MERLIN User's Manual

Analyses of the RILEM TC-259 Benchmark Problems for
AAR Model Validation

Boulder, May 2016

BY

Victor Saouma
The University of Colorado
Boulder

INCLUDE LOGO HERE

DISCLAIMER OF WARRANTIES AND LIMITATION OF LIABILITIES

ORGANIZATION(S) THAT PREPARED THIS REPORT:

UNIVERSITY OF COLORADO AT BOULDER

ORDERING INFORMATION

For information about ordering this report, contact TOKYO ELECTRIC POWER SERVICE COMPANY.

ABSTRACT

MERLIN is a finite element program with capabilities for both two- and three-dimensional

- Static and dynamic stress analysis,
- Steady state and transient heat transfer analysis,
- Steady state and transient seepage flow analysis, and

MERLIN has been designed to perform nonlinear stress analyses and linear heat transfer and seepage flow analyses. MERLIN can be used to perform linear stress analyses, but does not support a load case capability, making automated superposition of stresses from different load cases or individual components of a load case impossible. Nonlinear stress analyses are supported through an incremental-iterative solution strategy.

Whereas MERLIN was originally written for the nonlinear fracture mechanics analysis of concrete dams, it has numerous features which make it an attractive program for other stress analysis (crack propagation simulation in particular).

Merlin-II is identical to the original Merlin, however it does support smeared cracks and its development was not supported by EPRI.

A separate manual contains numerous examples of data files for MERLIN.

This manual, including latest updates, is available on line at the following Web sites:
`http://civil.colorado.edu/~saouma/Merlin` and
`http://civil.colorado.edu/~saouma/Spider`

ACKNOWLEDGMENTS

MERLIN was developped by by Drs. Jan Červenka (**?**), and Ron Reich (**?**) at the University of Colorado under contract from EPRI. Prof. V. Saouma was the principal investigator.

Currently, Dr. Eric Hansen is working on Merlin.

The smeared crack model of MERLIN-II was implemented by Cervenka Consulting.

Contents

List of Figures

List of Tables

Chapter 1

PROGRAM CONTROL BLOCK

The program control block is located at the beginning of the MERLIN input file. It allows the user to select various program options that are in effect for the duration of an analysis. Program options that fit this description include the number of increments to perform, the analysis method (i.e. either the mixed-iterative or displacement method), what information is to appear in the output files, whether the analysis is to be restarted from an output file, or bandwidth optimization of the global stiffness matrix. Program control options are selected using keywords of the form described in Section **??** and can appear in any order within the program control block. The keywords associated with the options available in the program control block along with a brief description of their functionality are given in Tables 1.1. A section of this chapter is devoted to a more detailed description of each of these options.

Options	Functionality
General Options	
`Comment`	Insert a one-line comment in the input file
`Title`	Define a title for the analysis
`Restart`	Restart the program from an unformatted output file
`SetAARTime`	Used with `Restart` allows for continuity in stiffness degradation
`BandwidthMin`	Renumber nodes to obtain a narrower bandwidth for the global stiffness matrix
`MidSideNodes`	Manual definition of midside and interior nodes for higher order elements.
`UserCurves`	Enables the user to define a series of curves for G_F or dynamic uplift.
`Increments`	Specify the number of load or time increments to be performed
`InterfSepar`	Adds the interface initial separation into interface relative displacements in the first increment
`UpdateCord`	Update nodal coordinates with nodal displacements at each increment (for Buckling load)
`InitialTemp`	Explicitly define the stress free reference temperature
`LoadDspCurve`	Defines the nodal forces and displacements (absolute or relative) to monitor, at one location, for each increment.
`MultLDCurves`	Defines the nodal forces and displacements (absolute or relative)
`TimeDspCrv`	Defines nodal displacements (absolute or relative) in terms of time for transient analysis, or increment in static analysis.
`TimeAccelCrv`	Defines nodal acceleration (absolute or relative) in terms of time for transient analysis only. to monitor, at one or more location, for each increment.

Options	Functionality
TimeStrsCrv	Defines nodal stress in terms of time for transient analysis, or increment in static analysis.
TimeStrnCrv	Defines nodal strain in terms of time for transient analysis, or increment in static analysis.
PointStress	Prints on the standard output file (or on a separate one) stresses at user specified x-y-(z) locations.
MaxIncStress	Keeps track of the maximum stress anywhere inside the mesh.
RealTimeView	Enables Merlin to graphically display deformed shapes in real time
EndControl	Signal end of program control options
	Automatic Crack Propagation
AutoProp	Automatic crack propagation enabled
NoNucBox	Prevents nucleation of cracks inside one or more "boxes".
AutoNucleate	Automatic crack nucleation enabled
AutoBranch	Automatic crack branching enabled
	Dynamic Analysis
InitAnal	Initializes the boundary conditions for dynamic analyses
NumEigenVals	Specify the number of eigenvalues and eigenmodes to be determined for a given increment.
OrthoMass	Enables an orthotropic definition of added mass.
RayleighByGr	Enables different stiffness and mass damping coefficients for different groups.
TangDamping	Updated the Damping Matrix at each time increment in a nonlinear analysis.
	Analysis Type Options
HeatTransfer	Perform a heat transfer analysis
SeepageFlow	Perform a seepage flow analysis
TransAlpha	Perform a transient implicit stress analysis (Hughe's alpha method)
Transient	Perform a transient seepage or heat analysis
TransExplct	Perform a transient explicit stress analysis
DispMethod	Perform a displacement method analysis
Thermo-elast	Perform an uncoupled thermo-elastic analysis starting from an unformatted output file from a heat transfer analysis
Poro-elastic	Perform an uncoupled poro-elastic analysis starting from an unformatted output file from a seepage flow analysis
Contact	Perform a static contact analysis
Impact	Perform a dynamic contact/impact analysis
	Fracture Mechanics Options
LEFM	Perform linear elastic fracture mechanics analysis
J-integral	Use the J-integral to compute stress intensity factors
S-integral	Use the contour integral method of **?** to compute stress intensity factors
B-integral	Use the integral method of **?** to compute stress intensity factors
p_W0-COD_W0	Define piecewise linear function for COD_{W0} in terms of p_{W0}
NLFM	Perform nonlinear fracture mechanics analysis using the fictitious crack model
UFCMLW	Use user subroutine `ufcmlw` to define fictitious crack model softening law

Options	Functionality
	Strain Smoothing Options
`StrainSmooth`	Project strain to nodes using a consistent projection matrix with an iterative procedure
`C-splitting`	Project strain to nodes using a modified consistent projection matrix with an iterative procedure
	Error Estimation Options
`ErrAnalysis`	Compute error estimates.
	Output Options
`PrintAAR`	Print AAR induced strains
`PrintConVar`	Print constraint variables on interface to formatted output file
`PrintCrack`	Print the crack opening, pressure and stress profiles to formatted output file
`PrintDisp`	Print displacements to formatted output file
`PrintDynam`	Print nodal velocities and accelerations on a separate file
`PrintEigen`	Print the eigenvalues and eigenmodes to the formatted output file
`PrintErrEst`	Print error estimates to formatted output file
`PrintFlux`	Print fluxes to formatted output file
`PrintHead`	Print heads to formatted output file
`PrintPress`	Print heads in the form of pressures to the formatted output file
`PrintReact`	Print "reactions" to formatted output file
`PrintReinf`	Print embedded reinforcement information
`PrintResid`	Print residuals to formatted output file
`PrintStrain`	Print strains at nodes to formatted output file
`PrintStress`	Print stresses at nodes to formatted output file
`PrintIStrain`	Print strains at integration points to the formatted output
`PrintIStress`	Print stresses at integration points to the formatted output
`PrintState`	Print state variables to formatted output file
`PrintTemp`	Print temperatures to formatted output file
`Print_t_sif`	Print time versus mode I stress intensity factor for dynamic crack propagation to the formatted output file
`PrintSplit`	Print output of dynamic analysis in more than one file
`BreakPSTFile`	Breaks the graphical output `.pst` file into multiple files

Table 1.1: Program Control Options

1.1 General

1.1.1 Comment

The `Comment` option enables the user to enter a one-line comment in the input file. The text of the comment appears on the same line as the keyword, with the keyword and the text being separated by one or more blank spaces. The keyword and the comment text may extend beyond column 80 of the input file, but only those characters that appear in columns 1 through 80 will be echoed to the formatted output file. **Comment** options can be placed in the file back-to-back for multi-line comments.

1.1.2 Title

The `Title` option enables the user to define a title for the analysis. The line in the input file following the keyword contains the title. The title should be 80 characters or less; the program ignores any extra characters. The title is written into the post file and will appear in all plots produced by the post-processor.

1.1.3 Restart

The `Restart` option, followed by the increment N, enables the user to resume execution Merlin. N corresponds to the increment number which is to be used for restart. This means that the next step after restart will be $N + 1$.

Note that the first load increment inside Merlin is zero. Hence if a one increment analysis is first performed and then interrupted, then the restart should specify 0.

In a dynamic analysis, first a static analysis is performed (to account for initial loads), followed by a restart with a dynamic analysis. Note that the `Restart` should include all the loads previously defined in the static analysis (otherwise the analysis will behave as if a load was "unloaded" dynamically). Also keep in mind that in the dynamic analysis, loads are not incremental but total for each time step (contrarily to static analysis). Hence each time step should include all the loads.

The increment number must be a number greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. The unformatted output file must contain information for the increment prior to the increment at which restart has been specified, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. All incremental results are stored in one unformatted output file (see Section 4.2 for a description of the contents of this file). Since restart information is written to the unformatted output file for each successfully completed increment of analysis, the only instance in which the information for the previous increment would not be in the unformatted output file is if the increment had not yet been performed. Should an analysis be resumed at the increment that does not immediately follow the last successfully completed increment, the results in the unformatted output file for those increments which are being re-analyzed are overwritten. Failure to include the `Restart` option in the program control block indicates that execution is to begin with increment zero.

1.1.4 SetAARTime

`SetAARTime` sets the initial AAR time.

In a regular (non Restart) analysis, it allows the user to specify that AAR is starting at a certain time not equal to zero.

In restart analyses (static or dynamic), it allows the user to take into account the degradation (and expansion) caused in a previous analysis at the sepcified time. This keyword is to be followed by the AAR Time (printed in the output of the previous analysis). User must use this option if the previous AAR analysis entailed stiffness degradation (through the β_E coefficient. Hence, if an AAR analysis of a structure is performed, and there is concrete deterioration, and this is to be followed by a dynamic analysis, then this option **must** be used.

1.1.5 BandwidthMin

The `BandwidthMin` option enables the user to invoke minimization of the bandwidth of the global "stiffness" matrix and, in effect, to decrease the amount of time spent decomposing the global "stiffness" matrix and solving for the primary unknowns. Bandwidth minimization is optional because there will be instances when more computational effort is required to perform the minimization and solve the optimized system of equations than to simply solve the non-optimized system of equations. The bandwidth is minimized by internally renumbering the nodes using an algorithm developed by Gibbs, Poole, and Stockmeyer (**?**). For output purposes, the nodal numbering scheme defined in the input file is used, but internally the program uses the optimized nodal numbering scheme. Failure to include the `BandwidthMin` option in the program control block indicates that no attempt will be made to minimize the bandwidth of the global stiffness matrix.

1.1.6 MidSideNodes

The `MidSideNodes` option enables the user to manually define the node number and the location of the midside and interior nodes of higher order elements. If this option is not present in the control block of the input file, only the element corner nodes should be defined in the input file, and the program automatically generates the additional nodes and inserts them into the element connectivity.

Master/Slave nodes and boundary conditions are automatically generated too. If the option `MidSideNodes` is present in the input file, user must specify the node numbers and coordinates of all nodes and appropriately include them in the element connectivity. The element numbering of higher order elements is described in Section 2.2.2.

If, on the other hand, midside nodes are explicitly defined by the user, then all nodes must be defined, and the `MidSideNodes` option should be present.

1.1.7 UserCurves

The `UserCurves` option is used to define one, or more of the following:

1. Multilinear G_F curves in the interface element model, (Sect. 2.2.3.6.2).
2. Viscosity of the interface element model (Sect. 2.2.3.6.2).
3. Material properties (group 15 and 16) (Sect. 2.2.3.5.2 and 2.2.3.5.3).
4. Dynamic uplift (Sect. 3.4.1.8.6).

```
UserCurves
    NCurves
    Npoints
    x1   y1
    x2   y2
    ....
    Npoints
    x1    y1
```

```
    ....
EndControl
```

1.1.8 Increments

The `Increments` option enables the user to specify the number of load or time increments that are to be applied in a given analysis. For each increment an incremental load block must appear in the input file. The line in the input file following the keyword contains the number of increments. The number of increments must be a number greater than zero, otherwise the program will print an error message to the formatted output file and program execution will be terminated prematurely. Failure to include the `Increments` option in the program control block indicates that only one increment of load is to be applied.

1.1.9 InterfSepar

Will Add the interface initial separation (i.e. crack opening displacement based on nodal coordinates) to the interface relative displacements in the first increment.

This option is useful for reanalyses where COD obtained from a previous analysis must be factored in a new one (otherwise the interface element "recovers" its original undamaged strength).

1.1.10 UpdateCoord

This option updates the nodal coordinates with nodal displacements at each load increment. This simple feature enables the user to determine the **buckling** load in some cases.

1.1.11 InitialTemp

Nodal temperatures are incremental ones with respect the stress free reference temperature.

The `InitialTemp` command enables the user to specify a non-zero reference temperature, and hence this temperature will be added to the total nodal computed temperatures for AAR.

It is a convenient way to specify a uniform temperature for an AAR analysis.

`InitialTemp` has no effect on stress analysis.

1.1.12 Specific Nodal Results

1.1.12.1 LoadDspCurve

The `LoadDspCurve` option enables the user to define nodal forces and displacements that will be monitored in each increment and displayed in Spider. The monitored values are displayed in Spider as an $X-Y$ plot. The sum of specified displacements is shown on the horizontal axis and the sum of specified force components is on the vertical axis. To each increment there is a corresponding point in this $X-Y$ plot. The monitored values are defined in the following format:

Title (Optional) - A Label which will appear in the graphical post-processor to identify the current plot.

- N_f - number of forces to monitor[1].
  - $node_1 \quad DOF_1$
  - $node_2 \quad DOF_2$
  - .....
  - $node_{N_f} \quad DOF_{N_f}$
- N_u - number of displacements to monitor.
  - $node_1 \quad DOF_1 \quad \beta_1$
  - $node_2 \quad DOF_2 \quad \beta_2$
  - .....
  - $node_{N_u} \quad DOF_{N_u} \quad \beta_{N_u}$

where $node_x$ denotes the node number of node X, DOF_x is the number of the degrees of freedom at node X and β_x is the multiplication factor. In the resulting XY plot, for every increment there is one corresponding point with coordinates given by the following formulas:

$$Y = \sum_{i=1}^{N_f} F_{DOF_i}^{node_i} , \qquad X = \sum_{i=1}^{N_u} (u_{DOF_i}^{node_i} \beta_i) \tag{1.1}$$

where F_j^x is the j-th component of the force vector at node X and u_j^x is the j-th component of the displacement vector at node X. The multiplication factor β can be used for subtracting or averaging of the displacements.

1.1.12.2 MultLDCurves

This option is identical to the preceding one `LoadDspCurve`, except that the user can request the load displacement curve for multiple locations.

First line is the number of curves.

Subsequent lines are identical to the ones for the `LoadDspCurve` option. This block is repeated as many times as there are curves requested.

1.1.12.3 TimeDspCrv

Defines nodal displacements (absolute or relative) in terms of time for transient analysis, or increment in static analysis.

The first line following the keyword, is the number of curves. Subsequent lines should have the following format, for each curve: N_a - number of displacements to monitor.

- $node_1 \quad DOF_1 \quad \beta_1$

[1] Warning: If higher order elements (quadratic) are used, MERLIN will automatically insert new midside nodes. Thus, if the `LoadDspCurve` option is present, two analyses must be run. In the first one, the added nodes are listed, and prior to the second analysis, the user must add those nodes in the `LoadDspCurve`. This is particularly important for the force monitor.

- $node_2 \quad DOF_2 \quad \beta_2$
- .....
- $node_{N_a} \quad DOF_{N_a} \quad \beta_{N_a}$

where $node_x$ denotes the node number of node X, DOF_x is the number of the degrees of freedom at node X and β_x is the multiplication factor. In the resulting time versus displacement plot, for every time increment there is one corresponding point with displacement given by the following formulas:

$$Y = \sum_{i=1}^{N_u} (u_{DOF_i}^{node_i} \beta_i) \tag{1.2}$$

where u_j^x is the j-th component of the displacement vector at node X. The multiplication factor β can be used for subtracting or averaging of the displacements.

1.1.12.4 TimeAccelCrv

This option is identical to `MultLDCurves`, however it prints time versus acceleration for the specified degree of freedom.

The first line following the keyword, is the number of curves. Subsequent lines should have the following format, for each curve: N_a - number of acceleration to monitor.

- $node_1 \quad DOF_1 \quad \beta_1$
- $node_2 \quad DOF_2 \quad \beta_2$
- .....
- $node_{N_a} \quad DOF_{N_a} \quad \beta_{N_a}$

where $node_x$ denotes the node number of node X, DOF_x is the number of the degrees of freedom at node X and β_x is the multiplication factor. In the resulting time versus acceleration plot, for every time increment there is one corresponding point with acceleration given by the following formulas:

$$Y = \sum_{i=1}^{N_u} (a_{DOF_i}^{node_i} \beta_i) \tag{1.3}$$

where a_j^x is the j-th component of the acceleration vector at node X. The multiplication factor β can be used for subtracting or averaging of the accelerations.

1.1.12.5 TimeStrsCrv

This option generates the time versus stress (transient analysis) or increment versus stress (static analysis).

The first line following the keyword, is the number of curves. Subsequent lines should have the following format, for each curve: Node Number followed (next line) by an integer:

1	2	3	4	5	6	10	11	12
σ_{xx}	σ_{yy}	σ_{zz}	σ_{yz}	σ_{xz}	σ_{xy}	σ_{p1}	σ_{p2}	σ_{p3}

1.1.12.6 TimeStrnCrv

This option generates the time versus strain (transient analysis) or increment versus stress (static analysis).

The first line following the keyword, is the number of curves. Subsequent lines should have the following format, for each curve: Node Number followed (next line) by an integer:

1	2	3	4	5	6	10	11	12
σ_{xx}	σ_{yy}	σ_{zz}	σ_{yz}	σ_{xz}	σ_{xy}	σ_{p1}	σ_{p2}	σ_{p3}

1.1.12.7 PointStress

This option prints on the standard output file (or on a separate one) In a static analysis, stresses will be tabulated in terms of load increments, in a dynamic one in terms of time.

First line is a code indicating where those stresses are to be printed:
0: standard output file `fn.out`
1: separate output file `fn.str`

Next line is the number of locations, to be followed by the x-y-(z) coordinates (one per line).

In addition, these stresses are converted into a time-history and sent to Spider for graphical display in terms of Time (in transient analysis), or increments (in a static analysis).

1.1.13 MaxIncStress

When this keyword is specified, it must be followed (next line) by a single integer corresponding to the stress component, or the max/min principal stresses to be tracked during analysis.

1	2	3	4	5	6	7
σ_{xx}	σ_{yy}	σ_{zz}	σ_{yz}	σ_{xz}	σ_{xy}	σ_{max} & σ_{min}

At the end of each increment, Merlin will then print in the `.out` file coordinates, element No., and magnitudes of the stress(es) for the current increment and for the total analysis (up to this load increment).

1.1.14 RealTimeView

Enables the user to visualize the deformed shape in real time during analysis.

The keyword is followed (next line) by two integers Δ_{inc}, `flag`.

Δ_{inc} specifies the difference between two succesive increments data to be stored in the `.rtv` file. Hence, $\Delta_{inc} = 1$ means each increment, and $\Delta_{inc} = 2$ means every other increment is to be saved.

