart data file. By default, all appropriate data from the restart file is used. This optional namelist provides variables to limit the restart data that will be used.

RESTART Namelist Features

Required / Optional : Optional

Single / Multiple Instances : single

Components

- `Ignore_Dt`
- `Ignore_Joule_Heat`
- `Ignore_Solid_Mechanics`
- `Temperature_Is_Primary`

Ignore_Dt

Description : When restarting, the initial time step size is normally extracted from the restart file. If this flag is true, then that value is ignored and the value specified by `Dt_Init` from the `NUMERICS` namelist is used instead. Note that if `Dt_Constant` in the `NUMERICS` namelist is specified then its value is used regardless.

Type : `logical`

Default : `.false.`

Ignore_Joule_Heat

Description : If this flag is true, the Joule heat data in the restart file (if any) will be ignored when initializing the code. This variable is only relevant for restart calculations with `Electromagnetics` enabled in the `PHYSICS` namelist.

Type : `logical`

Default : `.false.`

Ignore_Solid_Mechanics

Description : If this flag is true, the solid mechanics data in the restart file (if any) will be ingored when initializing the code. This variable is only relevant for restart calculations with `Solid_Mechanics` enabled in the `PHYSICS` namelist.

Type : `logical`

Default : `.false.`

Temperature_Is_Primary

Description : If this flag is true, the initial enthalpy data will be calculated from the restart file temperature and volume fraction data, using the material and phase change properties specified in the input file. Otherwise, the restart file enthalpy data will be considered to be primary, and the initial temperature data will be calculated from it. This variable is only relevant for restart calculations.

Type : `logical`

Default : `.false.`

Chapter 26

`REGION` Namelist

Overview

The `REGION` namelist defines regions of the mesh and assigns attributes to those regions. The current implementation can only define rectangular regions through upper and lower bounds. A cell is contained in the region if its centroid lies between the upper and lower bounds in every direction. The only attribute that is implemented is fluid flow solution.

`REGION` Namelist Features

Required / Optional : Optional

Single / Multiple Instances : Multiple

Components

- `x1`
- `y1`
- `z1`
- `x2`
- `y2`

- `z2`
- `flow_off`

x1

Description : Lower bound of the region in the x direction.

Physical Dimension : L

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

y1

Description : Lower bound of the region in the y direction.

Physical Dimension : L

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

z1

Description : Lower bound of the region in the z direction.

Physical Dimension : L

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

x2

Description : Upper bound of the region in the x direction.

Physical Dimension : L

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

y2

Description : Upper bound of the region in the y direction.

Physical Dimension : L

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

z2

Description : Upper bound of the region in the z direction.

Physical Dimension : L

Type : `real`

Default : 0.0

Valid values : $(-\infty, \infty)$

flow_off

Description : Logical input describing the state of fluid flow in the region. (Set this variable to `.true.` to turn off flow in the region.)

Physical Dimension : (none)

Type : `logical`

Default : `.false.`

Valid values : `.true.`, `.false.`

Chapter 27

SENS_VARIABLE Namelist

Overview

The Sens_Variable namelists define variables for parametric studies and sensitivity analyses. It is emphasized that the Sens_Variable_Value overwrites the value specified given for the Sens_Variable_Type – Sens_Variable_Identifier – Sens_Variable_Component variable.

Note : The Sens_Variable_Pert, Convergence_Criterion (of the Energy_Nonlinear_Solution namelist), and Convergence_Criterion, Maximum_Iterations, Preconditioning_Method, Preconditioning_Preconditioner, Preconditioning_Scope, Preconditioning_Steps, Relaxation_Parameter, Stopping_Criterion (of the Energy_Sensitivity_Solution namelist), variables must be carefully chosen as explained in Appendix E of the *Truchas Physics and Algorithms*.

Note : Presently, only certain select variables in BC, MATERIAL and INDUCTION_COIL namelists can be identified for this purpose.

Sens_Variable Namelist Features

Required / Optional : Optional for parametric studies and sensitivity analyses.

Single / Multiple Instances : Multiple

Components

- `Sens_Variable_Active`
- `Sens_Variable_Component`
- `Sens_Variable_Identifier`
- `Sens_Variable_Iparam`
- `Sens_Variable_Lower`
- `Sens_Variable_Name`
- `Sens_Variable_Pert`
- `Sens_Variable_Type`
- `Sens_Variable_Upper`
- `Sens_Variable_Value`

`Sens_Variable_Active`

Description : (Optional) Switch to activate the use of the variable for parametric and sensitivity studies.

Physical Dimension : N/A

Type : `logical`

Default : `.true.`

Valid values : `.true.` or `.false.`

`Sens_Variable_Component`

Description : (Required) Indicate the namelist component that is effected.

Physical Dimension : N/A

Type : string

Default : (none)

Valid values :

Sens_Variable_Type	Valid values
'BC'	'BC_Value' 'T0_Convection'
'EM_SOURCE_SCALE'	N/A
'MATERIAL'	'Conductivity_Constants' 'Conductivity_Exponents' 'Cp_Constants' 'Cp_Exponents'

Sens_Variable_Identifier

Description : (Required) Indicate the name of the namelist that is effected.

Physical Dimension : N/A

Type : string

Default : (none)

Valid values :

Sens_Variable_Type	Valid values
'BC'	any BC_Name string
'EM_SOURCE_SCALE'	N/A[1]
'MATERIAL'	any Material_Name string

1. The Current value in the Sens_Variable_Iparam time interval is scaled by the square of the Sens_Variable_Value value for all coils. The sensitivity is computed with respect to this scaling factor.

Sens_Variable_Iparam

Description : (Optional) Indicate the variable that is effected, e.g. first or fifth.

Physical Dimension : N/A

Type : integer

Default : 1

Valid values :

Sens_Variable_Type	Sens_Variable_Component	Valid values
'BC'	'BC_Value'	1 – 23
'BC'	'T0_Convection'	N/A
'EM_SOURCE_SCALE'	N/A	1 – 32[1]
'Material'	'Conductivity_Constants'	1 – 10
'Material'	'Conductivity_Exponents'	1 – 10
'Material'	'Cp_Constants'	1 – 10
'Material'	'Cp_Exponents'	1 – 10

1. The `Current` value in the indicated time interval (cf. `Source_Times`) is scaled by the square of the `Sens_Variable_Value` value for all coils. The sensitivity is computed with respect to this scaling factor.

Sens_Variable_Lower

Description : (Optional) The lower bound of this variable.

Physical Dimension : N/A

Type : `real`

Default : (none)

Valid values : $(-\infty, \infty)$

Notes : Only used with optimization.