`flag` is either 0 or 1. If it is zero it implies that no acceleration data is to be written in the `.rtv` file. If it is set to 1, then the subsequent line should contain the number of nodes, followed by the nodes 9one per line) whose accelerations (all components) will be dumped.

It should be noted that Spider can also be invoked to visualize the full set of results at any time during analysis.

1.1.15 EndControl

The `EndControl` option is used to inform the program that all program control options required to perform the analysis have been specified. This keyword must appear at the end of the program control block. Failure to include an `EndControl` option will prohibit the program from recognizing the next keyword appearing in the input file, which will cause the program to print an error message to the formatted output file and terminate execution prematurely.

1.2 Automatic Crack Propagation

1.2.1 AutoProp

The `AutoProp` Option is used to enable automatic crack propagation (in conjunction with the `cracker` program). It requires one integer parameter only. The parameter corresponds to the number of propagation instances (based on tensile stress exceeding tensile strength, or effective stress intensity factor exceeding fracture toughness) to skip before finally propagating a crack (useful to delay propagation).

1.2.2 NoNucBox Box

Prevents nucleation of cracks inside one or more "boxes". Following the keyword, an integer specifies the number of boxes `nbox`, followed by `nbox` lines, each containing x_{min}, x_{max}, y_{min} and y_{max}. Inside these boxes Merlin will not check for crack nucleation.

1.2.3 AutoNucleate

This flag enables Merlin/Cracker to automatically nucleate a crack (2D only).

1.2.4 AutoBranch

This flag enables Merlin/Cracker to enable crack branching (2D only). This is achieved by scanning the stresses along an existing crack.

1.3 Analysis Type

1.3.1 DispMethod

The `DispMethod` option enables the user to specify that the (standard) displacement method will be used for the analysis. Currently, this option applies only to stress analyses since the mixed-iterative method has not been implemented for heat transfer and seepage flow analyses; heat transfer and seepage flow analyses default to the displacement method. Projection of the strains from the integration points to the nodes and computation of nodal stress values from the projected strains will still be performed, as with the mixed-iterative method. However, the nodal values of stress and strain will only be used for output purposes, in both the formatted and unformatted output files; no further finite element computations will be performed using

these nodal values. Failure to include the `DispMethod` option in the program control block indicates that the mixed-iterative method is to be used for the analysis.

1.3.2 HeatTransfer

The `HeatTransfer` option enables the user to specify that a heat transfer analysis is to be performed. Inclusion of this option in the control block effectively eliminates other options in the control block that are directly related to stress or seepage flow analyses. In addition, certain options in the incremental load block are disabled when this option is included in the control block; the incremental load options that are disabled based on the analysis type will be identified in Chapter 3. If the user does not specify a heat transfer or seepage flow analysis, a stress analysis will be performed.

1.3.3 SeepageFlow

The `SeepageFlow` option enables the user to specify that a seepage flow analysis is to be performed. Inclusion of this option in the control block effectively eliminates other options in the control block that are directly related to stress or heat transfer analyses. In addition, certain options in the incremental load block are disabled when this option is included in the control block; the incremental load options that are disabled based on the analysis type will be identified in Chapter 3. If the user does not specify a seepage flow or heat transfer analysis, a stress analysis will be performed.

1.3.4 Implicit Time History Analysis

Merlin differentiates between Stress analysis The `TransAlpha` and scalar analysis (heat and seepage) in which case the `Transient` option should be specified.

1.3.4.1 Stress Analysis; `TransAlpha`

Hughe's alpha method is implemented to perform time history stress analysis. By selecting $\alpha = 0$, Newmark's beta method (a special case of Hughe's alpha method) is used.

Through the proper selection of input data parameters, the user can select an implicit or explicit scheme. The following Rayleigh damping is assumed $\mathbf{C} = \alpha\mathbf{K} + \eta\mathbf{M}$ The line following the `Transient` option should include, Table 1.2 where

Δt	α	β	γ	C_1	C_2	`Flag`

Table 1.2: Transient Stress Analyses Input

- Δt Time step.
- α Hughe's parameter (must be negative to provide damping, $\alpha \leq -1/3$, recommended default value is -0.05.
- β Hughe's parameter

- γ Hughe's parameter
- C_1 Stiffness matrix damping coefficient
- C_2 Mass matrix damping coefficient
- `Flag` 0: Accelerations are explicitly prescribed for each increment; 1 Acceleration history prescribed only once in one block. The first option (0) should be used if forces, displacements, velocities or accelerations are incrementally specified. The second option (1) reduces the amount of input data (with respect to option 0) for seismic analyses.

Note that $\gamma = 1/2$ and $\beta = 1/4$ correspond to an average constant acceleration, and $\gamma = 1/2$ and $\beta = 1/6$ correspond to a linear variation of the acceleration during the time step. A constant acceleration is always stable, however for a linear acceleration to be stable

$$\frac{\Delta t}{T_n} \leq \frac{1}{\pi\sqrt{2}}\frac{1}{\sqrt{\gamma - 2\beta}} \tag{1.4-a}$$

$$\leq 0.551 \tag{1.4-b}$$

Note that:

1. Alpha introduces a damping that grows with the ratio of time increment to the period of vibration of a node.
2. Negative values of α provide damping.
3. If $\alpha = 0$, we have no artificial damping (energy preserving) and is exactly the trapezoidal rule (Newmark's method if $\beta = 1/4$ and $\gamma = 1/2$).
4. Maximum value is $\alpha = -1/3$ which provides the maximum artificial damping. This results in a damping ratio of about 6% when the time increment is 40% of the period of oscillation of the mode being studied and smaller if the oscillation period increases.
5. This artificial damping is not very substantial for realistic time increment and low frequencies, but is non-negligible for high frequencies.
6. A default value of -0.05 is recommended.
7.

Note that if the `RayleighByGr` option is specified (Sect. 1.3.4.1.4), then the values of the Rayleigh damping specified here will be ignored. Those coefficients will be assigned for each element group in Sect. 2.2.

1.3.4.1.1 InitAnal This option is to be specified only with transient stress analysis. It initializes the full boundary conditions (displacements, velocities and accelerations) for dynamic analyses, and is only needed if accelerations are not prescribed. Two of the boundary conditions must be know, and the third one will be accordingly determined.

Hence, if this option is not present, the initial value of the displacement, velocity and acceleration vectors is determined from the first (zero) increment. zero is assumed for undefined quantities.

1: Initial accelerations are determined from the initial displacements and velocities

$$\mathbf{a} = -\mathbf{M}^{-1}(\mathbf{Ku} + \mathbf{Cv}) \tag{1.5}$$

2: Initial velocities are determined from the initial displacements and accelerations

$$\mathbf{v} = -\mathbf{C}^{-1}(\mathbf{Ku} + \mathbf{Ma}) \tag{1.6}$$

Hence, this option enables MERLIN to perform an initial analysis to initialize accelerations and velocities.

This option is not needed if the dynamic analysis is restarted with the **Restart** option.

1.3.4.1.2 NumEigenVals The `NumEigenVals` option alerts Merlin that there will be subsequent commands requesting calculations of the lowest n eigenvalues and eigenvectors, where n is the single integer which follows this keyword on the next line.

Note that this keyword by itself does not trigger the calculation of the eigenvalues, this is achieved through the `EigenModes` under load option.

1.3.4.1.3 OrthoMass This option is to be followed (next line) by Table 1.3 where α_i (less than one) is the mass multiplier for global coordinate direction i. This enables the analysis to account for added masses when the earthquake is not necessarily along the reservoir-dam direction.

α_1	α_2	$[\alpha_3]$

Table 1.3: `OrthoMass` Option Input

1.3.4.1.4 RayleighByGr When this option is defined in the control block, MERLIN will later assume that in the mesh definition block Rayleigh stiffness and mass damping coefficients are defined for each group separately (see Sect. 2.2).

When this option is used, the Rayleigh coefficients defined in the `Transient` option (Sect. 1.3.4) are ignored.

Note: If a dynamic analysis will be performed through a `Restart` following a static one, and the `RayleighByGroup` option is desired, then this option must also appear in the static analysis. The Rayleigh properties entered in the static analysis will first be ignored in the static analysis, but will be activated in the dynamic one through the `Restart`.

1.3.4.1.5 TangDamping This option will update the damping matrix $\mathbf{C} = a\mathbf{K} + b\mathbf{M}$ at each time increment to reflect changes in the stiffness matrix $\mathbf{K}$.

If this command is used, or if `TangentStiff` is used in a given load increment, then damping is calculated based on the current stiffness. Merlin never uses an exact tangent stiffness, but it always calculates a secant/unloading stiffness and it uses this stiffness to speed up convergence as well as for the damping matrix.

1.3.4.2 Transient Scalar Problems `Transient`

The `Transient` option enables the user to specify that a transient (i.e., time dependent) analysis is to be performed using an implicit time intergration strategy (**?**). Transient analyses can be performed in conjunction with `HeatTransfer` and `SeepageFlow` options.

1.3.4.2.1 Heat Transfer and Seepage Flow Analyses The time integration strategy requires two floating point input parameters, Table 1.4

Δt	α	0.	0.	0.	0

Table 1.4: Transient Heat/Seepage Analyses Input

Δtt is the size of the time step, α is the fraction of the time step when the equations of motion are evaluated. α must be greater than or equal to 0.5 and less than or equal to 1.0, with 0.868 being the optimum value (**?**). If the user does not specify a transient analysis, a static or steady state analysis will be performed.

Note:

1. Transient seepage flow is currently not working.
2. The additional zeros are required for compatibility with the transient option in the dynamic analysis.

1.3.5 Explicit Stress Time History Analysis

Explicit stress time history analysis takes place if the `TransExplct` keyword is specified.

This keyword should be followed by

Δt	α	C_1	C_2	`Flag`

Table 1.5: Explicit Stress Time History Analysis

where

1. Δt is the initially defined time step.
2. α the coefficient for the critical time step evaluation $\Delta t_{crit} = \alpha \min L/W$ where L and W are "length" and "width" of the element. Note that Merlin will internally compute/update critical time steps. However, the user specified Δt is used to control the frequency with which results are printed in the input file. Loading for the substeps are calculated by interpolation.
3. C_1 Stiffness matrix damping coefficient
4. C_2 Mass matrix damping coefficient

5. `Flag` 0: Accelerations are explicitly prescribed for each increment; 1 Acceleration history prescribed only once in one block. The first option (0) should be used if forces, displacements, velocitites or accelerations are incrementally specified. The second option (1) reduces the amount of input data (with respect to option 0) for seismic analyses.

1.3.6 Thermo-elast

Currently, this option has not been implemented and, therefore, should not be used until this notice is removed.

The `Thermo-elast` option enables the user to perform uncoupled thermo-elastic analyses using the unformatted output file for a previously performed heat transfer analysis to define the nodal temperatures. The line in the input file following the keyword contains the increment number in the unformatted output file from which temperatures will be read; the datum temperature used to determine the temperature differentials at the nodes; and the file name of unformatted output file. If the file cannot be opened or the specified increment does not appear in the file an error message will be written to the formatted output file and execution will be terminated prematurely. The results of the uncoupled thermo-elastic analysis cannot be written to the unformatted output file containing the results of the heat transfer analysis. If the user attempts to specify the same name for both unformatted output files an error message will be written to the formatted output file and execution will be terminated prematurely. Failure to include the `Thermo-elast` option in the program control block indicates that standard stress analysis is to be performed.

Increment No.	Datum Temperature	File Name

Table 1.6: `Thermo-elast` Option Input

1.3.7 Poro-elastic

Currently, this option has not been implemented and, therefore, should not be used until this notice is removed.

The `Poro-elastic` option enables the user to perform uncoupled poro-elastic analyses using the unformatted output file for a previously performed seepage flow analysis to define the nodal pressures. The line in the input file following the keyword contains the increment number in the unformatted output file from which heads will be read; the datum head used to determine the pressures at the nodes; the unit weight of the fluid; and the file name of unformatted output file. If the file cannot be opened or the specified increment does not appear in the file an error message will be written to the formatted output file and execution will be terminated prematurely. The results of the uncoupled poro-elastic analysis cannot be written to the unformatted output file containing the results of the seepage flow analysis. If the user attempts to specify the same name for both unformatted output files an error message will be written to the formatted output file and execution will be terminated prematurely. Failure to include the `Poro-elastic` option in the program control block indicates that standard stress analysis is to be performed.

Increment No.	Datum Head	Fluid Unit Weight	File Name

Table 1.7: `Poro-elastic` Option Input

1.3.8 Contact

`Note this option is temporarily deactivated.`

The `Contact` option enables the user to specify that a static contact analysis is to be performed using the fictitious force concept and including the effects of friction between the nodes in contact (**?**, **?**). This option applies only to stress analysis. The line following the `Contact` option should include, Table 1.8 The contact parameter is a number less than 1, which controls

Contact parameter	Contact increments

Table 1.8: `Contact` Option Input

the accuracy of the solution. The fictitious contact forces are multiplied by this parameter to release the overlap of nodes during each contact increment. The contact increment indicates the number of iterations to be used in order to release the overlap between nodes.

1.3.9 Impact

`Note this option is temporarily deactivated.`

The `Impact` option enables the user to specify that a dynamic contact/impact analysis is to be performed using the fictitious force concept and including the effects of friction between the nodes in contact in contact (**?**, **?**). The line following the `Impact` option should include, Table 1.9 The impact parameter is a number less than 1, which controls the accuracy of the solution.

Impact parameter	Impact increments

Table 1.9: `Impact` Option Input

The fictitious contact forces are multiplied by this parameter to release the overlap of nodes during each impact increment. The impact increment indicates the number of iterations to be used in order to release the overlap between nodes.

1.4 Fracture Mechanics

1.4.1 LEFM

The `LEFM` option enables the user to specify that a linear elastic fracture mechanics analysis is to be performed. If the user does not specify which method to use for the computation of the stress intensity factors (see Sections 1.4.1.1, 1.4.1.2, and 1.4.1.3), the displacement correlation method will be used. If a heat transfer or seepage flow analysis is to be performed, the `LEFM` option will be ignored and a warning message will be printed to the formatted output file.

Failure to include a `LEFM` option indicates that a standard analysis is to be performed even if crack topology information is included in the mesh definition.

1.4.1.1 J-integral

The `J-integral` option enables the user to specify that the J integral (**?**, **?**, **?**) is to be used to indirectly compute the stress intensity factors for a linear elastic fracture mechanics analysis. The contour integral is evaluated to determine the energy associated with mode I and mode II; the stress intensity factors are computed from the energy quantities. This method is useful for problems of a homogenous medium with mode I dominant, but does not perform well when the crack is subject to an internal pressure. When mode II is dominant, stress intensity factor extraction for the computed energy values fails. This option is used in conjunction with the `LEFM` option, otherwise the `J-integral` option is ignored by the program. The program prints a warning message to the formatted output file when the `J-integral` option is being ignored. Failure to include a `J-integral` option indicates that another method will be used to compute the stress intensity factors, with the default being a displacement correlation procedure with singular, crack tip elements.

Theoretical basis of the J integral is described in Chapter **??**.

1.4.1.2 S-integral

The `S-integral` option enables the user to specify that the contour integral method of Stern (**?**, **?**) is to be used to directly compute the stress intensity factors for a linear elastic fracture mechanics analysis. This method is based on a reciprocal work theorem which allows for the direct computation of the stress intensity factors for modes I and II. This method is applicable to both homogeneous and bi-material mediums. For bi-material mediums the interface must be a planar surface (i.e., a straight line in two-dimensions) and the crack must be on the interface. This option is used in conjunction with the `LEFM` option, otherwise the `S-integral` option is ignored by the program. The program prints a warning message to the formatted output file when the `S-integral` option is being ignored. Failure to include a `S-integral` option in an LEFM analysis indicates that another method will be used to compute the stress intensity factors, with the default being displacement correlation procedure with singular, crack tip elements.

Theoretical basis of the S integral is described in Chapter **??**.

1.4.1.3 B-integral

The `B-integral` option enables the user to specify that the volume integral of **?** is to be used to directly compute the stress intensity factors for a linear elastic fracture mechanics analysis. This method is based on a generalized energy release approach. It is applicable only to problems of a homogeneous medium.

This option is used in conjunction with the `LEFM` option, otherwise the `B-integral` option is ignored by the program. The program prints a warning message to the formatted output file when the `B-integral` option is being ignored. The user must specify which extraction function is to be used and the radii of the discs over which the volume integration is to be performed. The line of input following the keyword contains the extraction method identifier, the number

of discs to consider, and the radii of the discs. There are three extraction functions available to the user:

1. The one proposed by **?**,
2. The one proposed by **?**, and
3. The one proposed by Jin.

If the specified extraction method identifier is not 1, 2, or 3 an error message is printed to the formatted output file and program execution is terminated prematurely. The number of discs must be greater than zero and less than six, otherwise an error message is printed to the formatted output file and program execution is terminated prematurely. If this method is used with quadrilateral elements, the elements near the crack tip should either square or rectangular in shape and the radii should be even multiples of the element size. Failure to include a `B-integral` option with an LEFM analysis indicates that another method will be used to compute the stress intensity factors, with the default being displacement correlation procedure with singular, crack tip elements.

Extraction Method	No. of Discs	r_1	$[r_2]$	$[r_3]$	$[r_4]$	$[r_5]$

Table 1.10: `B-integral` Option Input

1.4.1.4 V-integral

The `V-integral` option enables the user to specify that the equivalent domain integral method of **?** and **?** is to be used to compute the energy release rates and stress intensity factors for a 3D linear elastic fracture mechanics analysis. This method is based on a virtual crack extension method. It is applicable only to problems of a homogeneous medium. This option is used in conjunction with the `LEFM` option, otherwise the `V-integral` option is ignored by the program. The program prints a warning message to the formatted output file when the `V-integral` option is being ignored. The user must specify radii of the volume over which the volume integration is to be performed. The line of input following the keyword contains the radii of the volume and the number of volume and crack surface integration points. The radii should be such that there are always at least two elements between the volume boundary and the crack tip. The number of volume integration points is recommended to be 40 and the number of crack surface integration points is to be 6. Failure to include a `V-integral` option indicates that another method will be used to compute the stress intensity factors, with the default being displacement correlation procedure with singular, crack tip elements.

Volume Radii	40	6

Table 1.11: `V-integral` Option Input

Theoretical basis of the V integral is described in Chapter **??**.

1.4.2 p_W0-COD_W0

The **p_W0-COD_W0** option enables the user to specify COD_{W0} as a piecewise linear function of p_{W0}, where p_{W0} is the full hydrostatic pressure in a discrete crack and COD_{W0} is the COD below which full hydrostatic pressure no longer acts. This option is used in conjunction with the `LEFM` option when a discrete crack will be loaded with a hydrostatic pressure that is a function of the COD (see the `Uplift` option in Section 3.4.1.8.4). The line of input following the keyword contains the number of points defining the piecewise linear function. The number of points must be greater than zero and less than six, otherwise an error message is printed to the formatted output file and program execution is terminated prematurely. The coordinates for each point are defined on a separate line in the input file with the pressure being the first coordinate and the COD being the second coordinate. Both of these values should be positive, so the program uses their absolute values to eliminate any erroneous negative values. If the `p_W0-COD_W0` option is not included in the program control block and an `Uplift` option is included in the incremental load block the program will print an error message to the unformatted output file and program execution will be terminated prematurely.

1.4.3 NLFM

The `NLFM` option enables the user to specify that a nonlinear fracture mechanics analysis is to be performed using the fictitious crack model (**?**). An assumed crack path must be defined in the mesh by interface elements. The assumed crack path must extend completely through the mesh, effectively separating the mesh into sub-domains connected along the assumed crack path. Terminating the assumed crack path on the interior of the mesh will cause the computed surface tractions on the bonded interface surfaces to oscillate, which will cause the crack to propagate at load levels that are either too low or too high. Failure to include a `NLFM` option indicates that a standard stress analysis is to be performed and, consequently, prevents the user from using the NLFM material model and the automatic load scaling option (see Section 3.3.3.3) for NLFM analyses.

1.4.3.1 UFCMLW

The `UFCMLW` option enables the user to specify that the fictitious crack model softening law is defined using the user subroutine `ufcmlw`. The argument list for `ufcmlw` is describes in Section B.1. The program already includes the linear and bilinear softening laws; the `UFCMLW` option provides the user with the means to use more complex functions to define the softening law. This option is used in conjunction with the `FCM` option, otherwise the `UFCMLW` option is ignored by the program. The program prints a warning message to the formatted output file when the `UFCMLW` option is being ignored. Failure to include a `UFCMLW` option indicates that one of the softening laws provided in the program is to be used.

1.5 Nodal Strains

1.5.1 StrainSmooth

The `StrainSmooth` option enables the user to compute improved values for the projected nodal strains using an iterative technique. This alternate method of strain projection is an adap-

tation of an iterative stress smoothing technique proposed by **?**. It provides the capability to approximate projected strain values computed with a consistent projection matrix without the added cost of inverting the consistent projection matrix. Since the projected nodal strains computed with a consistent projection matrix are generally better than those computed with a lumped projection matrix, which is the standard projection matrix for the mixed-iterative method, it is possible to improve the projected strains without a significant increase in computational effort. The line of input following the keyword contains the maximum number of iterations to perform and the relative tolerance used to determine convergence of the iterative procedure. The maximum number of iterations must be greater than zero or the `StrainSmooth` option will be ignored and the program will print a warning message to the formatted output file. Convergence is reached when the ratio of the Euclidean norms of the nodal strain corrections and the incremental nodal strains is less than the relative tolerance. Typically, five iterations with a tolerance of one per cent (i.e., a value of 0.01) are sufficient for convergence. If convergence is not reached within five iterations, the mesh used for the analysis probably requires some local refinements to capture high strain gradients (i.e., stress concentrations). Areas where high strain gradients occur can be determined by examining the residual loads; the largest residual loads are found where the computed stresses are the least accurate. Failure to include the `StrainSmooth` option indicates that nodal strain projection is to be performed using a lumped projection matrix.

No. of Iterations	Convergence Tolerance

Table 1.12: `StrainSmooth` Option Input

1.5.2 C-splitting

The `C-splitting` option enables the user to compute improved values for the projected nodal strains using an iterative technique. The technique used here is identical to that used for the `StrainSmooth` option except that the lumped projection matrix has been subtracted from the consistent projection matrix used in the iterative procedure. This modification of the consistent projection matrix decreases its spectral radius, resulting in slightly improved convergence characteristics over the strain-smoothing technique. The line of input following the keyword contains the maximum number of iterations to perform and the relative tolerance used to determine convergence of the iterative procedure. The maximum number of iterations must be greater than zero or the `C-splitting` option will be ignored and the program will print a warning message to the formatted output file. Convergence is reached when the ratio of the Euclidean norms of the nodal strain corrections and the incremental nodal strains is less than the relative tolerance. Typically, five iterations with a tolerance of one per cent (i.e., a value of 0.01) are sufficient for convergence. If convergence is not reached within five iterations, the mesh used for the analysis probably requires some local refinements to capture high strain gradients (i.e., stress concentrations). Areas where high strain gradients occur can be determined by examining the residual loads; the largest residual loads are found where the computed stresses are the least accurate. Failure to include the `C-splitting` option indicates that nodal strain projection is to be performed using a lumped projection matrix.

No. of Iterations	Convergence Tolerance

Table 1.13: `C-splitting` Option Input

1.6 Error Analysis

When the `ErrAnalysis` option is selected in the control block of the input file, the program performs an error analysis. The error analysis consists of computing the error estimates on the element by element level. In addition, the optimal element size is estimated for each element. The algorithms for the error analysis are described in more detail in (**?**). The results of the error analysis can be printed to the formatted output file or they can be viewed using Spider. It is also possible to take into account the estimated optimal element size and automatically regenerate the finite element mesh using KumoNoSu. This kind of adaptive analysis is described within KumoNoSu.

One argument is required for the `ErrAnalysis` option. The line of input following the keyword must contain the required amount of relative error. It should be a number between zero and one in the most cases. This information will be used for the computation of the optimal element sizes. Note that 0.1 is an "adequate" starting point (corresponding to 10% error). The value should depend on the element type, and usually the error is grossly overestimated for linear elements (resulting in very fine mesh if this option is used in conjunction with KumoNoSu).

1.7 Output

1.7.1 PrintAAR

The `PrintAAR` option enables the user to specify that AAR induced strains are to be printed to the formatted output file.

1.7.2 PrintConVar

The `PrintConVar` option enables the user to specify that the nodal constraint variables for a stress analysis be printed to the formatted output file. Constraint variables are surface tractions on the bonded interface surface. If a heat transfer or seepage flow analysis is to be performed, the `PrintConVar` option will be ignored and a warning message reflecting this will be printed to the formatted output file. If the `PrintConVar` option is included in the control block and a stress analysis without a bonded interface is being performed, the `PrintConVar` option will be ignored. If the `PrintConVar` option is not included in the control block and a stress analysis with a bonded interface is being performed, the nodal constraint variables will not be printed to the formatted output file.

1.7.3 PrintCrack

The `PrintCrack` option enables the user to specify that the crack displacement, pressure and stress profiles for fracture mechanics analysis be printed to the formatted output file. The appropriate crack profiles are selected automatically depending on the type of analysis. For

example, if the `LEFM` option is selected, only the crack displacement profiles for each crack are printed to the output file. If in addition the `Uplift` option is applied in the cracks, the pressure profile is printed for every crack, and if the `NLFM` option is used, the stress profiles are printed too. If the `PrintCrack` option is not included in the control block of the input file, no crack profile information will be printed to the formatted output file.

This option will also print for each crack:

- Average friction angle
- Total cohesive force F_c
- Total normal force F_n
- Total shear force F_s
- Total area A
- Global safety factor against sliding determine as $SFF = \frac{F_c + \Sigma F_n \tan \Phi}{\Sigma F_h}$
- $x - y - z$ coordinate of the resultant forces
- average safety factor (from those determined at each Gauss point)

Furthermore, to examine the crack sliding and opening in 3d, use the `PrintStrain` option, where the first two strains of the interface element (ICM) correspond to sliding, and the third one to opening.

1.7.4 PrintDisp

The `PrintDisp` option enables the user to specify that the nodal displacements for a stress analysis be printed to the formatted output file.

1.7.5 PrintDynam

The `PrintDynam` option enables the user to specify that the nodal accelerations and velocities (in this order) will be be printed to a separate output file (`fn-dyn.out` where the Merlin input file is `fn.inp`). This option is automatically activated by Kumonosu when a free field analysis is being performed.

1.7.6 PrintEigen

The `PrintEigen` option enables the user to specify that the eigenvalues are to be printed to the formatted output file.

1.7.7 PrintErrEst

The `PrintErrEst` option enables the user to specify that error estimates for the strain energy and strains computed in a stress analysis using the option `ErrAnalysis`, sect. 1.6, be printed to the formatted output file. The `ErrAnalysis` option must be present in the input file otherwise the `PrintErrEst` option will be ignored. The error estimates will be reported for each element and for the entire mesh.

1.7.8 PrintFlux

The `PrintFlux` option enables the user to specify that the nodal fluxes for either a heat transfer or seepage flow analysis be printed to the formatted output file. If a stress analysis is to be performed, the `PrintFlux` option will be ignored and a warning message reflecting this will be printed to the formatted output file. If the `PrintFlux` option is not included in the control block and a heat transfer or seepage flow analysis is being performed, the nodal fluxes will not be printed to the formatted output file.

1.7.9 PrintHead

The `PrintHead` option enables the user to specify that the nodal heads for a seepage flow analysis be printed to the formatted output file. If a stress or heat transfer analysis is to be performed, the `PrintHead` option will be ignored and a warning message reflecting this will be printed to the formatted output file. If the `PrintHead` option is not included in the control block and a seepage flow analysis is being performed, the nodal heads will not be printed to the formatted output file.

1.7.10 PrintPress

The `PrintPress` option enables the user to specify that the nodal heads in the form of pressures for seepage flow analysis be printed to the formatted output file. This option makes it easier to include the results from the seepage flow analysis into a subsequent stress analysis. This option requires one argument. The line of input following the keyword must contain the value of weight density for the fluid. This value will be used to multiply the nodal heads to obtain the nodal pressures. If a stress analysis is to be performed, the `PrintPress` option will be ignored. If the `PrintPress` option is not included in the control block and a heat transfer or seepage flow analysis is being performed, the nodal pressures will not be printed to the formatted output file.

1.7.11 PrintReact

The `PrintReact` option enables the user to specify that the nodal "reactions" for a stress, heat transfer, or seepage flow analysis be printed to the formatted output file. If the `PrintReact` option is not included in the control block, the nodal reactions will not be printed to the formatted output file.

Note: In Merlin, the reaction vector includes **both** the reactions and the applied forces.

1.7.12 PrintReinf

The `PrintReinf` option enables the user to specify that the embedded reinforcement information is to be printed to the formatted output file. Embedded reinforcement provides the user with a simple procedure to specify reinforcement inside a mesh by simply indicating the coordinates of the end nodes. If the `PrintReinf` option is not included in the control block, the nodal reactions will not be printed to the formatted output file.

1.7.13 PrintResid

The `PrintResid` option enables the user to specify that the nodal residuals for a stress, heat transfer, or seepage flow analysis be printed to the formatted output file. If the `PrintResid` option is not included in the control block, the nodal residuals will not be printed to the formatted output file.

1.7.14 PrintStrain

The `PrintStrain` option enables the user to specify that the nodal strains for a stress analysis be printed to the formatted output file. Strains are expressed in terms of the global coordinate system. Principal strains and their directions are not computed by MERLIN and, therefore, are not printed to the formatted output file. If a heat transfer or seepage flow analysis is to be performed, the `PrintStrain` option will be ignored and a warning message reflecting this will be printed to the formatted output file. If the `PrintStrain` option is not included in the control block and a stress analysis is being performed, the nodal strains will not be printed to the formatted output file.

1.7.15 PrintStress

The `PrintStress` option enables the user to specify that the nodal stresses for a stress analysis be printed to the formatted output file. Stresses are expressed in terms of the global coordinate system. Principal strains and their directions are not computed by MERLIN and, therefore, are not printed to the formatted output file. If a heat transfer or seepage flow analysis is to be performed, the `PrintStress` option will be ignored and a warning message reflecting this will be printed to the formatted output file. If the `PrintStress` option is not included in the control block and a stress analysis is being performed, the nodal stresses will not be printed to the formatted output file.

1.7.16 PrintIStrain

The `PrintIStrain` option enables the user to specify that the strains at the integration points (Gauss) are to be printed on the formatted output file.

Note that by default, Merlin prints nodal strains, which are projections of the Gauss strains, and usually less accurate.

1.7.17 PrintIStress

The `PrintIStress` option enables the user to specify that the stresses at the integration points (Gauss) are to be printed on the formatted output file.

Note that by default, Merlin prints nodal stresses, which are projections of the Gauss stresses, and usually less accurate.

1.7.18 PrintState

The `PrintState` option enables the user to specify that the nodal state variables for a stress analysis be printed to the formatted output file. If a heat transfer or seepage flow analysis is to be performed, the `PrintState` option will be ignored and a warning message reflecting this will be printed to the formatted output file. If the `PrintState` option is not included in the control block and a stress analysis is being performed, the nodal state variables will not be printed to the formatted output file.

1.7.19 PrintTemp

The `PrintTemp` option enables the user to specify that the nodal temperatures for a heat transfer analysis be printed to the formatted output file.

In a stress analysis, this option will print the nodal temperatures computed from an external file (when the tt PointTemps is used, Section 3.4.1.14.2) to be printed.

1.7.20 Print_t_sif

The `Print_t_sif` option enables the user to specify that the mode I stress intensity factors should be printed for each time increment for a dynamic crack propagation analysis to the formatted output file. This option is used in conjunction with the `Transient, Impact and LEFM` option, otherwise the `Print_t_sif` option will be ignored by the program. The line following the `Print_t_sif` option should include, Table 1.14 where C, m, n and p are constants

C	m	n	p

Table 1.14: `Print_t_sif` Option Input

in the model developed by Plizzari et al. (**?**) for the dynamic crack propagation. The optimized value of constant C are 1.48 $\times 10^{-5}$ inch/cycle for small specimens and 4.94 $\times 10^{-5}$ inch/cycle for large specimens. The optimized values of m, n and p are 2.0, 1.1 and 0.7, respectively.

1.7.21 PrintSplit

The `PrintSPlit` commands, followed (next line) by an integer n, will split the formatted output into many files, each one containing results of n increments. Output files will have the form `fnx-y.out` where `x` and `y` are the first and last increments in the corresponding file. The first part of the output file (echo of input data) remains in `fn.out`.

This is a particularly useful feature for dynamic or nonlinear analysis with many increments.

1.7.22 BreakPSTFile

The `BreakPSTFile` commands, followed (next line) by an integer n, will split the graphical output file `.pst` into multiple files, each one containing results of n increments.

Output `.pst` files will have the form `pst_file_name_XXX.pst` where `XXX` corresponds to the first increment in the `.pst` file.

Chapter 2

MESH DEFINITION BLOCK

The mesh definition block immediately follows the program control block in the MERLIN input file. It enables the user to define the topology, geometry, and attributes of the finite element mesh. Mesh definition options are selected using keywords of the form described in Section **??** and can appear in any order within the mesh definition block. However, less space is required in program memory if the element group data is defined first. The keywords associated with the options available in the mesh definition block along with a brief description of their functionality are given in Table 2.1. A section of this chapter is devoted to a more detailed description of each of these options.

Options	Functionality
General Options	
`Comment`	Insert a one-line comment in the input file
`ElementGroup`	Define element group data
`AARProp`	Define material properties associated with an AAR simulation
`EndMesh`	Signal end of mesh definition options
Mesh Definition Options	
`Connectivity`	Define element connectivity
`Coordinates`	Define nodal coordinates
`Master/Slave`	Define duplicate node pairs
`ReinfRods`	Define steel reinforcement
`NodalMass`	Defines lumped nodal masses
`NodalDamping`	Defines nodal damper boundary condition
`NodalSprings`	Defines nodal spring boundary condition
Crack Definition Options	
`CrackSurface`	Define topology of crack surfaces
`CrackFronts`	Define crack front nodes
`ContourPath`	Define paths for contour integration through mesh

Table 2.1: Mesh Definition Options

2.1 Comment

The `Comment` option enables the user to enter a one-line comment in the input file. The text of the comment appears on the same line as the keyword, with the keyword and the text being separated by one or more blank spaces. The keyword and the comment text may extend beyond column 80 of the input file, but only those characters that appear in columns 1 through 80 will be echoed to the formatted output file. `Comment` options can be placed in the file back-to-back for multi-line comments.

2.2 ElementGroup

The `ElementGroup` option enables the user to define the element groups used in the entire analysis (when defined inside the `EndMesh`.

The user can also alter the material properties during a load increment. From that point onward, the previously defined properties are superseded by the newly defined ones.

2.2.1 Context

2.2.1.1 Default

Following `EndControl`, `ElementGroup` enables the user to define the element type as well as the constitutive model.

The line in the input file following the keyword contains the number of element groups. The number of element groups must be greater than zero, otherwise the program will print an error message to the formatted output file and program execution will be terminated prematurely. For each element group, two lines are required. The first line includes the following three pieces of information:

1. Element type (see Section 2.2.2),
2. The material model identifier, and
3. The material identifier for bi-material LEFM analysis.
4. If the `RayleighByGroup` option was specified[1] (Sect. 1.3.4.1.4), then insert the Rayleigh coefficients for the current group α_i and η_i.

The second line contains the material properties. MERLIN currently supports fourteen material models, which are listed in Table 2.2. The properties for the material models are described in Sections 2.2.3.2.1 through 2.2.3.7.6. Failure to include an `ElementGroup` option is a fatal input error, which will cause the program to print an error message to the formatted output file and terminate execution prematurely.

The element type must be greater than 0 and less than 35 (i.e., acceptable values are 1 through 34), otherwise the program will print an error message to the formatted output file and execution

[1]Note: If a dynamic analysis will be performed through a `Restart` following a static one, and the `RayleighByGroup` will be activated, then this option must also appear in the static analysis, and the Rayleigh properties entered in the static analysis (they will be ignored in the static analysis, and activated in the dynamic one through the `Restart`.

Material Model	Description	Incremental Formulation	Recommended Solution Strategy
	Linear Elastic		
1	Isotropic, linear elastic	Yes	None
2	2-D Transversely isotropic, linear elastic	Yes	None
3	3-D Orthotropic, linear elastic	Yes	None
	Nearly Incompressible		
20	Nearly incompressible	-	-
	Plasticity		
4	J_2 plasticity with radial return	Yes	SecantNewton
5	Drucker-Prager plasticity	Yes	SecantNewton
6	Coloumb friction	Yes	TangentStiff, LineSearch
	Smeared Cracks		
15	Fracture Plastic; Time dependant properties	Yes	TangentStiff, LineSearch
16	Fracture Plastic; User Curve	Yes	TangentStiff, LineSearch
17	Deactivated	-	-
18	Confinement Sensitive Fracture-Plastic	-	-
19	Fracture Plastic Model	Yes	TangentStiff, LineSearch
26	Kawamoto/Rock	Yes	-
	Discrete Cracks		
7	Fictitious Crack Model	No	TangentStiff, LineSearch
8	Interface/Joint Crack Model 1	Yes	TangentStiff, LineSearch
21	Interface/Joint Crack Model 2 (Cyclic)	Yes	TangentStiff, LineSearch
22	Interface/Joint Crack Model 3 (Concrete-Concrete)	Yes	TangentStiff, LineSearch
23	Interface/Joint Crack Model 4 (Rock-Concrete)	Yes	TangentStiff, LineSearch
	Interface/Joint Dashpots		
24	Continuous Dashpot Model 2D	Yes	TangentStiff, LineSearch
25	Continuous Dashpot Model 3D	Yes	TangentStiff, LineSearch
	Field Problems		
9	Isotropic, linear heat transfer	Yes	None
10	Isotropic, linear seepage flow	Yes	None
11	2-D Orthotropic, linear heat transfer	Yes	None
12	2-D Orthotropic, linear seepage flow	Yes	None
13	3-D Orthotropic, linear heat transfer	Yes	None
14	3-D Orthotropic, linear seepage flow	Yes	None

Table 2.2: Description of Available Material Models

will be terminated prematurely. The material model identifier must be greater than 0 and less than 15 (i.e., acceptable values are 1 through 14), otherwise the program will print an error message to the formatted output file and execution will be terminated prematurely. The material identifier is used by the program only when the `LEFM` option (see Section 1.4.1) has been included in the program control block. If the structure to be analyzed is homogeneous or the crack is isolated within a homogeneous sub-domain of the structure, the value of the material identifier must be zero. If the structure has a crack on a bi-material interface, the values of the material identifier should be 1 and 2 for the materials adjacent to the interface/Joint and zero for materials outside the integral paths. The numbering convention for the material identifier is shown in Figure 2.1. The materials adjacent to the interface/Joint must be isotropic, linearly elastic (i.e., material model 1). If values other than 0, 1, or 2 are encountered or a non-zero material identifier is used with an illegal material model the program will print an error message to the formatted output file and execution will be terminated prematurely.