Sens_Variable_Name

Description : (Optional) The name of this `Sens_Variable`.

Physical Dimension : N/A

Type : string

Default : `SENS_VARIABLE_#` where # is the number of the design variable

Valid values : Any string

Sens_Variable_Pert

Description : (Required) The perturbation used for the sensitivity analysis.

Physical Dimension : N/A

Type : `real`

Default : (none)

Valid values : $(-\infty, \infty)$

Notes : This value should be large enough to prevent round-off error and small enough to prevent truncation error. A value ranging from 0.01% to 1.00% of the nominal `Sens_Variable_Value` value is recommended.

Sens_Variable_Type

Description : (Required) Indicate the type of the namelist that is effected.

Physical Dimension : N/A

Type : string

Default : (none)

Valid values : `'BC'`, `'MATERIAL'` or `'EM_SOURCE_SCALE'` [1]

Sens_Variable_Upper

Description : (Optional) The upper bound of this variable.

Physical Dimension : N/A

Type : `real`

Default : (none)

Valid values : $(-\infty, \infty)$

Notes : Only used with optimization.

Sens_Variable_Value

Description : (Optional) The value of this variable.

Physical Dimension : N/A

Type : `real`

Default : The parameter value specified for the `Sens_Variable_Type` – `Sens_Variable_Identifier` – `Sens_Variable_Component` variable. [2]

Valid values : $(-\infty, \infty)$

Notes : It is emphasized that the `Sens_Variable_Value` overwrites the value specified given for the `Sens_Variable_Type` – `Sens_Variable_Identifier` – `Sens_Variable_Component` variable. [3]

[1] The `Current` value in the `Sens_Variable_Iparam` time interval is scaled by the square of the `Sens_Variable_Value` value for all coils. The sensitivity is computed with respect to this scaling factor.

[2] The default value equals one if `Sens_Variable_Type` = `'EM_SOURCE_SCALE'`.

[3] If `Sens_Variable_Type` = `'EM_SOURCE_SCALE'` then the appropriate scaling takes place.

Chapter 28

SENS_FUNCTION Namelist

Overview

The SENS_FUNCTION namelist defines a response function for parametric studies and sensitivity analyses.

Note : Presently the response functions can only utilize the temperature, enthalpy, and solid volume fraction from the heat transfer analysis.

Sens_Function Namelist Features

Required / Optional : Optional for parametric studies and sensitivity analyses.

Single / Multiple Instances : Multiple

Components

- Sens_Function_Active
- Sens_Function_Iparameters
- Sens_Function_Name
- Sens_Function_Rparameters
- Sens_Function_Type

Sens_Function_Active

Description : (optional) Switch to activate the use of the response function for parametric and sensitivity studies.

Physical Dimension : N/A

Type : `logical`

Default : `.true.`

Valid values : `.true.` or `.false.`

Note : This is optional.

Sens_Function_Iparameters

Description : (optional) A list of 1–30 integer parameters that are used to describe the response function

Physical Dimension : N/A

Type : integer

Default : (none)

Valid values : any integer number

Notes : See `Sens_Function_Type` table for valid entries

Sens_Function_Name

Description : (optional) The name of this `Sens_Function`.

Physical Dimension : N/A

Type : string

Default : `SENS_FUNCTION_#` where # is the function number

Valid values : Any string

Sens_Function_Rparameters

Description : (optional) A list of 1–30 real parameters that are used to describe the response function

Physical Dimension : N/A

Type : real

Default : (none)

Valid values : any real number

Notes : See `Sens_Function_Type` table for valid entries

`Sens_Function_Type`

Description : (required) Indicate the type of response function.

Physical Dimension : N/A

Type : string

Default : (none)

Valid values : see table

Notes : Users may define there own response function in subroutine specify_sensitivity_function

`Sens_Function_Type`	**Description**	`Sens_Function_Iparameters`	`Sens_Function_Rparameters`
T	cell temperature	N/A	$x - y - z$ coordinates †
H	cell enthalpy	N/A	$x - y - z$ coordinates †
VOF	cell volume fraction	`Material_Number` ‡	$x - y - z$ coordinates †
sens_use_function	User defined function	function dependent	function dependent

† The desired response, e.g. temperature, will be obtain from the cell closest to the location provided by $x - y - z$ coordinates.

‡ The volume fraction will be obtained for the `MATERIAL` namelist with the indicated `Material_Number`.

Chapter 29

SO_SOLVER Namelist

Overview

The SO_SOLVER namelist sets parameters used in the solution of the linear system that occurs when the support operators discretization is used for heat transfer or fluid flow.

SO_SOLVER Namelist Features

Required / Optional : Optional

Single / Multiple Instances : Single

Components

- Convergence_Criterion
- Maximum_Iterations
- Name
- Output_Mode
- Stopping_Criterion

In the sections that follow we refer to the linear system being solved as $Ax = b$, where:

- A is the coefficient matrix,
- x is an estimate of the solution vector,
- b is the source vector,
- N is the number of unknowns,
- r is an estimate of the residual, $r = b - Ax$,
- x_0 is the initial guess for the solution vector, and
- r_0 is the initial estimate of the residual, $r_0 = b - Ax_0$.

Convergence_Criterion

Description : Value of error estimate used to determine when convergence has been reached. Meaning depends on `Stopping_Criterion`.

Physical Dimension : N/A

Type : `real`

Default : 10^{-8}

Valid values : $(0, 0.1)$

Maximum_Iterations

Description : Maximum number of iterations allowed.

Physical Dimension : N/A

Type : `integer`

Default : max(20, min(1000, $2 * \sqrt{N_{cells}}$))

Valid values : > 0

Name

Description : Arbitrary but unique string to identify a set of linear solver settings

Physical Dimension : N/A

Type : string

Default : `'default'`

Valid values : arbitrary string

Output_Mode

Description : Controls verbosity of linear solver.

Physical Dimension : N/A

Type : string

Default : `'none'`

Valid values :

- **`'none'`** - quiet
- **`'errors'`** - errors only
- **`'errors+warnings'`** - errors and warnings
- **`'warnings+errors'`** - same as above
- **`'summary'`** - errors, warnings, plus a one-line summary consisting of the iteration number, the norms of the calculated residual and error estimate, and the norms of the true residual and error estimate (if calculated) each time convergence is checked
- **`'iterates'`** - same as `'summary'`, plus the coefficient, preconditioner (if there is one), source, initial guess and converged solution, plus the current iterate and true residual (if computed) at each iteration
- **`'full'`** - same as `'iterates'`, plus more intermediate values computed during the course of each iteration (note that this can generate large amounts of output, and is primarily for debugging)

Stopping_Criterion

Description : Test used to estimate error and determine when convergence has been reached. Value is set by `Convergence_Criterion`.

Physical Dimension : N/A

Type : string

Default : `'||r||'`

Valid values :

- **`'||r||/||b||'`** - good for most situations.

`'||r||/(||A||*||x||+||b||)'` - NOT RECOMMENDED (requires $\|A\|$ (or some estimate), which is not available here)

`'||r||/||r0||'` - useful, but suffers from dependence on the initial guess.

`'||r||/||x||'` - useful, but is not dimensionless

`'||r||'` - note that this test is not scaled by some other characteristic "size" of the system

`'||x-xold||/||x||'` - NOT RECOMMENDED.

Bibliography

[1] V. Venkatakrishnan. On the accuracy of limiters and convergence to steady state solutions. Technical Report 93–0880, AIAA, 1993. Presented at the 31st Aerospace Sciences Meeting.

[2] T. J. Barth and D. C. Jesperson. The design and application of upwind schemes on unstructured meshes. Technical Report AIAA–89–0366, AIAA, 1989. Presented at the 27th Aerospace Sciences Meeting and Exhibit.

[3] B. van Leer. Towards the ultimate conservative difference scheme. III. Upstream-centered finite-difference schemes for ideal compressible flow. *Journal of Computational Physics*, 23:263–275, 1977.

[4] B. van Leer. Towards the ultimate conservative difference scheme. IV. A new approach to numerical convection. *Journal of Computational Physics*, 23:276–299, 1977.

[5] William J. Rider and Dana A. Knoll. Time step size selection for radiation diffusion calculations. *Journal of Computational Physics*, 152:790–795, 1999.

[6] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in Fortran*. Cambridge, 1986.

[7] W. J. Rider and D. B. Kothe. Reconstructing volume tracking. *Journal of Computational Physics*, 141:112–152, 1998.

[8] T. W. Clyne and W. Kurz. Solute redisribution during solidification with rapid solid state diffusion. *Metallurgical Transactions A*, 12:965–971, 1981.

Index

'2level', 90, 91
'ain', 88
'bcgstab', 88, 90
'cg', 88, 90
'concentration_polynomial', 101, 103, 105, 107, 109, 111, 113, 115, 117, 126, 128, 131
'constant', 101, 103, 104, 107, 109, 111, 113, 115, 117, 126, 128, 131
'diagonal', 89, 90
'elastic_only', 129
'fgmres', 87, 88, 90
'global', 91
'gmres', 87, 88, 90
'ilu0', 89, 90
'jacobi', 89–91
'local', 91
'lu', 89, 90
'nk', 143
'none', 88
'ssor', 89–91
'temperature_and_concentration_polynomial', 101, 103, 104, 107, 109, 111, 113, 115, 117, 126, 128, 131
'temperature_polynomial', 101, 103, 104, 107, 109, 111, 113, 115, 117, 126, 128, 131
'tfqmr', 88, 90
=, 191

a2_coeff, 183, 184
a3_coeff, 183, 185
Abs_Conc_Tol, 44, 47, 49
Abs_Enthalpy_Tol, 44, 45, 50
Abs_Temp_Tol, 44, 45, 50
Absolute_Zero, 16, 193, 194
advection_order_energy, 145, 147
advection_order_momentum, 145, 148
AIN_Max_Vectors, 139, 140
AIN_Preconditioner, 44, 46
AIN_Tol, 44, 46
AIN_Vec_Tol, 44, 47
AIN_Vector_Tolerance, 139, 140
Alittle, 145, 148
alphaCnK, 184, 186
ALTMESH, 9, 61, 63, 133
    Altmesh_Coordinate_Scale_Factor, 9, 10
    Altmesh_File, 9, 10
    Grid_Transfer_File, 9, 10
    Parallel, 9, 11
Altmesh_Coordinate_Scale_Factor, 9, 10
Altmesh_File, 9, 10
Axis, 27, 28

Back_Diffusion_Coefficient, 184, 186, 187
BC, 13, 95, 225
    BC_Name, 14, 15, 227
    BC_Type, 14, 15, 17, 18, 207
    BC_Use_Function, 14, 17
    BC_Value, 14, 16, 17, 25, 208
    BC_Variable, 14, 18, 19
    Bounding_Box, 14, 19
    Conic Equation, 13, 19–23
    Conic_Constant, 14, 19
    Conic_Relation, 14, 19, 20
    Conic_Tolerance, 13, 14, 20
    Conic_X, 14, 20

Conic_XX, 14, 21
Conic_XY, 14, 21
Conic_XZ, 14, 21
Conic_Y, 14, 21
Conic_YY, 14, 22
Conic_YZ, 14, 22
Conic_Z, 14, 22
Conic_ZZ, 14, 23
Inflow_Concentration, 14, 23
Inflow_Material, 14, 23
Inflow_Temperature, 14, 23
Mesh_Surface, 14, 24
Node_Disp_Coords, 14, 24
Surface_Materials, 14, 24
Surface_Name, 14, 19, 24, 25
T0_Convection, 14, 16, 25
BC_Name, 14, 15, 227
BC_Type, 14, 15, 17, 18, 53, 54, 57, 58, 207
BC_Use_Function, 14, 17
BC_Value, 14, 16, 17, 25, 53, 54, 57, 58, 208
BC_Variable, 14, 18, 19
Bi-CGSTAB, 88
BODY, 27, 51, 81, 95, 189
Axis, 27, 28
Concentration, 27, 28, 51
File_Name, 27, 29
Fill, 27, 29
Height, 27, 29
Length, 27, 30
Material_Number, 27, 30
Mesh_Material_Number, 28, 30
Radius, 28, 30
Rotation_Angle, 28, 31
Rotation_Pt, 28, 31
Rtheta_Tabular_Pt, 28, 31
RZ_Tabular_Pt, 28, 32
Surface_Name, 28, 32
Tabular_Type, 28, 32
Temperature, 28, 32, 51
TG_Axis, 28, 34
TG_Origin, 28, 33
TG_R_Bounds, 28, 35
TG_R_Constants, 28, 35
TG_R_Exponents, 28, 35
TG_Z_Bounds, 28, 34
TG_Z_Constants, 28, 34
TG_Z_Exponents, 28, 34
Translation_Pt, 28, 33
Velocity, 28, 33
Body_Force, 193, 194
Body_Force_Face_Method, 146, 148
Body_Force_Implicitness, 146, 149
Bounding_Box, 14, 19