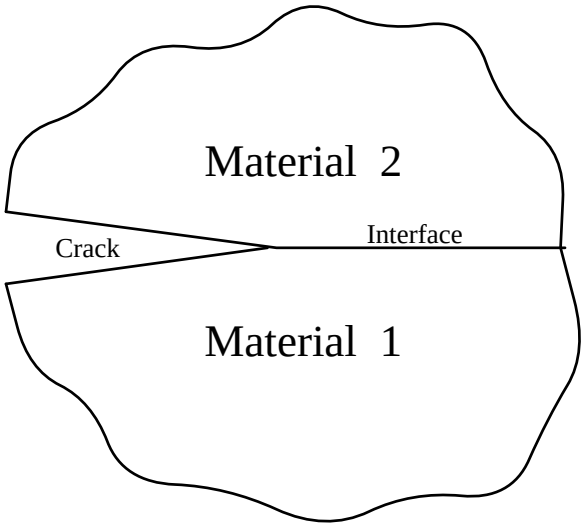

Figure 2.1: Numbering Conventions for Bi-Material Crack

The material models are not uniformly supported for all elements in the element library. In the case of the transversely isotropic, linearly elastic model the truss/spring and axisymmetric stress-strain idealizations simply are not appropriate. However, the 3-dimensional continuum elements should eventually support this material model, but currently lack the appropriate transformation matrix. Tables 2.3-2.5 indicate which material models are supported for each element in the element library. The reader is referred to Tables 2.6, 2.7 and 2.8 for descriptions of the elements in the element library.

2.2.1.2 Inside Load Increments

`ElementGroup` can also be used inside a load increment to alter the parameters of the corresponding constitutive model (such as Young's modulus, the coefficient of thermal expansion, or others).

When used in this context, following `ElementGroup` is the element group identifier to be

Elem. Type	Material Model																
	Linear Elastic			Plasticity			FCM/ICM		Field						Smeared		
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Truss/Spring																	
1	yes	no	no	yes	no	no	no	no	no	no	no	no	no	no	no	no	no
Quadrilaterals Linear																	
2	yes	yes	no	no	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
3	yes	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
4	yes	no	no	yes	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
5	yes	yes	no	no	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
6	yes	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
7	yes	no	no	yes	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
Triangles Linear																	
8	yes	yes	no	no	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
9	yes	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
10	yes	no	no	yes	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
Triangles Quadratic																	
11	yes	yes	no	no	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
12	yes	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
13	yes	no	no	yes	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
14	yes	yes	no	no	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
15	yes	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
16	yes	no	no	yes	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
Quadrilateral Quadratic																	
17	yes	yes	no	no	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
18	yes	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
19	yes	no	no	yes	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
20	yes	yes	no	no	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
21	yes	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
22	yes	no	no	yes	yes	no	no	no	no	no	no	no	no	no	yes	yes	yes
2D Interface/Joints																	
23	no	no	no	no	no	yes	yes	yes	no	no	no	no	no	no	yes	no	no
24	no	no	no	no	no	yes	yes	yes	no	no	no	no	no	no	no	no	no

Table 2.3: Availability of Material Models for Element Library; 2D Stress Analysis.

Elem. Type	Material Model																
	Linear Elastic			Plasticity			FCM/ICM		Field						Smeared		
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Tetrahedron																	
25	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	no	yes	yes	yes
26	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	no	yes	yes	yes
Wedge Linear																	
27	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	no	yes	yes	yes
Brick Linear																	
28	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	no	yes	yes	yes
29	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	no	yes	yes	yes
Wedge Quadratic																	
30	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	no	yes	yes	yes
31	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	no	yes	yes	yes
Brick Quadratic																	
32	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	no	yes	yes	yes
3D Interface/Joint																	
33	no	no	no	no	no	yes	yes	yes	no	no	no	no	no	no	no	no	no
34	no	no	no	no	no	yes	yes	yes	no	no	no	no	no	no	no	no	no
35	no	no	no	no	no	yes	yes	yes	no	no	no	no	no	no	no	no	no
36	no	no	no	no	no	yes	yes	yes	no	no	no	no	no	no	no	no	no
Pyramid Linear																	
45	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	no	yes	yes	yes
Pyramid Quadratic																	
46	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	no	yes	yes	yes
47	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	no	yes	yes	yes
48	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	no	yes	yes	yes

Table 2.4: Availability of Material Models for Element Library; 3D Stress Analysis.

Elem. Type	Material Model																
	Linear Elastic			Plasticity			FCM/ICM		Field						Smeared		
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Triangle																	
37	no	no	no	no	no	no	no	no	yes	yes	yes	yes	yes	yes	no	no	no
38	no	no	no	no	no	no	no	no	yes	yes	yes	yes	yes	yes	no	no	no
Quadrilateral																	
39	no	no	no	no	no	no	no	no	yes	yes	yes	yes	yes	yes	no	no	no
40	no	no	no	no	no	no	no	no	yes	yes	yes	yes	yes	yes	no	no	no
Tetrahedron																	
41	no	no	no	no	no	no	no	no	yes	yes	yes	yes	yes	yes	no	no	no
Wedge																	
42	no	no	no	no	no	no	no	no	yes	yes	yes	yes	yes	yes	no	no	no
Brick																	
43	no	no	no	no	no	no	no	no	yes	yes	yes	yes	yes	yes	no	no	no
Pyramid																	
76	no	no	no	no	no	no	no	no	yes	yes	yes	yes	yes	yes	no	no	no

Table 2.5: Availability of Material Models for Element Library; Heat-Seepage Elements

changed, followed by the new set of parameters (defined below).

2.2.2 Element Library

MERLIN includes a wide number of elements from which users can select to perform their analysis. These elements are referred to collectively as the `element library`. An element in the element library is identified by a integer which is called the `element type`. Each element has a fixed number of nodes arranged in a simple geometric configuration.

Elements types are given by

Table	Description
2.6	Two dimensional stress elements
2.7	Three dimensional stress elements
2.9	Rods (one dimensional) elements
2.8	Basic heat transfer elements
2.10	Water (nearly incompressible) elements with selective reduced integration
2.11	Elements with reduced integration and hourglass control.

The element descriptions are based on the attributes listed above and the operational status of an element in the mixed-iterative method. Unfortunately, not all elements in the element library can be used with the mixed-iterative method; some can only be used with the displacement method. For some elements, such as truss/spring elements, the concept of C^0 continuous strains and stresses makes no physical sense, but for the higher-order elements more research is required in order to produce a robust means of performing the nodal strain projection. At the present time, the mixed-iterative method has not been implemented for heat transfer and seepage flow analysis. Consequently, Table 2.8 indicates that none of the heat transfer/seepage flow elements can be used with the mixed-iterative method.

Element Type	Description				
	Number of Nodes	Geometry	Idealization	Element Formulation	Mixed-Iterative Method
1	2	2-D Reinf-rod	truss/spring	standard	no
2	4	quadrilateral	plane stress	standard	yes
3	4	quadrilateral	plane strain	standard	yes
4	4	quadrilateral	axisymmetric	standard	yes
5	4	quadrilateral	plane stress	SRI	yes
6	4	quadrilateral	plane strain	SRI	yes
7	4	quadrilateral	axisymmetric	SRI	yes
8	3	triangle	plane stress	standard	yes
9	3	triangle	plane strain	standard	yes
10	3	triangle	axisymmetric	standard	yes
11	6	triangle	plane stress	standard	yes
12	6	triangle	plane strain	standard	yes
13	6	triangle	axisymmetric	standard	yes
14	6	triangle	plane stress	singular	no
15	6	triangle	plane strain	singular	no
16	6	triangle	axisymmetric	singular	no
17	8	quadrilateral	plane stress	standard	yes
18	8	quadrilateral	plane strain	standard	yes
19	8	quadrilateral	axisymmetric	standard	yes
20	9	quadrilateral	plane stress	standard	yes
21	9	quadrilateral	plane strain	standard	yes
22	9	quadrilateral	axisymmetric	standard	yes
23	4	2-D line	interface/Joint	standard	yes
24	6	2-D line	interface/Joint	standard	yes

Note: Singular element numbering should begin with crack tip node.

Table 2.6: 2D Elements for Stress Analysis in Element Library

Element Type	Description				
	Number of Nodes	Geometry	Idealization	Element Formulation	Mixed-Iterative Method
25	4	tetra/Linear	3-D continuum	standard	yes
26	10	tetra/quadratic	3-D continuum	standard	yes
27	6	wedge/Linear	3-D continuum	standard	no
28	8	brick/Linear	3-D continuum	standard	yes
29	8	brick/Linear	3-D continuum	SRI	yes
30	15	wedge/quadratic	3-D continuum	standard	no
31	15	wedge/quadratic	3-D continuum	singular	no
32	20	brick/quadratic	3-D continuum	standard	no
33	6	triangle/linear	interface/Joint	standard	yes
34	8	quadrilateral/linear	interface/Joint	standard	yes
35	12	triangle/quadratic	interface/Joint	standard	yes
36	16	quadrilateral/quadratic	interface/Joint	standard	yes
45	5	pyramid/linear	3-D continuum	standard	no
46	13	pyramid/quadratic	3-D continuum	standard	no
47	13	pyramid/quadratic	3-D continuum	singular	no
48	13	pyramid/quadratic	3-D continuum	singular	no

Table 2.7: 3D Elements for Stress Analysis in Element Library

Element Type	Description				
	Number of Nodes	Geometry	Idealization	Element Formulation	Mixed-Iterative Method
37	3	triangle	2-D continuum	standard	no
38	3	triangle	axisymmetric	standard	no
39	4	quadrilateral	2-D continuum	standard	no
40	4	quadrilateral	axisymmetric	standard	no
41	4	tetrahedron	3-D continuum	standard	no
76	5	pyramid	3-D continuum	standard	no
42	6	wedge	3-D continuum	standard	no
43	8	brick	3-D continuum	standard	no

Table 2.8: Elements for Heat Transfer and Seepage Flow Analysis in Element Library

Element Type	Description				
	Number of Nodes	Geometry	Idealization	Element Formulation	Mixed-Iterative Method
44	2	3D/Line	Truss/Spring	standard	no

Table 2.9: Reinforcement Rods, Springs and Dashpots in Element Library

Element Type	Description				
	Number of Nodes	Geometry	Idealization	Element Formulation	Mixed-Iterative Method
51	4	quadrilateral	Plane strain	SRI	no
52	4	quadrilateral	axisymmetric	SRI	no
53	3	triangle	plane strain	SRI	no
54	3	triangle	axisymetric	SRI	no
55	8	quadrilateral	plane strain	SRI	no
56	8	quadrilateral	axisymmetric	SRI	no
57	9	quadrilateral	plane strain	SRI	no
58	9	quadrilateral	axisymmetric	SRI	no
59	6	triangle	plane strain	SRI	no
60	6	triangle	axisymetric	SRI	no
61	4	tetrahedron	3-D	SRI	no
62	10	tetrahedron	3-D	SRI	no
63	6	pentahedron	3-D	SRI	no
64	8	hexahedron	3-D	SRI	no
65	15	pentahedron	3-D	SRI	no
66	20	hexahedron	3-D	SRI	no
67	5	Pyramid	3-D	SRI	no
68	13	Pyramid	3-D	SRI	no

Table 2.10: Elements for Displacement Based Nearly Incompressible Material

Element Type	Description				
	Number of Nodes	Geometry	Idealization	Element Formulation	Mixed-Iterative Method
71	4	quadrilateral	Plane stress	RI	no
72	4	quadrilateral	Plane strain	RI	no
73	4	quadrilateral	Axisymmeteric	RI	no
74	8	hexahedron	3-D	RI	no
75	5	Pyramid	3-D	RI	no

Table 2.11: Reduced Integration Elements with Hourglass Control

When selecting which elements will be used, the user should consider what type of analysis will be performed. For instance, if a LEFM analysis is to be performed in 2-D and the stress intensity factors are to be computed using the displacement correlation method, then the elements surrounding the crack tip must be 6-node triangles with the singular formulation (i.e., element types 14, 15, or 16) and the elements in the rest of the mesh must be either standard formulation 6-node triangles (i.e., element types 11 to 13) or 9-node quads (i.e., element types 17, 18, or 19). When mixing element types within a mesh, the elements must not be of different order and, except in special circumstances, should not be of different idealizations. Mixing element types of different order (e.g., a 4-node quad and a 6-node triangle) is not permitted because the program does not support a constraint mechanism to tie the primary field variables for the mid-side node of the higher-order element to those at the corner nodes of the lower-element. A case in which it may be necessary to mix elements with different idealizations would be a buttressed dam, with the dam consisting of plane strain elements and the buttress consisting of plane stress elements. Mixing elements for stress analysis with elements for heat transfer or seepage flow analysis is not permitted in the same mesh; two meshes using different but compatible element types are required to perform uncoupled thermo-elastic or poro-elastic analysis.

For stress analysis the user is confronted with a number of choices when selecting an element type. For LEFM analysis using the displacement correlation method of stress intensity factor computation and fracture mechanics analysis with mixed mode crack propagation, higher-order elements are required. When remeshing, triangular elements are more appropriate and ideally higher-order elements should be used. We should note that an excessive number of lower-order triangles would be required to produce the same level of accuracy as a much coarser mesh of higher-order elements. If a higher-order element is not required for the analysis to be performed, the user is free to choose between the standard lower-order and higher-order elements and the SRI lower-order elements. In many cases, the SRI lower-order elements will yield results comparable to those of the standard higher-order elements with the same mesh topology (i.e., with the same number of elements but significantly fewer degrees of freedom). The SRI lower-order elements are superior to the standard lower-order elements for coarse to medium density meshes; they tend to converge to the same values as the mesh is refined. Therefore, the SRI lower-order elements are recommended for all analysis that do not require the use of higher-order elements.

Selecting the proper element for an analysis is very important, but the user should be aware that a "good" element in a "bad" mesh will most likely produce mediocre results at best. The mesh should be denser in areas where the gradient of the primary field variable is known to be high than in areas where the gradient of the primary field variable is known to be low. If mesh transitions are used between adjacent areas of high and low mesh density, they should be sufficiently far away from the area where the gradient of the primary field variable is high, so that the transition will not adversely affect the computed primary field variables in areas of higher mesh density. In general, collapsing 4-node quads to create 3-node triangles should be avoided. However, if creating a mesh transition without using triangles becomes difficult, it may become necessary to collapse 4-node quads into triangles or use 3-node triangles. When this situation arises, the collapsing of 4-node quads should be done in moderation, with as few 3-node triangles as possible.

2.2.2.1 Element Numbering

2.2.2.1.1 Solid Elements The nodal numbering conventions for the various combinations of nodes and geometric configurations are given in Figures 2.2 through 2.13.

Separate element types are provided for stress analysis and for heat transfer or seepage flow analysis. For stress analysis, each element is limited to modeling only one of the following six idealizations:

1. Truss/Spring.
2. Plane stress.
3. Plane strain.
4. Axisymmetric.
5. 3-D continuum.
6. Interface/Joint.

For heat transfer and seepage flow analysis, each element is limited to modeling only one of the following three idealizations:

1. 2-D continuum.
2. Axisymmetric.
3. 3-D continuum.

All elements in the element library are based on standard isoparametric formulations, but some elements have formulations that have been modified to produce enhanced behavior in situations where the standard formulations are deficient. Elements are grouped into families according to the order of their shape functions and their formulation, with each family generally including both 2- and 3-D elements. A typical element family for stress analysis includes elements with plane stress, plane strain, axisymmetric, and 3-D continuum idealizations, while a typical element family for heat transfer and seepage flow analysis includes 2-D continuum, axisymmetric, and 3-D continuum idealizations. There is a family of lower-order elements for stress analysis (i.e., three 4-node quads and an 8-node brick) which use a selective-reduced integration technique (SRI) to produce enhanced results for problems with significant bending modes. A family of higher-order elements (i.e., three 6-node triangles and a 15-node wedge) with the positions of certain midside nodes shifted to produce a singularity at one of the nodes are available to be used as crack tip elements in stress analysis.

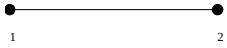

Figure 2.2: Nodal Numbering for 2-Node Bar Element

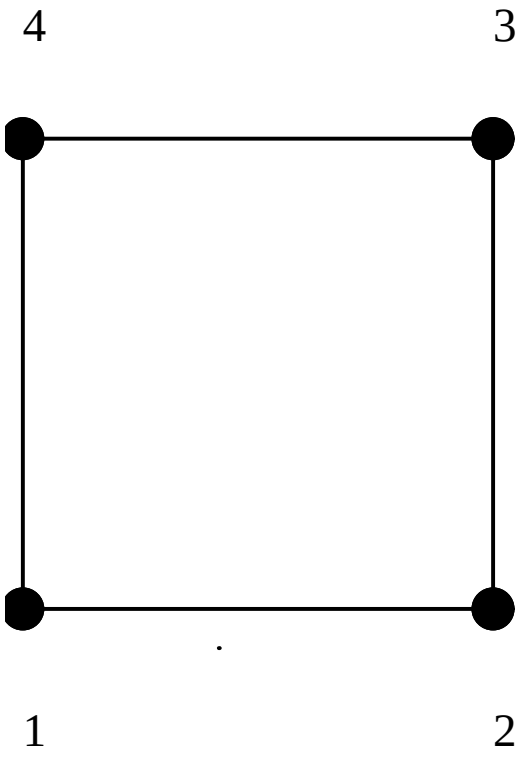

Figure 2.3: Nodal Numbering for 4-Node Quadrilateral Element

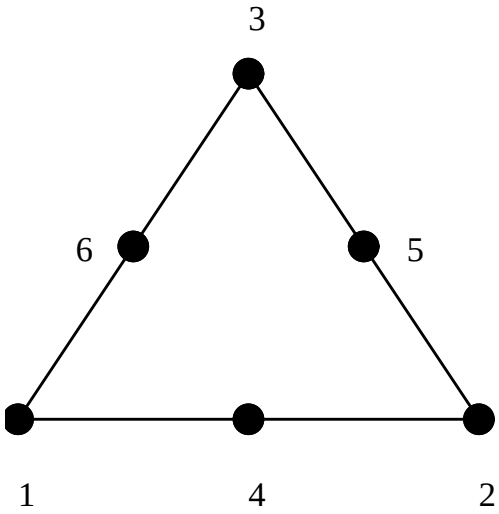

Figure 2.4: Nodal Numbering for 6-Node Triangle Element

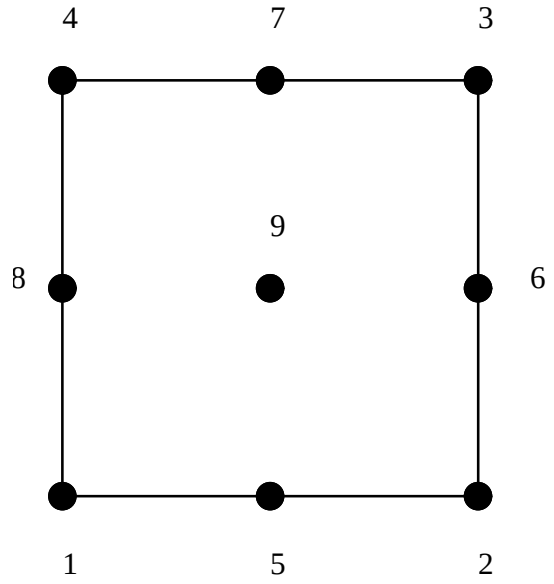

Figure 2.5: Nodal Numbering for 9-Node Quadrilateral Element

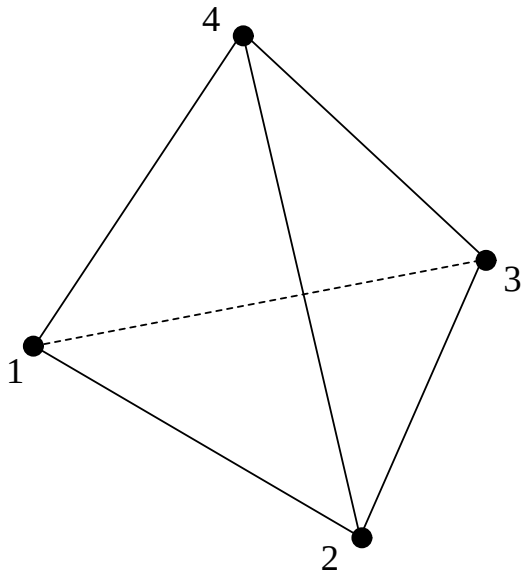

Figure 2.6: Nodal Numbering for 4-Node Tetrahedral Element

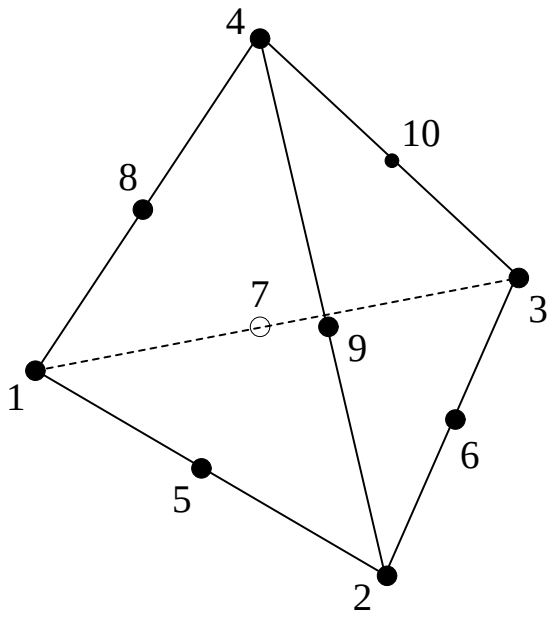

Figure 2.7: Nodal Numbering for 10-Node Tetrahedral Element

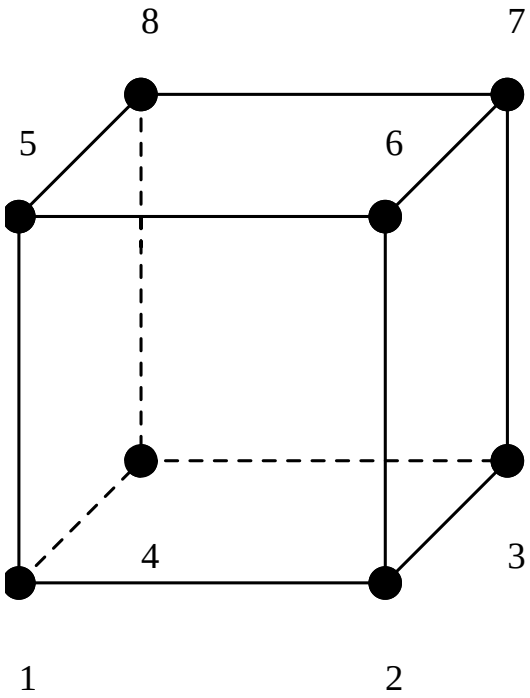

Figure 2.8: Nodal Numbering for 8-Node Brick Element

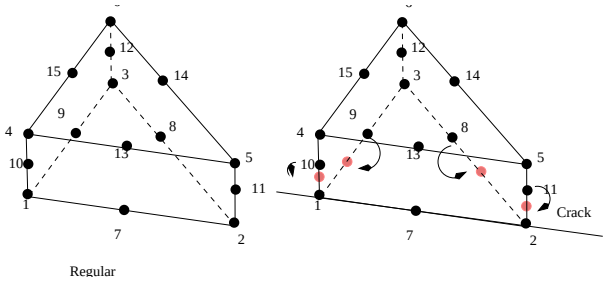

Figure 2.9: Nodal Numbering for 15-Node Wedge Element

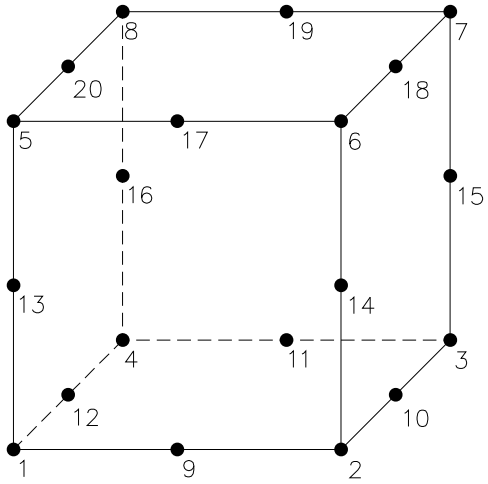

Figure 2.10: Nodal Numbering for 20-Node Brick Element

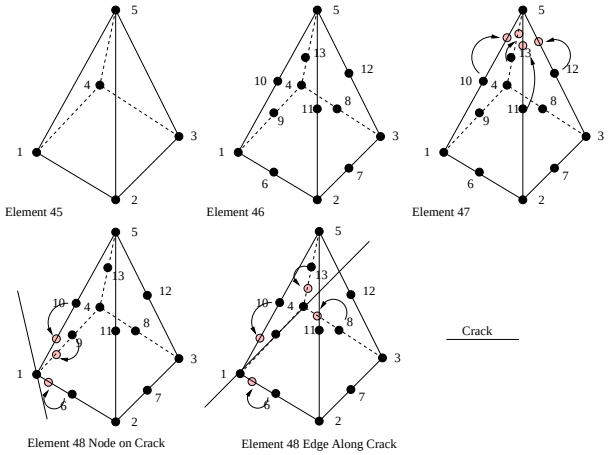

Figure 2.11: Pyramid Elements; linear, Quadratic, and Singulars

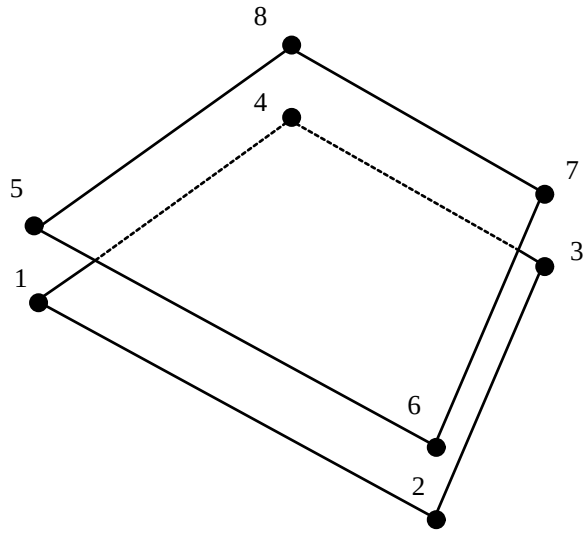

Figure 2.12: Nodal Numbering for 8-Node Interface/Joint 3-D Element

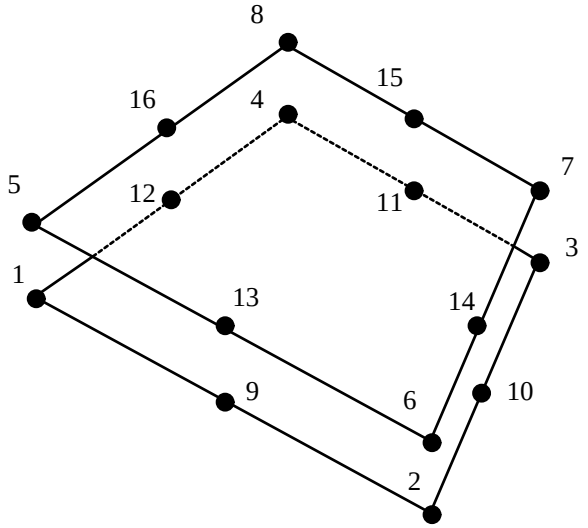

Figure 2.13: Nodal Numbering for 16-Node Interface/Joint 3-D Element

2.2.2.1.2 Interface Elements There is a family of interface elements available for modeling the fracture process zone (FPZ) in the FCM or non-monolithic connection between dissimilar materials, Figs 2.14 to 2.17.

Interface element must be numbered counterclockwise for the crack to recognize if it is opening and closing.

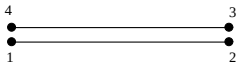

Figure 2.14: Nodal Numbering for 4-Node Interface/Joint 2-D Element

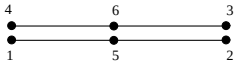

Figure 2.15: Nodal Numbering for 6-Node Interface/Joint 2-D Element

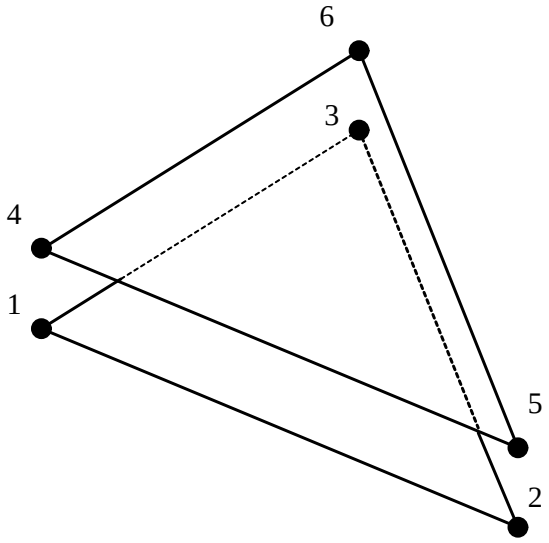

Figure 2.16: Nodal Numbering for 6-Node Interface/Joint 3-D Element

2.2.2.2 Element Surface Numbering

Element surfaces are defined counter-clockwise around the boundary of a 2-dimensional element. An individual element surface is defined to extend from the node defining one corner of the element to the node defining the next corner of the element passing through any nodes in between. The first element surface always begins at node 1 and the last element surface always ends at node 1. For example, a 4-node quadrilateral, as shown in Figure 2.18, has 4 surfaces defined by nodes 1 and 2; 2 and 3; 3 and 4; and 4 and 1.

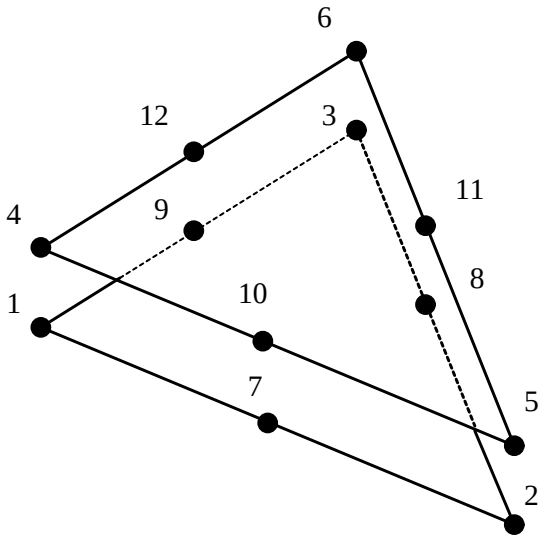

Figure 2.17: Nodal Numbering for 12-Node Interface/Joint 3-D Element

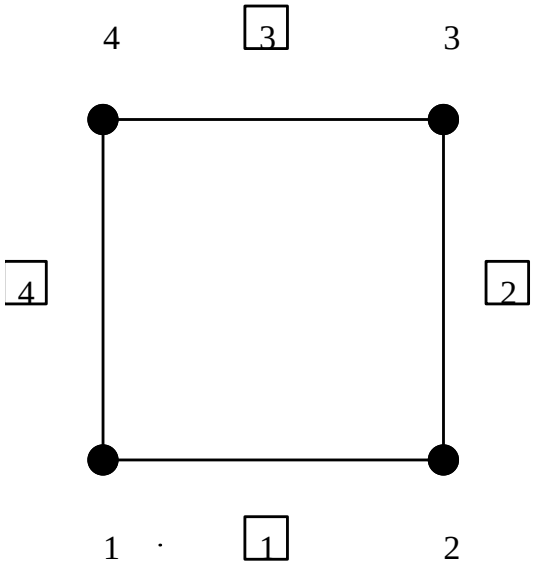

Figure 2.18: Element Surfaces for a 4-Node Quad

For three-dimensional elements there are no simple rules that can be used to define element surfaces. Element surface definitions for three-dimensional elements are shown in Figures 2.19 through 2.23. In these figures the elements are "unfolded" to simplify identification of the various surfaces.

As a rule, numbering is counterclockwise with respect to the "lower surface" (defined first), and the "upper surface" (defined next).

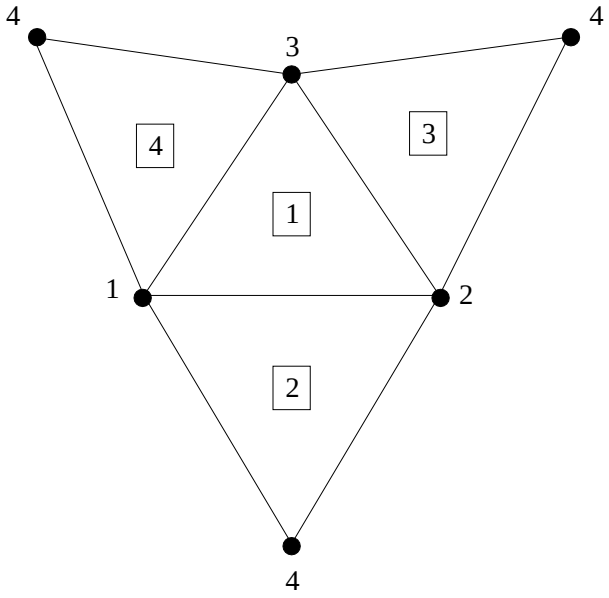

Figure 2.19: Element Surfaces for a 4-Node Tetrahedron

2.2.2.3 Connectivity

The `Connectivity` option enables the user to define the connectivity for each element. The line in the input file following the keyword contains the number of elements. The number of elements must be greater than zero, otherwise the program will print an error message to the formatted output file and program execution will be terminated prematurely. The connectivity for each element must be explicitly defined in the input file; no generation capabilities are included in the program. When defining the connectivity for higher order elements (i.e., elements with mid-side nodes), only the corner nodes are defined by the user in the input file; the program automatically inserts the mid-side nodes. All element edges with mid-side nodes are straight, eliminating the errors associated with isoparametric distortion that can occur when edges are curved. The element number and the element group number to which the element belongs precede the connectivity. The element number is checked for consistency (i.e., the fifth element must be element number 5) and the element group number is checked for validity, which means that the `ElementGroup` option must precede this option in the mesh definition block. Failure to define the element group data before the connectivity or discovery of an inconsistent element number or an invalid element group number are errors, both of which will cause the program to print an error message to the formatted output file and terminate execution prematurely. Failure to include an `Connectivity` option is a fatal input error, which will cause the program to print an error message to the formatted output file and terminate execution prematurely.

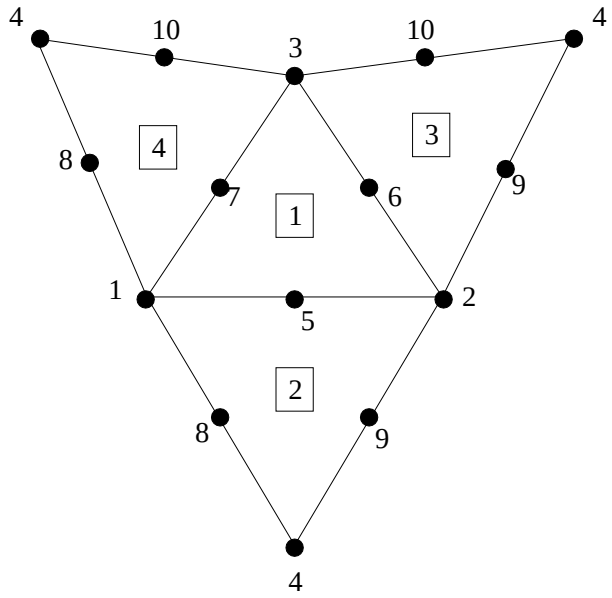

Figure 2.20: Element Surfaces for a 10-Node Tetrahedron

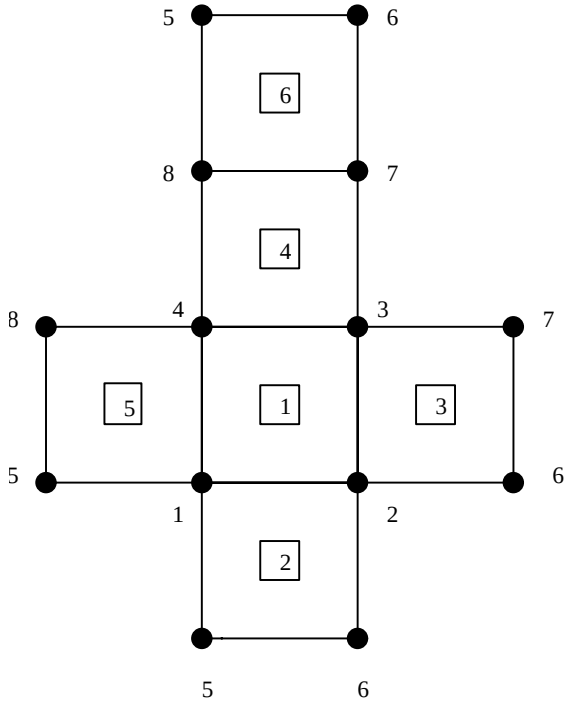

Figure 2.21: Element Surfaces for a 8-Node Brick

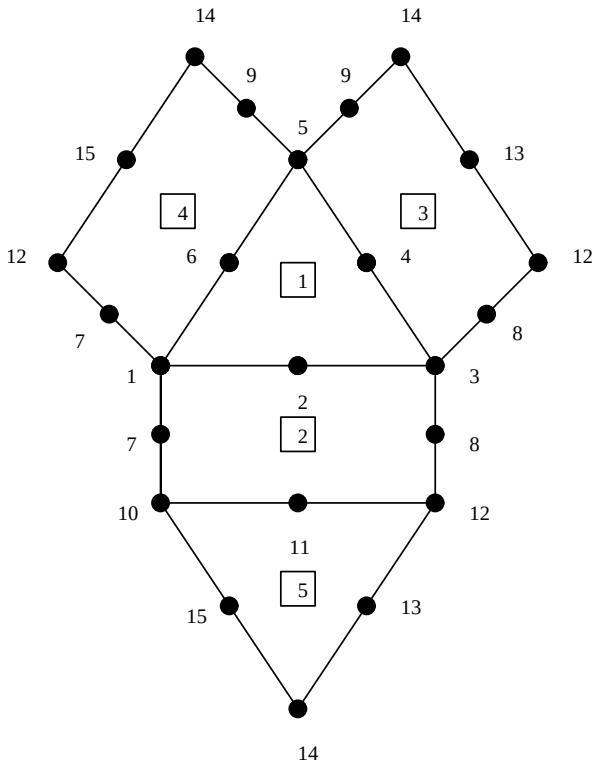

Figure 2.22: Element Surfaces for a 15-Node Wedge

Element No.	Group No.	Node 1	...	Node n

Table 2.12: Element Connectivity Input

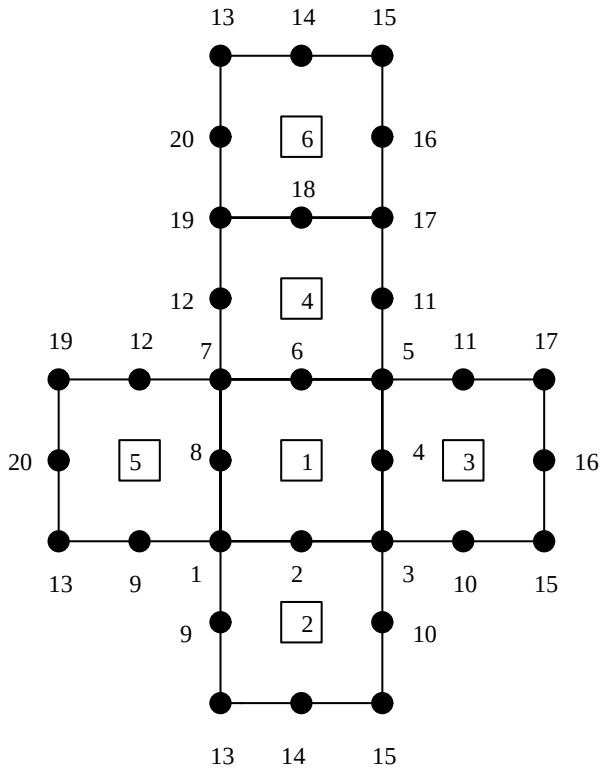

Figure 2.23: Element Surfaces for a 20-Node Brick

2.2.3 Constitutive Models

2.2.3.1 Preliminary Notes

A few preliminary important notes:

Thickness of solid elements: for plane strain it is internally defaulted to 1., no matter what the user specifies. For Axisymmetric analysis, the user should specify $2\pi R$.