cell_set_id_list, 71–73, 75
Center, 78
CG, 88, 92
CG_Stopping_Tolerance, 62, 66
chaparral_Amax, 207, 208, 211, 214
chaparral_iParam, 207, 208, 211, 212, 214
chaparral_rParam, 207, 208, 211, 213, 214
chaparral_symmetry, 208, 214
CHEMICAL_REACTION, 37
CR_Active, 38
CR_Cmax, 38, 39
CR_Ea1, 38, 40
CR_Ea2, 38, 40
CR_Htot, 38, 40
CR_ko1, 38, 39
CR_ko2, 38, 39
CR_m, 38, 39
CR_n, 38, 39
CR_Products_id, 38, 41
CR_Reactants_id, 38, 40
Comment, 208, 210
Concentration, 27, 28, 51
Conduction_Implicitness, 146, 149, 150, 161
Conductivity_Conc_Bounds, 97, 101
Conductivity_Constants, 97, 102
Conductivity_Exponents, 97, 102
Conductivity_Relation, 97, 102
Conductivity_Temp_Bounds, 97, 103
Conic Equation, 13, 19–23
Conic_Constant, 14, 19
Conic_Relation, 14, 19, 20

Conic_Tolerance, 13, 14, 20
Conic_X, 14, 20
Conic_XX, 14, 21
Conic_XY, 14, 21
Conic_XZ, 14, 21
Conic_Y, 14, 21
Conic_YY, 14, 22
Conic_YZ, 14, 22
Conic_Z, 14, 22
Conic_ZZ, 14, 23
Contact_Distance, 146, 150
Contact_Norm_Trac, 146, 151
Contact_Penalty, 146, 151
Convergence_Criterion, 85, 86, 88, 92, 139, 141, 225, 235–237
Coord, 133, 134
Coordinate_Scale_Factor, 134, 137, 204
Courant_Number, 146, 152
Cp_Conc_Bounds, 97, 103
Cp_Constants, 97, 104
Cp_Exponents, 97, 104
Cp_Relation, 97, 104
Cp_Temp_Bounds, 97, 105
CR_Active, 38
CR_Cmax, 38, 39
CR_Ea1, 38, 40
CR_Ea2, 38, 40
CR_Htot, 38, 40
CR_ko1, 38, 39
CR_ko2, 38, 39
CR_m, 38, 39
CR_n, 38, 39
CR_Products_id, 38, 41
CR_Reactants_id, 38, 40
CSF_Mollify_Passes, 146, 149
CSF_Normal, 193, 195
CSF_Tangential, 193, 195
CTE_Conc_Bounds, 97, 100
CTE_Constants, 97, 100
CTE_Exponents, 97, 100
CTE_Relation, 97, 101
CTE_Temp_Bounds, 97, 101
Current, 67, 78, 79, 227–229
Curvature_Model, 193, 195
Cutvof, 146, 152
Cycle_Max, 146, 153
Cycle_Number, 146, 153

Damper_Parameters, 139, 142–144
deltempnucl, 183, 185
deltempsigma, 184, 185
Density, 51, 97, 105
Density_Change_Coefficients_C, 97, 105
Density_Change_Coefficients_T, 97, 106
Density_Change_Exponents_C, 97, 106
Density_Change_Exponents_T, 97, 106
Density_Change_Relation, 97, 107
Density_Conc_Bounds, 97, 107
Density_Temp_Bounds, 97, 108
diagonal, 89, 90
DIFFUSION_SOLVER, 43
    Abs_Conc_Tol, 44, 47, 49
    Abs_Enthalpy_Tol, 44, 45, 50
    Abs_Temp_Tol, 44, 45, 50
    AIN_Preconditioner, 44, 46
    AIN_Tol, 44, 46
    AIN_Vec_Tol, 44, 47
    Max_AIN_Itr, 44, 47
    Max_AIN_Vec, 44, 47
    Max_Step_Tries, 44, 48
    PC_AMG_Cycles, 44, 46, 48
    PC_SSOR_Relax, 44, 46, 48
    PC_SSOR_Sweeps, 44, 46, 49
    Rel_Conc_Tol, 44, 49
    Rel_Enthalpy_Tol, 44, 45, 50
    Rel_Temp_Tol, 44, 46, 50
    System_Type, 44, 50, 196
    Verbose_Stepping, 44, 51
Diffusion_Solver, 43, 193, 196
Discrete_Ops_Type, 146, 153, 154, 160, 161
Displacement_Nonlinear_Solution, 146, 154
DS_SPECIES_BC, 51, 53
    BC_Type, 53, 54
    BC_Value, 53, 54
    Face_Set_IDs, 53, 54

Name, 53, 55
DS_TEMPERATURE_BC, 51, 57
BC_Type, 57, 58
BC_Value, 57, 58
Face_Set_IDs, 57, 58
Name, 57, 59
Dt_Constant, 43, 146, 155–157, 218
Dt_Grow, 43, 146, 155
Dt_Init, 146, 155, 156, 218
Dt_Max, 146, 155, 156
Dt_Min, 146, 155–157

ELECTROMAGNETICS, 61, 77, 79, 108, 145
CG_Stopping_Tolerance, 62, 66
EM_Domain_Type, 62, 63, 69
Epsilon_0, 61–63
Graphics_Output, 62, 64
Material_Change_Threshold, 62, 64
Maximum_CG_Iterations, 62, 65
Maximum_Source_Cycles, 62, 65
Mu_0, 61, 62, 65
Num_Etasq, 62, 66
Output_Level, 62, 66
Source_Frequency, 62, 67, 68
Source_Times, 62, 67, 69, 79, 228
SS_Stopping_Tolerance, 62, 65, 68
Steps_Per_Cycle, 62, 68
Symmetry_Axis, 62, 63, 69
Uniform_Source, 62, 67, 69
Electromagnetics, 194, 196, 218
EM_Conductivity_Conc_Bounds, 97, 108
EM_Conductivity_Constants, 97, 108
EM_Conductivity_Exponents, 97, 109
EM_Conductivity_Relation, 97, 109
EM_Conductivity_Temp_Bounds, 97, 109
EM_Domain_Type, 62, 63, 69
EM_Permeability_Conc_Bounds, 97, 110
EM_Permeability_Constants, 97, 110
EM_Permeability_Exponents, 98, 110
EM_Permeability_Relation, 98, 111
EM_Permeability_Temp_Bounds, 98, 111
EM_Permittivity_Conc_Bounds, 98, 112
EM_Permittivity_Constants, 98, 112
EM_Permittivity_Exponents, 98, 112
EM_Permittivity_Relation, 98, 113
EM_Permittivity_Temp_Bounds, 98, 113
Enclosure_Diagnostics, 175, 176
Enclosure_Flux_Convergence, 146, 157
Enclosure_Implicit_Solution, 146, 157
Enclosure_Max_Iterations, 146, 158
Enclosure_Use_Cell_Temps, 146, 158
EnclosureID, 16, 208
Energy_Nonlinear_Solution, 146, 158, 225
Energy_Sensitivity_Pseudo_Load, 146, 159
Energy_Sensitivity_Solution, 146, 159, 225
Epsilon_0, 61–63
Eutectic_Concentration, 184, 187
Eutectic_Temperature, 184, 187

Face_Set_IDs, 53, 54, 57, 58
FF_Discrete_Ops_Type, 146, 154, 159, 160
FGMRES, 86–88, 92
File_Name, 27, 29
Fill, 27, 29
flow_off, 222, 223
Fluid_Flow, 193, 196
Flux_Vol_Iter_Max, 146, 160
Forward_Diffusion_Coefficient, 184, 187
Fourier_Number, 146, 160
full_coupling_flag, 184, 186
Fuzz, 133, 135

Gap_Element_Blocks, 133, 135
Gauss-Seidel, 90
GMRES, 86–88, 92
Graphics_Output, 62, 64
Grid_Transfer_File, 9, 10

Heat_Conduction, 194, 196, 197
HEAT_SOURCE, 51, 71
cell_set_id_list, 71–73, 75
heat_source_center, 71–73, 75
heat_source_constant, 71, 72, 74
heat_source_radius, 71–75
heat_source_time, 71, 72, 74
heat_source_type, 71–73

travel_speed, 71, 72, 75
Volumetric_Heat_Source, 73
volumetric_heat_source, 71–73, 75
heat_source_center, 71–73, 75
heat_source_constant, 71, 72, 74
heat_source_radius, 71–75
heat_source_time, 71, 72, 74
heat_source_type, 71–73
Height, 27, 29
Heps, 133, 136
Hi_Temp_Phase_Material_ID, 184, 188
HT_Discrete_Ops_Type, 146, 154, 161

Ignore_Dt, 156, 217, 218
Ignore_Joule_Heat, 217, 218
Ignore_Solid_Mechanics, 217, 218
ILU, 89, 90
Immobile, 98, 113
INDUCTION_COIL, 61, 67, 69, 77, 225
Center, 78
Current, 67, 78, 79, 227–229
Length, 78, 79
NTurns, 78, 79
Radius, 78, 79
inFile, 208–210
inFileType, 208, 210
Inflow_Concentration, 14, 23
Inflow_Material, 14, 23
Inflow_Temperature, 14, 23
Int_Output_Dt_Multiplier, 175, 176
Int_Particles, 81, 82
Interface_Area, 146, 162
Interface_Geometry, 146, 162, 163
Interface_Smoothing_Length, 146, 163
Interface_Topology_Model, 147, 163
INTERFACES, 81
Int_Particles, 81, 82
Vof_Max_Recursion, 81, 83
Vof_Method, 81, 82
Vof_Particles, 81, 82
Vof_Tolerance, 81, 83
Inviscid, 194, 197

Jacobi, 89, 90

Krylov_Vectors, 85, 86, 88

Lame1_Conc_Bounds, 98, 114
Lame1_Constants, 98, 114
Lame1_Exponents, 98, 115
Lame1_Relation, 98, 115
Lame1_Temp_Bounds, 98, 115
Lame2_Conc_Bounds, 98, 116
Lame2_Constants, 98, 116
Lame2_Exponents, 98, 116
Lame2_Relation, 98, 117
Lame2_Temp_Bounds, 98, 117
Latent_Heat, 184, 188
Length, 27, 30, 78, 79
Limiter_Type, 147, 164
LINEAR_SOLVER, 85, 88, 93, 142, 145, 159, 167, 170
Convergence_Criterion, 85, 86, 88, 92, 225
Krylov_Vectors, 85, 86, 88
Maximum_Iterations, 85, 87, 88, 225
Method, 85, 87, 90
Name, 85, 87, 88, 142, 159, 167, 170
Output_Mode, 85, 89
Preconditioning_Method, 85, 89–91, 225
Preconditioning_Preconditioner, 86, 90, 91, 225
Preconditioning_Scope, 86, 91, 225
Preconditioning_Steps, 86, 90, 225
Relaxation_Parameter, 86, 91, 225
Status_Frequency, 86, 92
Stopping_Criterion, 86, 88, 92, 225, 236
Linear_Solver_Name, 88, 93, 139, 142
LinearSolver, 208, 211
Liquidus_Slope, 184, 188
Liquidus_Temp, 184, 189
Lo_Temp_Phase_Material_ID, 184, 189
Long_Edit_Bounding_Coords, 175, 176
Long_Output_Dt_Multiplier, 175, 177
LU, 89, 90