Thickness of interface elements: Those elements do not know if the analysis is plane stress, strain or axisymmetric (in 2D), thus the thickness should be the same as in the surrounding elements ($2\pi R$ for axisymmetric, and 1 for plane strain).

Tractions Are always in units of F/L^2

2.2.3.2 Linear Elastic

2.2.3.2.1 Isotropic, Linear Elastic (MM: 1)

The properties required for the isotropic, linear elastic material model (i.e., the value of the material model identifier is 1) are as follows:

1. Element thickness, h , or cross sectional area, A when used in conjunction with `ReinfRod` (steel reinforcement).
2. Mass density[2] of the material, ρ.
3. Coefficient of thermal expansion, α.
4. Elastic modulus of the material, E.
5. Poisson's ratio of the material, ν.
6. Fracture toughness of the material, K_{Ic}.
7. Critical energy release rate of the material, G_{Ic}.
8. Tensile strength (needed if `AutoCrackProp` option is specified).

For plane strain and axisymmetric idealizations the program automatically assigns the element thickness. 1 for plane strain, and $2\pi r$ where $r = x$ for axisymmetric problems; the user cannot override these values.

For 3-D continuum idealizations the program ignores the element thickness, but a value for it must be provided in the input file. The last two pieces of information, K_{Ic} and G_{Ic}, are only used by the program for LEFM analysis, but values for them must be provided in the input file in all cases.

When used in conjunction with `ReinfRod` (steel reinforcement), then only A, E and ν should be defined, all other entries should be zero.

[2]Not to be confused with the weight density γ. g will be specified later in the `BodyForces` option.

h or A	ρ	α	E	ν	K_{Ic}	G_{Ic}

Table 2.13: Properties for Material Model 1

2.2.3.2.2 Transversely Isotropic, Linear Elastic (MM: 2)

The properties for the transversely isotropic, linear elastic (**?**) material model (i.e., the value of the material model identifier is 2) are as follows:

1. Element thickness, h.
2. Mass density of the material, ρ.
3. Coefficient of thermal expansion in the isotropic plane of the material, α_1.
4. Coefficient of thermal expansion perpendicular to the isotropic plane of the material, α_2.
5. Elastic modulus in the isotropic plane of the material, E_1.
6. Poisson's ratio in the isotropic plane of the material, ν_1.
7. Elastic modulus perpendicular to the isotropic plane of the material, E_2.
8. Poisson's ratio perpendicular to the isotropic plane of the material, ν_2.
9. Shear modulus perpendicular to the isotropic plane of the material, G_{12}.
10. Angle of orientation for the isotropic plane, θ.

This material model is not supported for 1-D elements or 3-D continuum elements. For plane strain and axisymmetric idealizations the program automatically assigns a value for the element thickness; the user cannot override these values. The angle of rotation is measured in degrees from the x-axis and is considered to be positive in the counter-clockwise direction.

h	ρ	α_1	α_2	E_1	ν_1	E_2	ν_2
G_{12}	θ						

Table 2.14: Properties for Material Model 2

2.2.3.2.3 3-D Orthotropic, Linear Elastic (MM: 3)

The properties for the 3-D orthotropic, linear elastic (**?**) material model (i.e., the value of the material model identifier is 3) are as follows:

1. Element thickness, h.
2. Mass density of the material, ρ.
3. Coefficient of thermal expansion in material direction 1, α_1.

4. Coefficient of thermal expansion in material direction 2, α_2.
5. Coefficient of thermal expansion in material direction 3, α_3.
6. Elastic modulus in material direction 1, E_1.
7. Poisson's ratio in material direction 1, ν_{23}.
8. Elastic modulus in material direction 2, E_2.
9. Poisson's ratio in material direction 2, ν_{13}.
10. Elastic modulus in material direction 3, E_3.
11. Poisson's ratio in material direction 3, ν_{12}.
12. Shear modulus in material direction 1, G_{23}.
13. Shear modulus in material direction 2, G_{13}.
14. Shear modulus in material direction 3, G_{12}.
15. x-component of a vector defining material direction 1, v_x^1.
16. y-component of a vector defining material direction 1, v_y^1.
17. z-component of a vector defining material direction 1, v_z^1.
18. x-component of a vector defining material plane 1-2, v_x^2.
19. y-component of a vector defining material plane 1-2, v_y^2.
20. z-component of a vector defining material plane 1-2, v_z^2.

This material model is not supported for 1-D elements or 2-D continuum elements. For 3-D continuum idealizations the program ignores the element thickness, but a value for it must be provided in the input file. The direction vectors $\mathbf{v}^1$ and $\mathbf{v}^2$ are not required to be unit vectors; the program unitizes them when computing the transformation matrix for the material stiffness matrix.

h	ρ	α_1	α_2	α_3	E_1	ν_{23}	E_2
ν_{13}	E_3	ν_{12}	G_{23}	G_{13}	G_{12}	v_x^1	v_y^1
v_z^1	v_x^2	v_y^2	v_z^2				

Table 2.15: Properties for Material Model 3

2.2.3.3 Nearly Incompressible (MM: 20)

Displacement based nearly incompressible elements must be used in conjunction with this material type defined in terms of:

1. Thickness; (internally ignored; taken as 1. for plane strain, $2\pi x$ for axisymmetry).

2. Mass density of the fluid, ρ.
3. Coefficient of thermal expansion, α.
4. Elastic bulk modulus, K.
5. Shear modulus G.
6. Viscosity μ.

h	ρ	α	E	ν	μ

Table 2.16: Properties for Material Model 20;

Recall that

$$K = \frac{E}{3(1-2\nu)} \tag{2.1-a}$$

$$G = \frac{E}{2(1+\nu)} \tag{2.1-b}$$

Table 2.17 gives the elastic properties of water an other engineering materials. It should be

Material	E GPa	ν	G GPa	K GPa
Steel	207	0.25	82.8	138
Concrete	27.6	0.20	11.5	15.3
Water	0	0.50	0	2.1
"Water"	6.0×10^{-4}	0.49995	2.1×10^{-4}	2.1

Table 2.17: Elastic Properties of Steel, Concrete and Water

noted that shear modulus is zero, however under dynamic loading viscosity and boundary layer effects allow fluids to resist shear (**?**).

2.2.3.4 Plasticity

2.2.3.4.1 J_2 Plasticity (MM: 4)

The properties for the J_2 plasticity (**?**) material model (i.e, the value of the material model identifier is 4) are as follows:

1. Element thickness, h (ignored in plane strain, $2\pi x$ for axisymmetry), or cross sectional area if element 1 or 44.
2. Mass density of the material, ρ.
3. Coefficient of thermal expansion, α.

4. Elastic modulus of the material, E.
5. Poisson's ratio of the material, ν (zero for bar element)
6. Yield stress, σ_Y.
7. Hardening modulus, H.

For plane strain and axisymmetric idealizations the program automatically assigns a value for the element thickness; the user cannot override these values. For 3-D continuum idealizations the program ignores the element thickness, but a value for it must be provided in the input file.

h	ρ	α	E	ν	σ_Y	H

Table 2.18: Properties for Material Model 4

2.2.3.4.2 Drucker-Prager Plasticity (MM: 5)

The properties for the Drucker-Prager plasticity (**?**) material model (i.e, the value of the material model identifier is 5) are as follows:

1. Element thickness, h (automatically set to $2\pi x$ for axisymmetry, 1 for plane strain).
2. Mass density of the material, ρ.
3. Coefficient of thermal expansion, α.
4. Elastic modulus of the material, E.
5. Poisson's ratio of the material, ν.
6. Maximal step size for the strains, ϵ_{max}.
7. Compressive strength, f_C.
8. Angle of friction, ϕ.
9. Direction of plastic flow, ψ.
10. Hardening/softening parameter, β.
11. Decimal percentage of the original cohesion that defines the residual strength envelope, c_{res}.

For plane strain and axisymmetric idealizations the program automatically assigns a value for the element thickness; the user cannot override these values. **Currently, this material model has not been implemented for 3-D continuum and, therefore, should not be used for 3-D continuum until this notice is removed.**

The direction of the plastic flow ψ is the dilatancy angle and is usually smaller than ϕ. The hardening/softening parameter β can have values in the interval from -0.01 to 1, where values

less than 0.0 means softening, the zero value means perfectly plastic behaviour and the values larger than zero means hardening.

The material parameters for Drucker-Prager plasticity should be determined by fitting the experimental data from tension and compression tests. For more details about this model see (**?**).

h	ρ	α	E	ν	ϵ_{max}	f_C	ϕ	ψ	β	c_{res}

Table 2.19: Properties for Material Model 5

2.2.3.4.3 Coulomb Friction (MM: 6)

The properties for the Coulomb friction (**?**) material model (i.e, the value of the material model identifier is 6) are as follows:

1. Element thickness, h.
2. Mass density of the material, ρ.
3. Coefficient of thermal expansion, α.
4. Cohesion, c.
5. Friction angle, ϕ.
6. Normal stiffness, k_n.
7. Tangential stiffness, k_t.

If this element is used in an axisymmetric model, the thickness must be specified as $2\pi x$ ($x = r$).

As this material model is supported only for the interface/Joint elements, the mass density is not used by the program, but a value for it must be provided in the input file. The friction angle is measured in degrees. The normal stiffness defines the normal stress in terms of the relative normal displacements for the two faces of the interface/Joint elements and the tangential stiffness defines the shear stress in terms of the relative tangential displacements for the two faces of the interface/Joint elements. Both of these values must be sufficiently large to prevent excessive relative displacements but not so large as to dominate the other terms in the global stiffness matrix for the affected degrees of freedom. A trial and error procedure may be required to determine suitable values.

h	ρ	α	c	ϕ	k_n	k_t

Table 2.20: Properties for Material Model 6

2.2.3.5 Smeared Cracks

All smeared crack models are based on the Fracture Plastic model of (**?**). What differentiates them is that:

- Model 19 is the standard implementation.
- Model 15 allows the user to specify an evolution of tensile strength, compressive strength, fracture energy and onset of compressive nonlinearities in terms of increments (through user defined curves).
- Model 16 allows the user to define curve for some of the parameters, but is also a generalization of Model 19 as it accounts for xxx
- Model 18 is the most advanced model and accounts for the effect of confinement.

2.2.3.5.1 Fracture Plastic Model (MM: 19)

1. Thickness
2. Mass density
3. Thermal expansion
4. E
5. Poisson's ratio
6. f_t Tensile strength
7. G_F Exponential softening based on Hordijk is used
8. f_c Compressive strength
9. w_d Compressive critical displacement (should be -0.0005 m)
10. Factor β for return direction 0.0 - material volume is preserved <0.0 material is compacting > 0.0 material is dilating.
11. Factor e which defines roundness of failure surface, $0.5 \leq e \leq 1.0$, where $e = 0.5$ corresponds to triangle, and $e = 1.0$ corresponds to a circle, Fig. **??**.
12. f_{c0} Onset of nonlinearity in compression, must be negative.
13. ε_{cp} Plastic strain at compressive strength, must be negative.

This model accounts for the nonlinear pre-peak stress-strain curve.

Note: Tensile strength should not be higher than $f'_c/10$, f_{c0} should not be less than 3/5 f'_c.

2.2.3.5.2 Fracture Plastic Model; Time Dependant Properties (MM: 15)

Non-linear incremental fracturing material with variable properties (analogous to Material 19).

1. Thickness
2. Mass density
3. Thermal expansion
4. ID_E ID of curve defining the time increment evolution of the Elastic Modulus
5. Poisson's ratio
6. $ID_{f'_t}$ Id of the user curve (`UserCurve`) defining the time increment evolution of the tensile strength f'_t
7. ID_{G_F} Id of the user curve (`UserCurve`) defining the time increment evolution of the softening G_F
8. $ID_{f'_c}$ Id of the user curve (`UserCurve`) defining the time increment evolution of the compressive strength f'_c
9. w_d Compressive critical displacement (should be -0.0005 m)
10. Factor β for return direction 0.0 - material volume is preserved <0.0 material is compacting > 0.0 material is dilating.
11. Factor e which defines roundness of failure surface, $0.5 \leq e \leq 1.0$, where $e = 0.5$ corresponds to triangle, and $e = 1.0$ corresponds to a circle, Fig. **??**.
12. $ID_{f_{c0}}$ Id of the user curve (`UserCurve`) defining the time increment evolution of the onset of nonlinearity in compression, must be negative.
13. ε_{cp} Plastic strain at compressive strength, must be negative.

This model accounts for the nonlinear pre-peak stress-strain curve.

Note: Tensile strength should not be higher than $f'_c/10$, f_{c0} should not be less than 3/5 f'_c.

2.2.3.5.3 Fracture Plastic Model with User Curves (MM: 16)

Non-linear incremental fracturing material with possibility to specify user-defined laws for certain material properties.variable properties (analogous to Material 19).

1. Thickness
2. Mass density
3. Thermal expansion
4. E
5. Poisson's ratio

6. f_t Tensile strength
7. $ID_{f_t^{norm}}$ Id of the user curve (`UserCurve`) defining the hardeining/softening law for f_t - tensile strength, x - strain, y - relative tensile strength normalized with respect to fc.
8. l_{ch}^t characteristic size for tension, used to scale the strain in the hardening/softening law after the localization onset in order to recover a mesh size objective behavior (usually 1.5 aggregate size).
9. ε_{loc}^t Onset of localization in tension (needed namely to differentiate hardening and softening)
10. f_c Compressive strength
11. $ID_{f_c^{norm}}$ Id of the user curve (`UserCurve`) defining the hardeining/softening law for f_c - compressive strength, x - strain, y - relative compressive strength normalized with respect to f_c.
12. l_{ch}^t Characteristic size for compression - used to scale the strain in the hardening/softening law after the localization onset in order to recover a mesh size objective behavior (usually 3 times the aggregate size).
13. ε_{loc}^c Onset of localization in tension (needed namely to differentiate hardening and softening)
14. Factor β for return direction 0.0 - material volume is preserved < 0.0 material is compacting > 0.0 material is dilating.
15. e Defines roundness of failure surface, $0.5 \leq e \leq 1.0$, where e = 0.5 corresponds to triangle, and e = 1.0 corresponds to a circle, Fig. **??**.
16. Id of the user curve (`UserCurve`) defining the law for tensile reduction due to minimal compressive stress, x - relative minimal compression normalized with respect to f_c, y - relative tensile strength.
17. Id of the user curve (`UserCurve`) defining the shear retention factor (i.e. shear stiffness reduction after cracking based on the strain normal to the crack), x - normal strain, y - relative shear stiffness reduction.
18. $\varepsilon_{loc}^{shear}$ Onset of localization in shear (needed namely to differentiate hardening and softening)

This model accounts for the nonlinear pre-peak stress-strain curve.

Note: Tensile strength should not be higher than $f'_c/10$, f_{c0} should not be less than 3/5 f'_c.

2.2.3.5.4 Pressure Sensitive Fracture Plastic Model (MM: 18)

1. Thickness
2. Mass density

3. Thermal expansion
4. E
5. Poisson's ratio
6. f_t Tensile strength
7. G_F Exponential softening based on Hordijk is used
8. f_c Compressive strength (should be negative)
9. Factor e which defines roundness of failure surface, $0.5 \leq e \leq 1.0$, where $e = 0.5$ corresponds to triangle, and $e = 1.0$ corresponds to a circle, Fig. **??**, recommended value 0.52
10. $\varepsilon^p_{\nu,p}$ Volumetric plastic strain at compressive strength
11. t parameter for compressive softening (usually $t = 3\varepsilon^p_{\nu,p}$)
12. l^c_{ch} Characteristic length size in compression (size of laboratory cylinders for compreesion tests, typically 0.1-0.2 m)
13. ε^c_{loc} onset of strain localization in compression (usually $\varepsilon^c_{loc} = \varepsilon^p_{\nu,p}$)
14. c maximal reduction of compressive strength due to concrete cracking (analogous to the compression field theory of Collins, recommended value)
15. γ relative residual tensile tensile crack opening displacement (recommended value 0.3)

This model accounts for the nonlinear pre-peak stress-strain curve.

Note: Tensile strength should not be higher than $f'_c/10$, f_{c0} should not be less than $3/5$ f'_c.

2.2.3.5.5 Kawamoto/Rock (MM: 26) This is based on Kawamoto's model for Rock. It should be used with great care as the model doe not explicitly account for unloading.

1. Thickness
2. Mass density
3. Thermal expansion
4. E_0
5. μ initial Poisson's ratio
6. C_p Shear Shear strength (cohesion)
7. f_t tensile strength
8. ϕ_p - Friction angle
9. E_1 shear softening, > 0 hardening, < 0 softening

10. E_f residual modulus after tensile cracking, can not be zero (select a very small value).
11. C_r residual shear strength (residual cohesion), can be zero.
12. ϕ_r residual friction angle
13. μ_f fracture/crushing Poisson ratio (default 0.49999)
14. a coefficient for E evolution
15. b coefficient for E evolution
16. A coefficient for μ evolution
17. B coefficient for μ evolution
18. f_s compressive/shear brittle softening flag: 1 - brittle sudden drop 0: E_1 is used for shear softening
19. f_r flag affecting the return direction on failure surface: 0 - σ_{max} = constant. $1-(\sigma_{max}+\sigma_{min})/2$ = constant

2.2.3.6 Discrete Cracks

It should be noted that for all discrete elements, Merlin will define the following stresses for the SPider `.pst` file:

σ_{xx}	σ_{yy}	σ_{zz}	ε_{xx}	ε_{yy}	ε_{zz}
τ_{n1}	τ_{n2}	σ_{nn}	CSD_1	CSD_2	COD

2.2.3.6.1 Fictitious Crack Model (MM: 7)

The properties required for the FCM (**?**) material model (i.e., the value of the material model identifier is 7) are as follows:

1. Element thickness, h (used only for 2D problems. If this element is used in an axisymmetric model, the thickness must be specified as $2\pi x$ ($x = r$). In plane strain $h = 1$.
2. Mass density of the material, ρ (zero).
3. Coefficient of thermal expansion, α.
4. Specific fracture energy release rate of the material, G_F.
5. Uniaxial tensile strength, f_t.
6. Crack opening displacement at the break-point for the bilinear law, w_1.
7. Tensile stress at the break-point for the bilinear law, s_1.

The last two pieces of information, w_1 and s_1, are only used by the program when they are positive and non-zero, which indicates that the bilinear softening law will be used. If either value is negative or zero, the linear softening law will be used. Since the user subroutine `ufcmlw` (see Section B.1) has access to this information, w_1 and s_1 could be used to define parameters in the user defined softening law. This material model is accessable only when the `FCM` option has been included in program control block. If an attempt is made to use this material model with non-FCM analysis the program will print a message to the formatted output file and execution will be terminated prematurely.

h	ρ	α	G_F	f_t	w_1	s_1

Table 2.21: Properties for Material Model 7

2.2.3.6.2 Interface/Joint Crack Model 1-Original (MM: 8)

The interface/Joint crack model is to be used in the problems when there is a contact between two different materials and cracking along this contact is expected. Model parameters are 18:

1. Element thickness, h (used only for 2D problems). If this element is used in an axisymmetric model, the thickness must be specified as $2\pi x$ ($x = r$). In plane strain $h = 1$.
2. Mass density of the material, ρ (used only for explicit dynamic analysis).
3. Coefficient of thermal expansion, α (zero).
4. Tangential stiffness K_t.
5. Normal stiffness K_n.
6. Tensile strength, σ_t (can not be equal to zero).
7. Cohesion, c (must be greater than the tensile strength).
8. Friction angle, ϕ_f, [degrees].
9. Dilatancy angle, ϕ_D, [degrees].
10. Specific mode I fracture energy, G_F^I.
11. Specific mode II fracture energy, G_F^{IIa}.
12. Relative value of irreversible deformation, γ. It is the ratio of irreversible crack opening and total inelastic crack opening and it defines the amount of inelastic crack opening that is irreversible, $\gamma \in < 0, 1 >$.
13. Maximal displacement for dilatancy, u_{Dmax}.
14. Tensile stress at the break-point for the bilinear law, s_1.
15. Crack opening displacement at the break-point for the bilinear law, sw_1.
16. Cohesion at the break-point for the bilinear law, c_1.

17. Crack sliding displacement at the break-point for the bilinear law, cw_1.

18. Curve id defining viscosity in terms of crack opening rate (to be specified only for transient stress analysis) (units $FL^{-2}T$), Fig. 2.24.

The last four pieces of information, s_1, sw_1, c_1 and cw_1, are only used by the program when they are positive and non-zero, which indicates that the bilinear softening law will be used. If either value is negative or zero, the linear softening law will be used.

h	ρ	α	K_t	K_n	σ_t	c	ϕ_f	ϕ_D	G_F^I
G_F^{IIa}	γ	u_{Dmax}	s_1	sw_1	c_1	cw_1	μ_{id}		

Table 2.22: Properties for Material Model 8

Note:

1. If G_F is defined explicitly by a curve (sect. 1.1.7), then the value of G_F should be replaced by the negative of the curve number. Internally, Merlin will determine G_F from the absolute value of the specified curve. For help on the selection of the above material properties, refer to Sect. **??**.

2. The following relationship must be satisfied:

$$f_t' < \frac{c}{\tan\phi} \tag{2.2}$$

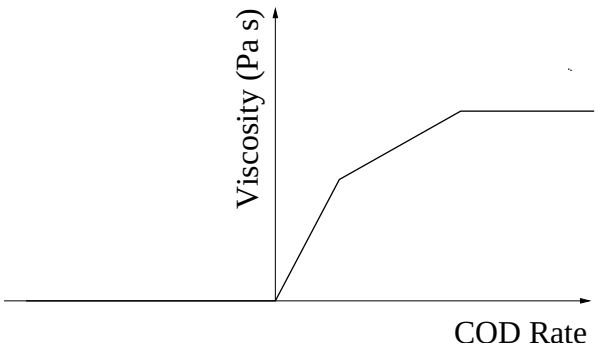

Figure 2.24: Interface/Joint Element Viscosity in terms of Crack Opening Displacement Rate

3. An extended version of this model, one which accounts for the cyclic load, is available under Model 21 (see Sect. **??**).

4. A good estimate for K_n and K_t is the elastic modulus divided by a "negligible" dimension. The larger the K, the smaller the overlap in compression, but the less stable is the solution (which may not converge). If it is smaller, then the solution is likely to be more stable however there may be a non-negligible overlap in compression.

5. Internally, Merlin enforces a minimum stiffness of $K_n/1000$ to preserve the positive definitness of the global stiffness matrix.

2.2.3.6.3 Interface/Joint Crack Model 2-Cyclic (MM: 21)

This is an extension of the original ICM model to account for reverse cyclic shear and degradation of the asperities. Model parameters are 23:

1. Element thickness, h (used only for 2D problems). If this element is used in an axisymmetric model, the thickness must be specified as $2\pi x$ ($x = r$). In plane strain $h = 1$.
2. Mass density of the material, ρ (used only for explicit dynamic analysis).
3. Coefficient of thermal expansion, α (zero).
4. Tangential stiffness K_t.
5. Normal stiffness K_n.
6. Tensile strength, σ_t.
7. Cohesion, c (must be greater than the tensile strength).
8. Friction angle of second order asperities, ϕ_f, [degrees].
9. Specific mode I fracture energy, G_F^I.
10. Specific mode II fracture energy, G_F^{IIa}.
11. Relative value of irreversible deformation, γ. It is the ratio of irreversible crack opening and total inelastic crack opening and it defines the amount of inelastic crack opening that is irreversible, $\gamma \in < 0, 1 >$.
12. Tensile stress at the break-point for the bilinear law, s_1.
13. Crack opening displacement at the break-point for the bilinear law, sw_1.
14. Cohesion at the break-point for the bilinear law, c_1.
15. Crack sliding displacement at the break-point for the bilinear law, cw_1.
16. Asperities type: 0: Gaussian; 1: Hyperbolic.
17. Asperities parameter p_1^A equals to h_0 for Gaussian and r_0 for hyperbolic asperity.
18. Asperities parameter p_2^A equals to l_0 for Gaussian and $\mu_{\alpha 0}$ for hyperbolic asperity.
19. Critical cumulative sliding displacement for asperityy degradation w_c.
20. Normal stress at which asperity height is zero, d.
21. Penalty stiffness K_{na} when interface/Joint closure approaches a.
22. Critical value of joint overlapping a for which the penalty stiffness K_{na} is obtained.
23. Curve id defining viscosity in terms of crack opening rate (to be specified only for transient stress analysis) (units $FL^{-2}T$), Fig. 2.24.

h	ρ	α	K_t	K_n	σ_t	c	ϕ_f	G_F^I	G_F^{IIa}
γ	u_{Dmax}	s_1	sw_1	c_1	cw_1	Asp$_{id}$	p_1^A	p_2^A	w_c
d	K_{na}	a	μ_{id}						

Table 2.23: Properties for Material Model 21

s_1, sw_1, c_1 and cw_1, are only used by the program when they are positive and non-zero, which indicates that the bilinear softening law will be used. If either value is negative or zero, the linear softening law will be used.

Note:

1. If G_F is defined explicitly by a curve (sect. 1.1.7), then the value of G_F should be replaced by the negative of the curve number. Internally, Merlin will determine G_F from the absolute value of the specified curve. For help on the selection of the above material properties, refer to Sect. **??**.
2. The following relationship must be satisfied:

$$f_t' < \frac{c}{\tan\phi} \tag{2.3}$$

2.2.3.6.4 Interface/Joint Crack Model 3-Concrete-Concrete (MM: 22)

This interface/Joint model is a subset of the original ICM-1 (MM: 8) with the following differences:

1. The failure envelope is not hyperbolic, but pure Mohr-Coulomb.
2. There is no dilatancy
3. There is stiffness degradation
4. There are two different normal stiffnesses for tension and compression
5. There are two different shear stiffnesses for tension and compression

Model parameters are 13:

1. Element thickness, h (used only for 2D problems). If this element is used in an axisymmetric model, the thickness must be specified as $2\pi x$ ($x = r$). In plane strain $h = 1$.
2. Mass density of the material, ρ (used only for explicit dynamic analysis).
3. Coefficient of thermal expansion, α (zero).
4. Tangential stiffness in compression K_t^c.
5. Normal stiffness in compression K_n^c.
6. Tensile strength, σ_t (can be zero).

7. Cohesion, c (must be greater than the tensile strength).
8. Friction angle, ϕ_f, [degrees].
9. Specific mode I fracture energy, G_F^I (can be zero for brittle models).
10. Specific mode II fracture energy, G_F^{IIa} (can be zero for brittle response after shear strength is reached).
11. Curve id defining viscosity in terms of crack opening rate (to be specified only for transient stress analysis) (units $FL^{-2}T$), Fig. 2.24.
12. Shear stiffness in tension K_n^t.
13. Normal stiffness in tension K_t^t.

h	ρ	α	K_t^c	K_n^c	σ_t	c	ϕ_f	G_F^I
G_F^{IIa}	μ_{id}	K_n^t	K_t^t					

Table 2.24: Properties for Material Model 22

Note:

1. If G_F is defined explicitly by a curve (sect. 1.1.7), then the value of G_F should be replaced by the negative of the curve number. Internally, Merlin will determine G_F from the absolute value of the specified curve. For help on the selection of the above material properties, refer to Sect. **??**.
2. The following relationship must be satisfied:

$$f_t' < \frac{c}{\tan\phi} \tag{2.4}$$

2.2.3.6.5 Interface/Joint Crack Model 4-Rock-Concrete (MM: 23)

This interface/Joint model is a subset of the original ICM-1 (MM:8) with the following differences:

1. There is no dilatancy
2. There is stiffness degradation
3. There are two different normal stiffnesses for tension and compression
4. There are two different shear stiffnesses for tension and compression

Model parameters are 13:

1. Element thickness, h (used only for 2D problems). If this element is used in an axisymmetric model, the thickness must be specified as $2\pi x$ ($x = r$). In plane strain $h = 1$.

2. Mass density of the material, ρ (used only for explicit dynamic analysis).
3. Coefficient of thermal expansion, α (zero).
4. Tangential stiffness in compression K_t^c.
5. Normal stiffness in compression K_n^c.
6. Tensile strength, σ_t (can be zero).
7. Cohesion, c (must be greater than the tensile strength).
8. Friction angle, ϕ_f, [degrees].
9. Specific mode I fracture energy, G_F^I (can be zero for brittle models).
10. Specific mode II fracture energy, G_F^{IIa} (can be zero for brittle response after shear strength is reached).
11. Curve id defining viscosity in terms of crack opening rate (to be specified only for transient stress analysis) (units $FL^{-2}T$), Fig. 2.24.
12. Shear stiffness in tension K_n^t.
13. Normal stiffness in tension K_t^t.

h	ρ	α	K_t^c	K_n^c	σ_t	c	ϕ_f	G_F^I
G_F^{IIa}	μ_{id}	K_n^t	K_t^t					

Table 2.25: Properties for Material Model 23

Note:

1. If G_F is defined explicitly by a curve (sect. 1.1.7), then the value of G_F should be replaced by the negative of the curve number. Internally, Merlin will determine G_F from the absolute value of the specified curve. For help on the selection of the above material properties, refer to Sect. **??**.
2. The following relationship must be satisfied:

$$f_t' < \frac{c}{\tan\phi} \tag{2.5}$$

2.2.3.6.6 Interface/Joint Continuous Dashpot, 2D (MM 24)

This interface/Joint models a continuous dashpot along the boundary (for silent boundary conditions in the dynamic analysis).

Model parameters are 6:

1. Element thickness, h (1. for plane strain) If this element is used in an axisymmetric model, the thickness must be specified as $2\pi x$ ($x = r$). In plane strain $h = 1$.

2. Mass density of the material, ρ (zero always).
3. Coefficient of thermal expansion, α (zero always).
4. C_{tt} Normal viscosity per unit length. Usually $C_{nn} = \rho V_p$.
5. C_{nn} Tangential viscosity per unit length. Usually $C_{tt} = \rho V_s$.
6. C_{nt} Cross viscosity. Usually zero.

where $V_s = \sqrt{\frac{G}{\rho}}$ and $V_p = \frac{V_s}{\sqrt{\frac{1-2\nu}{2(1-\nu)}}}$

h	ρ	α	C_{nn}	C_{tt}	C_{nt}

Table 2.26: Properties for Material Model 24

2.2.3.6.7 Interface/Joint Continuous Dashpot, 3D (MM 25)

This interface/Joint models a continuous dashpot along the boundary (for silent boundary conditions in the dynamic analysis).

Model parameters are 9:

1. Element thickness, h (1. for plane strain).
2. Mass density of the material, ρ (zero always).
3. Coefficient of thermal expansion, α (zero always).
4. $C_{t_1t_1}$ Tangential viscosity per unit length. Usually $C_{t_1t_1} = \rho V_s$.
5. $C_{t_2t_2}$ Tangential viscosity per unit length. Usually $C_{t_2t_2} = \rho V_s$.
6. C_{nn} Normal viscosity per unit length. Usually $C_{nn} = \rho V_p$.
7. C_{nt_1} Cross viscosity. Usually zero.
8. $C_{t_1t_2}$ Cross viscosity. Usually zero.
9. C_{nt_2} Cross viscosity. Usually zero.

where $V_s = \sqrt{\frac{G}{\rho}}$ and $V_p = \frac{V_s}{\sqrt{\frac{1-2\nu}{2(1-\nu)}}}$

h	ρ	α	$C_{t_1t_1}$	$C_{t_2t_2}$	C_{nn}
C_{nt_1}	$C_{t_1t_2}$	C_{nt_2}			

Table 2.27: Properties for Material Model 25

2.2.3.7 Field Problems

2.2.3.7.1 Isotropic, Linear Heat Transfer (MM: 9)

The properties for the isotropic, linear heat transfer material model (i.e, the value of the material model identifier is 9) are as follows:

1. Element thickness, h.
2. Mass density of the material, ρ.
3. Specific heat, c.
4. Conductivity, k.

For axisymmetric idealizations the program automatically assigns a value for the element thickness; the user cannot override this value. For 3-D continuum idealizations the program ignores the element thickness, but a value for it must be provided in the input file.

h	ρ	c	k

Table 2.28: Properties for Material Model 9

2.2.3.7.2 Isotropic, Linear Seepage Flow (MM: 10)

The properties for the isotropic, linear seepage flow material model (i.e, the value of the material model identifier is 10) are as follows:

1. Element thickness, h.
2. Mass density of the material, ρ.
3. Unknown quantity, c.
4. Permeability, k.

For axisymmetric idealizations the program automatically assigns a value for the element thickness; the user cannot override this value. For 3-D continuum idealizations the program ignores the element thickness, but a value for it must be provided in the input file.

h	ρ	c	k

Table 2.29: Properties for Material Model 10

2.2.3.7.3 2-D Orthotropic, Linear Heat Transfer (MM: 11)

The properties for the 2-D orthotropic, linear seepage flow material model (i.e, the value of the material model identifier is 11) are as follows:

1. Element thickness, h.
2. Mass density of the material, ρ.
3. Specific heat, c.
4. Conductivity in material direction 1, k_1.
5. Conductivity in material direction 2, k_2.
6. Angle of orientation for material direction 1, θ.

This material model is not supported for 1-D elements or 3-D continuum elements. For axisymmetric idealizations the program automatically assigns a value for the element thickness; the user cannot override this value. The angle of orientation is measured in degrees from the x-axis and is considered to be positive in the counter-clockwise direction.

h	ρ	c	k_1	k_2	θ

Table 2.30: Properties for Material Model 11

2.2.3.7.4 2-D Orthotropic, Linear Seepage Flow (MM: 12)

The properties for the 2-D orthotropic, linear seepage flow material model (i.e, the value of the material model identifier is 12) are as follows:

1. Element thickness, h.
2. Mass density of the material, ρ.
3. Storativity, c.
4. Permeability in material direction 1, k_1.
5. Permeability in material direction 2, k_2.
6. Angle of orientation for material direction 1, θ.

This material model is not supported for 1-D elements or 3-D continuum elements. For axisymmetric idealizations the program automatically assigns a value for the element thickness; the user cannot override this value. The angle of orientation is measured in degrees from the x-axis and is considered to be positive in the counter-clockwise direction.

h	ρ	c	k_1	k_2	θ

Table 2.31: Properties for Material Model 12

2.2.3.7.5 3-D Orthotropic, Linear Heat Transfer (MM: 13)

The properties for the 3-D orthotropic, linear seepage flow material model (i.e, the value of the material model identifier is 13) are as follows:

1. Element thickness, h.
2. Mass density of the material, ρ.
3. Specific heat, c.
4. Conductivity in material direction 1, k_1.
5. Conductivity in material direction 2, k_2.
6. Conductivity in material direction 3, k_3.
7. x-component of a vector defining material direction 1, v_x^1.
8. y-component of a vector defining material direction 1, v_y^1.
9. z-component of a vector defining material direction 1, v_z^1.
10. x-component of a vector defining material plane 1-2, v_x^2.
11. y-component of a vector defining material plane 1-2, v_y^2.
12. z-component of a vector defining material plane 1-2, v_z^2.

This material model is not supported for 1-D elements or 2-D continuum elements. For 3-D continuum idealizations the program ignores the element thickness, but a value for it must be provided in the input file. The direction vectors $\mathbf{v}^1$ and $\mathbf{v}^2$ are not required to be unit vectors; the program utilizes them when computing the transformation matrix for the material "stiffness" matrix.

h	ρ	c	k_1	k_2	k_3	v_x^1	v_y^1
v_z^1	v_x^2	v_y^2	v_z^2				

Table 2.32: Properties for Material Model 13

2.2.3.7.6 3-D Orthotropic, Linear Seepage Flow (MM: 14)

The properties for the 3-D orthotropic, linear seepage flow material model (i.e, the value of the material model identifier is 14) are as follows:

1. Element thickness, h.
2. Mass density of the material, ρ.
3. Storativity, c.
4. Permeability in material direction 1, k_1.
5. Permeability in material direction 2, k_2.
6. Permeability in material direction 3, k_3.
7. x-component of a vector defining material direction 1, v_x^1.
8. y-component of a vector defining material direction 1, v_y^1.
9. z-component of a vector defining material direction 1, v_z^1.
10. x-component of a vector defining material plane 1-2, v_x^2.
11. y-component of a vector defining material plane 1-2, v_y^2.
12. z-component of a vector defining material plane 1-2, v_z^2.

This material model is not supported for 1-D elements or 2-D continuum elements. For 3-D continuum idealizations the program ignores the element thickness, but a value for it must be provided in the input file. The direction vectors $\mathbf{v}^1$ and $\mathbf{v}^2$ are not required to be unit vectors; the program utilizes them when computing the transformation matrix for the material "stiffness" matrix.

h	ρ	c	k_1	k_2	k_3	v_x^1	v_y^1
v_z^1	v_x^2	v_y^2	v_z^2				

Table 2.33: Properties for Material Model 14

2.2.4 AARProp

The `AARProp` option enables the user to define the parameters associated with an AAR expansion simulation.

Following the `AARProp` keyword, the number N of element groups affected by AAR must be specified, then for each group, data should be defined as:

2.2.4.1 Saouma and Perotti

1. ID_{group} Group ID.
2. $\varepsilon^{\infty}|_{T=T_0^{test}}$ Maximum volumetric strain at temperature T_0^{test}.
3. $\tau_C(\theta_0^{test})$ Characteristic time at temperature $\theta_0^{test} = 273 + T_0^{test}$.
4. $\tau_L(\theta_0^{test})$ Latency time at temperature $\theta_0^{test} = 273 + T_0^{test}$.

5. U_C Activation energy associated with $\tau_C(\theta_0^{test})$ (suggested value 5,400 K).
6. U_L Activation energy associated with $\tau_L(\theta_0^{test})$ (suggested value 9,400 K).
7. Γ_r Residual reduction factor for Γ_t.
8. γ_t Fraction of f_t prior to reduction of AAR expansion due to macro cracking.
9. f_c Compressive strength (must be negative).
10. f_t Tensile strength.
11. a Γ_2 Shape parameter (0. for straight line).
12. T_0^{Test} Reference temperature (^{o}C) of tests for τ_L and τ_C.
13. σ_U Upper compressive stress beyond which there is no more AAR expansion; must be negative.
14. β_E Reduction fraction for Young's Modulus when AAR reaction ends.
15. β_f Reduction faction for tensile strength when AAR reaction ends.

Note:

1. If a linear time dependant expansion is desired (instead of the sigmoid), then τ_C and τ_L should be assigned very large and small values respectively (such as 10,000.0 and 0.1; However care should be exercised in selecting those values).
2. If the effect of temperature is to be neglected, than U_C and U_L should be set to zero.

Note: for β_E, and β_f to be active, there has to be a *DeltaTime* specified in the corresponding load increment (it can be set to zero if there is no expansion).

Nodal temperature must be specified in either of two modes:

1. Use of `InitialTemp` which assigns same nodal temperature to all the nodes through towlines in the control part.
2. Specify `Temperature` in the first increment for all the nodes.