Mass_Limiter, 147, 164

Mass_Limiter_Cutoff, 147, 165
MATERIAL, 40, 41, 61, 95, 99, 127, 128, 225
Conductivity_Conc_Bounds, 97, 101
Conductivity_Constants, 97, 102
Conductivity_Exponents, 97, 102
Conductivity_Relation, 97, 102
Conductivity_Temp_Bounds, 97, 103
Cp_Conc_Bounds, 97, 103
Cp_Constants, 97, 104
Cp_Exponents, 97, 104
Cp_Relation, 97, 104
Cp_Temp_Bounds, 97, 105
CTE_Conc_Bounds, 97, 100
CTE_Constants, 97, 100
CTE_Exponents, 97, 100
CTE_Relation, 97, 101
CTE_Temp_Bounds, 97, 101
Density, 51, 97, 105
Density_Change_Coefficients_C, 97, 105
Density_Change_Coefficients_T, 97, 106
Density_Change_Exponents_C, 97, 106
Density_Change_Exponents_T, 97, 106
Density_Change_Relation, 97, 107
Density_Conc_Bounds, 97, 107
Density_Temp_Bounds, 97, 108
EM_Conductivity_Conc_Bounds, 97, 108
EM_Conductivity_Constants, 97, 108
EM_Conductivity_Exponents, 97, 109
EM_Conductivity_Relation, 97, 109
EM_Conductivity_Temp_Bounds, 97, 109
EM_Permeability_Conc_Bounds, 97, 110
EM_Permeability_Constants, 97, 110
EM_Permeability_Exponents, 98, 110
EM_Permeability_Relation, 98, 111
EM_Permeability_Temp_Bounds, 98, 111
EM_Permittivity_Conc_Bounds, 98, 112
EM_Permittivity_Constants, 98, 112
EM_Permittivity_Exponents, 98, 112
EM_Permittivity_Relation, 98, 113
EM_Permittivity_Temp_Bounds, 98, 113
Immobile, 98, 113
Lame1_Conc_Bounds, 98, 114
Lame1_Constants, 98, 114
Lame1_Exponents, 98, 115
Lame1_Relation, 98, 115
Lame1_Temp_Bounds, 98, 115
Lame2_Conc_Bounds, 98, 116
Lame2_Constants, 98, 116
Lame2_Exponents, 98, 116
Lame2_Relation, 98, 117
Lame2_Temp_Bounds, 98, 117
Material_Feature, 98, 118
Material_Name, 98, 118, 227
Material_Number, 40, 41, 99, 118
MTS_b, 98, 119
MTS_d, 98, 119
MTS_edot_0i, 98, 119
MTS_g_0i, 98, 120
MTS_k, 98, 120
MTS_mu_0, 98, 120
MTS_p_i, 98, 121
MTS_q_i, 98, 121
MTS_sig_a, 98, 121
MTS_sig_i, 98, 121
MTS_temp_0, 98, 122
Permeability_Constant, 99, 122
Priority, 99, 122
Pwr_Law_A, 99, 123
Pwr_Law_N, 99, 123
Pwr_Law_Q, 99, 123
Pwr_Law_R, 99, 124
Reference_Concentration, 99, 101, 103, 105, 107, 109, 111, 113, 115, 117, 124, 128, 131
Reference_Temperature, 99, 101, 103, 105, 107, 109, 111, 113, 115, 117, 124, 128, 131
Sound_Speed, 99, 125
SPC_Diffusivity_Constants, 99, 125
SPC_Diffusivity_Exponents, 99, 125
SPC_Diffusivity_Relation, 99, 126
Stress_Reference_Temperature, 99, 126
Surf10_Conc_Bounds, 99, 127
Surf10_Constants, 99, 127, 128
Surf10_Exponents, 99, 127, 128

Surf10_Material_Number, 99, 128
Surf10_Relation, 99, 127, 128
Surf10_Temp_Bounds, 99, 129
Transmits_Radiation, 99, 129
Viscoplastic_Model, 99, 129
Viscosity_Conc_Bounds, 99, 129
Viscosity_Constants, 99, 130
Viscosity_Exponents, 99, 130
Viscosity_Relation, 99, 130
Viscosity_Temp_Bounds, 99, 131
Void_Temperature, 99, 131
Material_Change_Threshold, 62, 64
Material_Feature, 98, 118
Material_Name, 98, 118, 227
Material_Number, 27, 30, 40, 41, 99, 118
Max_AIN_Itr, 44, 47
Max_AIN_Vec, 44, 47
Max_Step_Tries, 44, 48
Maximum_CG_Iterations, 62, 65
Maximum_Iterations, 85, 87, 88, 139, 143, 225, 235, 236
Maximum_Source_Cycles, 62, 65
Mechanical_Energy_Bound, 147, 165
Melting_Temperature, 184, 189
Melting_Temperature_Solvent, 184, 189
MESH, 133
Coord, 133, 134
Coordinate_Scale_Factor, 134, 137, 204
Fuzz, 133, 135
Gap_Element_Blocks, 133, 135
Heps, 133, 136
Mesh_File, 134, 136
Mesh_File_Format, 25, 134, 136
NCell, 133, 134
Partitions_Per_Process, 134, 137
Partitions_Total, 134, 137
Ratio, 133–135
Mesh_File, 134, 136
Mesh_File_Format, 25, 134, 136
Mesh_Material_Number, 28, 30
Mesh_Surface, 14, 24
Method, 85, 87, 88, 90, 139, 143, 208, 209
MinFaceFraction, 147, 166
Momentum_Solidify_Implicitness, 147, 166
MTS_b, 98, 119
MTS_d, 98, 119
MTS_edot_0i, 98, 119
MTS_g_0i, 98, 120
MTS_k, 98, 120
MTS_mu_0, 98, 120
MTS_p_i, 98, 121
MTS_q_i, 98, 121
MTS_sig_a, 98, 121
MTS_sig_i, 98, 121
MTS_temp_0, 98, 122
Mu_0, 61, 62, 65
multigrid, 90

Name, 53, 55, 57, 59, 85, 87, 88, 140, 142, 143, 154, 158, 159, 167, 170, 235, 236
NCell, 133, 134
ndmaxnucl, 184, 186
Node_Disp_Coords, 14, 24
NONLINEAR_SOLVER, 88, 139, 143, 145, 154, 158
AIN_Max_Vectors, 139, 140
AIN_Vector_Tolerance, 139, 140
Convergence_Criterion, 139, 141, 225
Damper_Parameters, 139, 142–144
Linear_Solver_Name, 88, 93, 139, 142
Maximum_Iterations, 139, 143
Method, 88, 139, 143
Name, 140, 143, 154, 158
Perturbation_Parameter, 140, 143, 144
Use_Damper, 142–144
NTurns, 78, 79
Num_Etasq, 62, 66
NUMERICS, 43, 85, 88, 139, 145, 155, 218
advection_order_energy, 145, 147
advection_order_momentum, 145, 148
Alittle, 145, 148
Body_Force_Face_Method, 146, 148
Body_Force_Implicitness, 146, 149
Conduction_Implicitness, 146, 149, 150, 161
Contact_Distance, 146, 150

Contact_Norm_Trac, 146, 151
Contact_Penalty, 146, 151
Courant_Number, 146, 152
CSF_Mollify_Passes, 146, 149
Cutvof, 146, 152
Cycle_Max, 146, 153
Cycle_Number, 146, 153
Discrete_Ops_Type, 146, 153, 154, 160, 161
Displacement_Nonlinear_Solution, 146, 154
Dt_Constant, 43, 146, 155–157, 218
Dt_Grow, 43, 146, 155
Dt_Init, 146, 155, 156, 218
Dt_Max, 146, 155, 156
Dt_Min, 146, 155–157
Enclosure_Flux_Convergence, 146, 157
Enclosure_Implicit_Solution, 146, 157
Enclosure_Max_Iterations, 146, 158
Enclosure_Use_Cell_Temps, 146, 158
Energy_Nonlinear_Solution, 146, 158, 225
Energy_Sensitivity_Pseudo_Load, 146, 159
Energy_Sensitivity_Solution, 146, 159, 225
FF_Discrete_Ops_Type, 146, 154, 159, 160
Flux_Vol_Iter_Max, 146, 160
Fourier_Number, 146, 160
HT_Discrete_Ops_Type, 146, 154, 161
Interface_Area, 146, 162
Interface_Geometry, 146, 162, 163
Interface_Smoothing_Length, 146, 163
Interface_Topology_Model, 147, 163
Limiter_Type, 147, 164
Mass_Limiter, 147, 164
Mass_Limiter_Cutoff, 147, 165
Mechanical_Energy_Bound, 147, 165
MinFaceFraction, 147, 166
Momentum_Solidify_Implicitness, 147, 166
Porous_Implicitness, 147, 166
Projection_Linear_Solution, 88, 89, 93, 147, 167
Sfront_Number, 147, 167, 168
Strain_Limit, 147, 168
Stress_Reduced_Integration, 147
Surften_Number, 147, 168
t, 147, 153, 169
Temp_Init_Iter_Max, 147, 169
Viscous_Implicitness, 147, 170
Viscous_Linear_Solution, 90, 93, 147, 170
Viscous_Number, 147, 170
Volume_Track_Brents_Method, 147, 171
Volume_Track_Interfaces, 147, 172
Volume_Track_Iter_Max, 147, 172
Volume_Track_Iter_Tol, 147, 172, 173
Volume_Track_Subcycles, 147, 173
optional, 3
outFile, 208, 210
outFileType, 208, 210
Output_Dt, 175, 177
Output_Level, 62, 66
Output_Mode, 85, 89, 235, 237
Output_T, 153, 175, 177
OUTPUTS, 153, 175
Enclosure_Diagnostics, 175, 176
Int_Output_Dt_Multiplier, 175, 176
Long_Edit_Bounding_Coords, 175, 176
Long_Output_Dt_Multiplier, 175, 177
Output_Dt, 175, 177
Output_T, 153, 175, 177
Precise_Output, 175, 178
Probe_Output_Cycle_Multiplier, 176, 178
Retain_Last_Step, 176, 178
Short_Output_Dt_Multiplier, 176, 178
User_Output_Dt_Multiplier, 176, 179
XML_Output_Dt_Multiplier, 176, 179
Parallel, 9, 11
PARALLEL_PARAMETERS, 181
Partitioner, 182
Processor_Array, 182

Partial, 208, 211, 213
Partition_Coefficient_Constants, 184, 190
Partition_Coefficient_Exponents, 184, 190
Partition_Coefficient_Relation, 184, 190
Partitioner, 182
Partitions_Per_Process, 134, 137
Partitions_Total, 134, 137
PC_AMG_Cycles, 44, 46, 48
PC_SSOR_Relax, 44, 46, 48
PC_SSOR_Sweeps, 44, 46, 49
Permeability_Constant, 99, 122
Perturbation_Parameter, 140, 143, 144
Phase_Change, 194, 197, 198
Phase_Change_Active, 184, 191, 197
Phase_Change_Model, 184, 186, 191, 192
PHASE_CHANGE_PROPERTIES, 95, 105, 183
    a2_coeff, 183, 184
    a3_coeff, 183, 185
    alphaCnK, 184, 186
    Back_Diffusion_Coefficient, 184, 186, 187
    deltempnucl, 183, 185
    deltempsigma, 184, 185
    Eutectic_Concentration, 184, 187
    Eutectic_Temperature, 184, 187
    Forward_Diffusion_Coefficient, 184, 187
    full_coupling_flag, 184, 186
    Hi_Temp_Phase_Material_ID, 184, 188
    Latent_Heat, 184, 188
    Liquidus_Slope, 184, 188
    Liquidus_Temp, 184, 189
    Lo_Temp_Phase_Material_ID, 184, 189
    Melting_Temperature, 184, 189
    Melting_Temperature_Solvent, 184, 189
    ndmaxnucl, 184, 186
    Partition_Coefficient_Constants, 184, 190
    Partition_Coefficient_Exponents, 184, 190
    Partition_Coefficient_Relation, 184, 190
    Phase_Change_Active, 184, 191, 197
    Phase_Change_Model, 184, 186, 191, 192
    Phase_Change_Type, 184–186, 189, 191
    Secondary_Half_Arm_Spacing, 184, 186, 192
    Solidification_Time, 184, 186, 192
Phase_Change_Type, 184–186, 189, 191
    =, 191
PHYSICS, 93, 193, 218
    Absolute_Zero, 16, 193, 194
    Body_Force, 193, 194
    CSF_Normal, 193, 195
    CSF_Tangential, 193, 195
    Curvature_Model, 193, 195
    Diffusion_Solver, 43, 193, 196
    Electromagnetics, 194, 196, 218
    Fluid_Flow, 193, 196
    Heat_Conduction, 194, 196, 197
    Inviscid, 194, 197
    Phase_Change, 194, 197, 198
    Porous_Flow, 194, 197
    Solid_Mechanics, 194, 196, 198, 218
    Solid_Mechanics_Phase_Change, 194, 198
    Stefan_Boltzmann, 16, 194, 198
    Stokes, 194, 199
    Surface_Tension_Model, 194, 199
    Turbulence_CMU, 194, 199
    Turbulence_KE_Fraction, 194, 200
    Turbulence_Length, 194, 200
    Turbulence_Model, 194, 200
    Void_Pressure, 194, 201
Porous_Flow, 194, 197
Porous_Implicitness, 147, 166
Precise_Output, 175, 178
Preconditioning_Method, 85, 89–91, 225
Preconditioning_Preconditioner, 86, 90, 91, 225
Preconditioning_Scope, 86, 91, 225
Preconditioning_Steps, 86, 90, 225
Priority, 99, 122
Probe, 203
    Probe_Coords, 203, 204

Probe_Coords_Scale, 203, 204
Probe_Description, 203, 204
Probe_Name, 203, 204
Probe_Coords, 203, 204
Probe_Coords_Scale, 203, 204
Probe_Description, 203, 204
Probe_Name, 203, 204
Probe_Output_Cycle_Multiplier, 176, 178
Processor_Array, 182
Projection_Linear_Solution, 88, 89, 93, 147, 167
Pwr_Law_A, 99, 123
Pwr_Law_N, 99, 123
Pwr_Law_Q, 99, 123
Pwr_Law_R, 99, 124

RADIATION_ENCLOSURE, 207
chaparral_Amax, 207, 208, 211, 214
chaparral_iParam, 207, 208, 211, 212, 214
chaparral_rParam, 207, 208, 211, 213, 214
chaparral_symmetry, 208, 214
Comment, 208, 210
EnclosureID, 16, 208
inFile, 208–210
inFileType, 208, 210
LinearSolver, 208, 211
Method, 208, 209
outFile, 208, 210
outFileType, 208, 210
Partial, 208, 211, 213
Radius, 28, 30, 78, 79
Ratio, 133–135
Reference_Concentration, 99, 101, 103, 105, 107, 109, 111, 113, 115, 117, 124, 128, 131
Reference_Temperature, 99, 101, 103, 105, 107, 109, 111, 113, 115, 117, 124, 128, 131
REGION, 221
flow_off, 222, 223
x1, 221, 222
x2, 221, 223
y1, 221, 222
y2, 221, 223
z1, 221, 222
z2, 222, 223
Rel_Conc_Tol, 44, 49
Rel_Enthalpy_Tol, 44, 45, 50
Rel_Temp_Tol, 44, 46, 50
Relaxation_Parameter, 86, 91, 225
RESTART, 156, 217
Ignore_Dt, 156, 217, 218
Ignore_Joule_Heat, 217, 218
Ignore_Solid_Mechanics, 217, 218
Temperature_Is_Primary, 217, 218
Retain_Last_Step, 176, 178
Rotation_Angle, 28, 31
Rotation_Pt, 28, 31
Rtheta_Tabular_Pt, 28, 31
RZ_Tabular_Pt, 28, 32

Secondary_Half_Arm_Spacing, 184, 186, 192
Sens_Function, 231
Sens_Function_Active, 231, 232
Sens_Function_Iparameters, 231, 232
Sens_Function_Name, 231, 232
Sens_Function_Rparameters, 231, 232
Sens_Function_Type, 231, 233
Sens_Function_Active, 231, 232
Sens_Function_Iparameters, 231, 232
Sens_Function_Name, 231, 232
Sens_Function_Rparameters, 231, 232
Sens_Function_Type, 231, 233
Sens_Variable, 225
Sens_Variable_Active, 226
Sens_Variable_Component, 225–227, 229
Sens_Variable_Identifier, 225–227, 229
Sens_Variable_Iparam, 226, 227, 229
Sens_Variable_Lower, 226, 228
Sens_Variable_Name, 226, 228
Sens_Variable_Pert, 225, 226, 228
Sens_Variable_Type, 225–227, 229
Sens_Variable_Upper, 226, 229
Sens_Variable_Value, 225–229
Sens_Variable_Active, 226
Sens_Variable_Component, 225–227, 229
Sens_Variable_Identifier, 225–227, 229
Sens_Variable_Iparam, 226, 227, 229

Sens_Variable_Lower, 226, 228
Sens_Variable_Name, 226, 228
Sens_Variable_Pert, 225, 226, 228
Sens_Variable_Type, 225–227, 229
Sens_Variable_Upper, 226, 229
Sens_Variable_Value, 225–229
Sfront_Number, 147, 167, 168
Short_Output_Dt_Multiplier, 176, 178
SO_SOLVER, 235
    Convergence_Criterion, 235–237
    Maximum_Iterations, 235, 236
    Name, 235, 236
    Output_Mode, 235, 237
    Stopping_Criterion, 235, 237
Solid_Mechanics, 194, 196, 198, 218
Solid_Mechanics_Phase_Change, 194, 198
Solidification_Time, 184, 186, 192
Sound_Speed, 99, 125
Source_Frequency, 62, 67, 68
Source_Times, 62, 67, 69, 79, 228
SPC_Diffusivity_Constants, 99, 125
SPC_Diffusivity_Exponents, 99, 125
SPC_Diffusivity_Relation, 99, 126
SS_Stopping_Tolerance, 62, 65, 68
Status_Frequency, 86, 92
Stefan_Boltzmann, 16, 194, 198
Steps_Per_Cycle, 62, 68
Stokes, 194, 199
Stopping_Criterion, 86, 88, 92, 225, 235–237
Strain_Limit, 147, 168
Stress_Reduced_Integration, 147
Stress_Reference_Temperature, 99, 126
Surf10_Conc_Bounds, 99, 127
Surf10_Constants, 99, 127, 128
Surf10_Exponents, 99, 127, 128
Surf10_Material_Number, 99, 128
Surf10_Relation, 99, 127, 128
Surf10_Temp_Bounds, 99, 129
Surface_Materials, 14, 24
Surface_Name, 14, 19, 24, 25, 28, 32
Surface_Tension_Model, 194, 199
Surften_Number, 147, 168
Symmetric Gauss-Seidel, 89
Symmetric Successive Over-Relaxation, 89, 90
Symmetry_Axis, 62, 63, 69
System_Type, 44, 50, 196

t, 147, 153, 169
T0_Convection, 14, 16, 25
Tabular_Type, 28, 32
Temp_Init_Iter_Max, 147, 169
Temperature, 28, 32, 51
Temperature_Is_Primary, 217, 218
TFQMR, 88
TG_Axis, 28, 34
TG_Origin, 28, 33
TG_R_Bounds, 28, 35
TG_R_Constants, 28, 35
TG_R_Exponents, 28, 35
TG_Z_Bounds, 28, 34
TG_Z_Constants, 28, 34
TG_Z_Exponents, 28, 34
Translation_Pt, 28, 33
Transmits_Radiation, 99, 129
travel_speed, 71, 72, 75
Turbulence_CMU, 194, 199
Turbulence_KE_Fraction, 194, 200
Turbulence_Length, 194, 200
Turbulence_Model, 194, 200

Uniform_Source, 62, 67, 69
Use_Damper, 142–144
User_Output_Dt_Multiplier, 176, 179

Velocity, 28, 33
Verbose_Stepping, 44, 51
Viscoplastic_Model, 99, 129
Viscosity_Conc_Bounds, 99, 129
Viscosity_Constants, 99, 130
Viscosity_Exponents, 99, 130
Viscosity_Relation, 99, 130
Viscosity_Temp_Bounds, 99, 131
Viscous_Implicitness, 147, 170
Viscous_Linear_Solution, 90, 93, 147, 170
Viscous_Number, 147, 170

Vof_Max_Recursion, 81, 83
Vof_Method, 81, 82
Vof_Particles, 81, 82
Vof_Tolerance, 81, 83
Void_Pressure, 194, 201
Void_Temperature, 99, 131
Volume_Track_Brents_Method, 147, 171
Volume_Track_Interfaces, 147, 172
Volume_Track_Iter_Max, 147, 172
Volume_Track_Iter_Tol, 147, 172, 173
Volume_Track_Subcycles, 147, 173
Volumetric_Heat_Source, 73
volumetric_heat_source, 71–73, 75

x1, 221, 222
x2, 221, 223
XML_Output_Dt_Multiplier, 176, 179

y1, 221, 222
y2, 221, 223

z1, 221, 222
z2, 222, 223