Failure to specify temperature in AAR analysis, will give wrong results (Merlin will then assume that the temperature is zero).

2.2.4.2 Charlwood

1. ID_{group} Group ID
2. $\varepsilon^{\infty}|_{T=T_0^{test}}$ Maximum volumetric strain at temperature T_0^{test}.
3. $\tau_C(\theta_0^{test})$ Characteristic time at temperature $\theta_0^{test} = 273 + T_0^{test}$.
4. $\tau_L(\theta_0^{test})$ Latency time at temperature $\theta_0^{test} = 273 + T_0^{test}$.

5. U_C Activation energy associated with $\tau_C(\theta_0^{test})$ (suggested value 5,400 K).
6. U_L Activation energy associated with $\tau_L(\theta_0^{test})$ (suggested value 9,400 K).
7. -1.0 Code to switch to Charlwood's model
8. 0. Not used
9. f_c Compressive strength of concrete (must be negative; can not be left as zero)
10. 0. Not used
11. σ_L Compressive stress for which logarithmic reduction begins
12. T_0^{Test} Reference temperature (^{o}C) of tests for τ_L and τ_C.
13. σ_U Upper compressive stress beyond which there is no more AAR expansion must be negative.
14. β_E Reduction fraction for Young's Modulus when AAR reaction ends.
15. β_f Reduction faction for tensile strength when AAR reaction ends.

2.2.5 EndMesh

The `EndMesh` option is used to inform the program that all mesh definition options required to perform the analysis have been specified. This keyword must appear at the end of the mesh definition block. Failure to include an `EndMesh` option will prohibit the program from recognizing the next keyword appearing in the input file, which will cause the program to print an error message to the formatted output file and terminate execution prematurely.

2.3 Nodal Definition

2.3.1 Coordinates

The `Coordinates` option enables the user to define coordinates for each node. The line in the input file following the keyword contains the number of nodes. The number of nodes must be greater than zero, otherwise the program will print an error message to the formatted output file and program execution will be terminated prematurely. The coordinates of each node appearing in an element connectivity definition in the input file must be defined explicitly in the input file; no generation capabilities are included in the program. The coordinates of mid-side nodes are assigned automatically by the program. Coordinates are defined as x-y pairs for two-dimensional analysis, r-z pairs for axisymmetric analysis, and x-y-z triplets for three-dimensional analysis. Only as many coordinates as are required by the elements used in the mesh are entered. Therefore, the ElementGroup option must precede this option in the mesh definition block or the program will print an error message to the formatted output file and terminate execution prematurely. The node number precedes the coordinates and is checked for consistancy (i.e. the tenth node must be node number 10). Discovery of an inconsistant node number is an error, which will cause the program to print an error message to the formatted output file and terminate execution prematurely. Failure to include an `Coordinates` option is

a fatal input error, which will cause the program to print an error message to the formatted output file and terminate execution prematurely.

Node No.	Coordinate 1	Coordinate 2	[Coordinate 3]

Table 2.34: Nodal Coordinate Input

2.3.2 Master/Slave

The `Master/Slave` option enables the user to define pairs of duplicate nodes. Duplicate nodes are pairs of nodes that have identical coordinates and primary field variables, but different intermediate and flux-like field variables. They are used primarialy to model discontinuities in the intermediate and flux-like fields at the interface/Joint between two materials. Duplicate nodes are required when materials with different intermediate and flux-like field variables (e.g., the FCM and isotropic, linearly elastic material models) or different idealizations (e.g., plane stress and plane strain) are on opposite sides of the interface, otherwise the nodal values at the shared nodes for intermediate and flux-like field variables are meaningless. The program prints a warning message for each node that is shared by different element groups with the same idealization. When a node is shared by different element groups with different idealizations the program prints an error message to the formatted output file and terminates execution prematurely.

The line in the input file following the keyword contains the number of duplicate node pairs. The number of duplicate node pairs must be greater than zero, otherwise the program will print an error message to the formatted output file and program execution will be terminated prematurely. Each pair of duplicate nodes must be explicitly defined in the input file; no generation capabilities are included in the program. If all corner nodes of an element surface are duplicate nodes, the program will automatically insert duplicate node pairs for the mid-side nodes. When defining pairs of duplicate nodes, the master node appears first and the slave node appears second. There is no convention that defines which node of a pair should be the master node or the slave node. However, consistency should be maintained when defining the duplicate node pairs: all nodes on a particular side of an interface/Joint should be either master nodes or slave nodes, not a combination of master and slave nodes. This is especially true when using duplicate nodes with higher order elements or to connect interface/Joint elements to continuum elements. When using duplicate nodes with higher order elements the convention established by the user in the input file is also used in the program for the inserted duplicate nodes and when connecting interface/Joint elements to continuum elements the program uses the duplicate node pairs to identify nodes subject to constrains. Failure to include a `Master/Slave` option indicates that there are no duplicate nodes in the mesh.

Master Node No.	Slave Node No.

Table 2.35: Master/Slave Nodes Input

2.3.3 ReinfRods

The `ReinfRods` option enables the user to define axial members crossing the mesh (typically, steel reinforcement).

Merlin will internally determine which continuum elements are crossed by the reinforcement, and properly adjust the stiffness matrix. Perfect bond is assumed between the reinforcement and the continuum.

The line in the input file following the keyword contains the number of reinforcing rods. This number must be greater than zero. Reinforcing rods should have element type 1 for 2D problems, and element type 44 for 3D problems. Constitutive models 4 (J_2 plasticity with radial return) is suggested.

		First end			*Second end*		
Reinf. No.	Elem. Group	Coord. 1	Coord. 2	[Coord. 3]	Coord. 1	Coord. 2	[Coord. 3]

Table 2.36: Reinforcement Input

2.3.4 NodalMass

The `NodalMass` option enables the user to specify nodal masses for a dynamic analysis. **Note, this option has no effect on static analysis**. `NodalMass` (Sect. 3.4.1.8.3) option should be used for dynamic analysis.

The line in the input file following the keyword contains the number of nodes for which nodal masses are specified. The number of nodes for which nodal masses are specified must be greater than zero, otherwise the program will print an error message to the formatted output file and execution will be terminated prematurely. Following the line containing the number of nodes for which nodal masses are specified there must be one line for each node and its specified mass.

Each line includes two pieces of information defining the nodal mass:

1. Node number
2. Nodal mass

Note:

1. The `NodalMass` might be generated from Kumo to account for the water using Westergaard's or Zangar's method.
2. For static analysis, one should use the `Westergaard` or `Zangar` option.

2.3.5 NodalDamping

Defines nodal damper boundary condition. The first integer following the keyword is the total number of nodal dashpots.

Then for each nodal dashpot there will be three entries, Fig. 2.37 where $F = C\dot{u}$

It should be noted that this option is used to connect a node to a rigid support through a dashpot (and not two separate nodes).

Node No.	DOF No.	Damping C

Table 2.37: Nodal Damping Input

2.3.6 NodalSprings

Defines nodal spring boundary condition. The first integer following the keyword is the total number of nodal springs.

Then for each nodal spring there will be three entries, Fig. 2.38 where $F = K\dot{u}$ It should be

Node No.	DOF No.	Stiffness K

Table 2.38: Nodal Damping Input

noted that this option is used to connect a node to a rigid support through a spring. (and not two separate nodes).

2.4 Crack Definitions

2.4.1 CrackSurface

The `CrackSurface` option enables the user to define the topology of the crack surfaces for each discrete crack.

This is needed only if uplift pressures are to be applied on the crack surfaces, or if crack information (such as COD, CSD) are to be evaluated, or contour line integrals for stress intensity factor calculation are to be used.

The line in the input file following the keyword contains the number of discrete cracks. The number of discrete cracks must be greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. The topology of each crack is defined separately, with the first line of the topology definition being the number of element pairs on the crack surface. The number of element pairs on the crack surface must be greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. One line is required for each element pair and the contents of the line is as follows:

1. Element number of the element on the upper surface,
2. Element surface number of the element on the upper surface,
3. Element number of the element on the lower surface, and
4. Element surface number of the element on the lower surface.

Conventions for the upper and lower surfaces of a discrete crack are discussed in Section **??**. Consistency checks are performed on the input data to insure that the element numbers and the element surface numbers are valid. Both the `ElementGroup` and `Connectivity` options

must precede this option in the mesh definition block, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. Specifying invalid element numbers and element surface numbers and incompatible element surfaces (i.e. surfaces with different numbers of nodes) are also errors which cause the program to print an error message to the formatted output file and terminate execution prematurely. For a LEFM analysis using the displacement correlation method or a FCM analysis it is necessary to identify only that portion of the crack surface that is adjacent to the crack front. A LEFM analysis using a contour integral method (see Sections 1.4.1.1 and 1.4.1.2) requires that the crack surface be defined from the crack front to the outer-most contour path with the surface pairs listed in that order. However, it is desirable to define the topology for the full length of the crack with those element pairs adjacent to the crack front appearing at the end of the list of element pairs. Failure to include a `CrackSurface` option is a fatal input error if the `LEFM` or `FCM` options have been included in the program control block. If a `CrackSurface` option is required but not included, an error message will be printed to the formatted output file and execution will be terminated prematurely.

Upper Element No.	Surface No.	Lower Element No.	Surface No.

Table 2.39: Crack Surface Topology Input

Note: It is not essential to define `CrackSurface` in NLFM analysis. However, this option must be used if `PrintCrack` will be used.

2.4.2 CrackFronts

The `CrackFronts` option enables the user to define the nodes that are located along the crack fronts for each discrete crack.

This option is needed only if stress intensity factors are to be computed.

The line in the input file following the keyword contains the number of discrete cracks. The number of discrete cracks must be greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. The crack front nodes for each crack are defined separately, with the first line of the crack front nodes definition being the number of nodes on the crack front. The number of nodes on the crack front must be greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. Failure to include a `CrackFronts` option is a fatal input error if the `LEFM` or `FCM` options have been included in the program control block. If a `CrackFronts` option is required but not included, an error message will be printed to the formatted output file and program execution will be terminated prematurely.

Crack Front Node 1	...	[Crack Front Node n]

Table 2.40: Crack Front Node List Input

2.4.3 ContourPath

The `ContourPath` option enables the user to define the paths to be used with contour integral methods for computation of stress intensity factors (see Sections 1.4.1.1 and 1.4.1.2). Contour paths are defined around a crack front using element surfaces (see Section **??**). The line in the input file following the keyword contains the number of contour paths. The number of contour paths must be greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. Each contour path is defined separately, with the first line of the contour path definition being the number corresponding to the crack surface about which the contour path is defined. The next line contains the number of element surfaces defining the contour path. The number of element surfaces defining the contour path must be greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. One line is required for each element surface on the contour path and the contents of the line is as follows:

1. Element number of the element on the contour path and
2. Element surface number of the element on the contour path.

Element surfaces are defined tip to tail in the counter-clockwise direction around the contour path (i.e., starting at the upper crack surface and ending at the lower crack surface). Failure to include a `ContourPath` option is a fatal input error if the `LEFM` and either the `J-integral` or `S-integral` options have been included in the program control block. If a `ContourPath` option is required but not included, an error message will be printed to the formatted output file and program execution will be terminated prematurely.

Element No.	Surface No.

Table 2.41: Contour Path Input

Chapter 3

INCREMENTAL LOAD BLOCK

Incremental load blocks appear at the end of the MERLIN input file. One block is required for each increment specified in the `Increments` option of the program control block (see Section 1.1.8). It enables the user to define the applied incremental loads and incremental displacement boundary conditions as well as convergence tolerances for the equilibrium iterations. Incremental load options are selected using keywords of the form described in Section **??** and can appear in any order within the incremental load block. The keywords associated with the options available in the incremental load block along with a brief description of their functionality are given in Table 3.1. A section of this chapter is devoted to a more detailed description of each of these options.

3.1 General

3.1.1 Comment

The `Comment` option enables the user to enter a one-line comment in the input file. The text of the comment appears on the same line as the keyword, with the keyword and the text being separated by one or more blank spaces. The keyword and the comment text may extend beyond column 80 of the input file, but only those characters that appear in columns 1 through 80 will be echoed to the formatted output file. `Comment` options can be placed in the file back-to-back for multi-line comments.

3.1.2 TotalMatForm

The `TotalMatForm` switches the local load increment to a total formulation ($\sigma^n = E_{sec}(\varepsilon^{n-1} + d\varepsilon^n)$ instead of the default incremental one $\sigma^n = \sigma^{n-1} + E_t d\varepsilon^n$).

This is particularly useful in modeling creep through the equivalent young modulus and creep coefficient (in conjunction with the change of material group).

3.1.3 ZerroDispl

The `ZerroDispl` reset all displacements to zero at the beginning of the current increment. Stresses are not affected.

Options	Functionality
	General
`Comment`	Insert a one-line comment in the input file
`TotalMatForm`	Switch to a total formulation for the current load increment
`ZerroDispl`	Reinitialize the displacements (but not the stresses) to zero
`ElementGroup`	Modify the constitutive model parameters within a load increment
`SaveReacts`	Save the nodal reactions of user selected nodes/dof
`Reacts2Loads`	Transfer the saved ractions as nodal loads
`SuppressPost`	Suppress printing of results for increment to post file
`EigenFile`	Specifies name of the output file for graphical display of the eigenmodes
`EndIncrement`	Signal end of incremental load options
	Iteration Options
`Iterations`	Specify maximum number of iterations
`EnergyError`	Specify relative convergence tolerance in terms of energy
`RelResidErr`	Specify relative convergence tolerance in terms of residual forces
`RelResidErr`	Specify absolute convergence tolerance in terms of residual forces
`DispError`	Specify relative convergence tolerance in terms of displacements
	Solution Options
`TangentStiff`	Assemble global tangent stiffness matrix
`LineSearch`	Use line search acceleration with iterative procedure
`SecantNewton`	Use secant Newton update with iterative procedure
`ArcLength`	Use spherical arc-length method for step size control
`SpecifyCOD`	Use relative displacement method for step size control
`LoadScale`	Use constraint scaling method for step size control
`NoLoadScale`	Do not use constraint scaling method for step size control
	Loading Options
`DispBCs`	Define incremental displacement boundary conditions
`PointLoads`	Define incremental point loads
`Tractions`	Define incremental surface tractions
`UTRACT`	Use user subroutine to define non-constant surface tractions
`AARLoad`	Define internal AAR expansion
`AARMaxMod`	Modify maximum AAR induced volumetric strain
`Hydrostatic`	Define hydrostatic load on an element edge
`Westergaard`	Define pseudo-hydrodynamic load using Westergaard's method
`Zangar`	Defines pseudo-hydrodynamic load using Zangar's method
`Uplift`	Define uplift along cracked and uncracked interfaces in 2D
`Uplift2`	Define uplift along cracked and uncracked interfaces in 3D
`FulUplNegCod`	Apply full uplift if joint has failed with -ve COD
`DynamUplift`	Describe the uplift model during dynamic analysis
`BodyForces`	Define incremental acceleration vector for body forces
`BodyFrcList`	Define material groups subjected to body forces at a given increment
`UBODY`	Use user subroutine `ubody` to define non-constant body forces
`EigenModes`	Determines the eigenvalues and eigenmodes for a given load increment
`EigenFile`	Define the filename for the external viewer of the mode shapes.
`Velocities`	Define incremental velocity boundary conditions
`Acceleration`	Define incremental acceleration boundary conditions
`Centrifugal`	Define incremental axis of rotation and angular velocity for centrifugal forces
	Seepage/Heat Transfer
`Pressures`	Define incremental hydrostatic pressures for stress analyses
`Temperatures`	Define incremental temperatures for stress and heat transfer analyses
`PointTemps`	Define a spatial temperature distribution from which nodal temperature will be determined
`TempVarElev`	Defines two nodal temperature based on node coordinates
`PointFluxes`	Defines incremental point fluxes
`Fluxes`	Defines incremental fluxes on element surfaces
`FluxVarElev`	Defines two nodal fluxes depending on node coordinates
`UFLUX`	Use user subroutine `uflux` to define non-constant fluxes
`Films`	Define incremental convective heat fluxes on element surfaces
`UFILM`	Use user subroutine `ufilm` to define non-constant convective heat fluxes
`BodyHeat`	Define incremental internally generated heat
`UHEAT`	Use user subroutine `uheat` to define non-constant internally generated heat
`Heads`	Define incremental hydraulic boundary conditions

Table 3.1: Incremental Load Options

This is a particularly useful command in dynamic analysis where effect of initial static analysis must be accounted for (stresses), but only those displacements caused by the dynamic excitation are of interest.

3.1.4 ElementGroup

The `ElementGroup` option allos the modification of the constitutive model parameters within a load increment. This is particularly useful to simulate staged construction/excavation or creep.

Following the tag, an integer identifies the number of element groups to be changed.

Following this is line the model parameters is to be defined identically as in Section 2.2 with the exception that the first entry is an integer identifying the element group id.

3.1.5 SaveReacts

Save the nodal reactions of user selected nodes and degrees of freedom into an array for subsequent application as nodal loads. This is most useful in dynamic analysis (which includes gravity load) where we start with a constrained static analysis, to be followed by a dynamic one with no constraints.

Following the `SaveReacts` keyword, an integer specifying the total number of reaction forces to be released must be specified. Then, for each reaction, user must specify:

Node id	DOF No. (1-3)

Table 3.2: Nodal Reaction Forces to Be Saved

3.1.6 Reacts2Loads

This single keyword, when inserted inside a load increment, will apply the previously saved nodal reactions (through `SaveReacts`) as a nodal forces in the current increment.

3.1.7 SuppressPost

The `SuppressPost` option enables the user to suppress the writing of computed results for the current increment to the unformatted output file.

Note, this option can not be used with the first increment.

For analyses with many increments the unformatted output file can be quite large and in any cases results will be reviewed at only a few key increments. This option allows the user to selectively write to the unformatted output file the results for only those increments that are of interest. However, the user should be aware that an analysis may only be restarted (see Section 1.1.3) at increments that immediately follow those included in the unformatted output file. If the `SuppressPost` option does not appear in the incremental load block the computed results for the current increment will automatically be written to the unformatted output file.

Note: Windows does not support files greater than 2.1 GB. Hence, if the generated .pst file reaches this limit, there will be an abnormal interruption (without warning) of the program.

3.1.8 EndIncrement

The `EndIncrement` option is used to inform the program that all incremental load options required to perform the analysis have been specified. This keyword must appear at the end of the incremental load block. Failure to include an `EndIncrement` option will cause the program to continue reading incremental load information from the input file until the end-of-file is encountered, which will cause the program to print an error message to the formatted output file and terminate execution prematurely.

3.2 Iterations Control

3.2.1 Iterations

The `Iterations` option enables the user to specify the maximum number of iterations to be performed for the modified-Newton algorithm (**?**). The line in the input file following the keyword contains the maximum number of iterations to perform. The number of iterations must be greater than or equal to zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. The number of iterations required for convergence depends on the presence of sharp corners or cracks in the mesh, the level of mesh discretization, and the specified convergence tolerances. For a linear analysis with no sharp corners or cracks using the mixed-iterative method, convergence should be reached within two to three iterations provided the mesh discretization is satisfactory. For a linear analysis with sharp corners or cracks using the mixed-iterative method, convergence should be reached within three to five iterations provided the mesh discretization is satisfactory. For a nonlinear analysis, convergence should be reached in five to thirty iterations provided the mesh discretization is satisfactory. In any case, an excessive number of iterations typically indicates the need for a denser mesh, particularly in areas of high primary field variable gradients such as would be present near a sharp corner or crack tip. For each iteration, a summary of the error measures and the specified convergence tolerances are printed in the formatted output file (see Section 4.1). When convergence is reached, a message is printed to the formatted file indicating that this is the case. If the `Iterations` option is omitted from the incremental load block, the number of iterations will default to a value of zero (i.e., the modified-Newton algorithm is disabled), which is acceptable for linear analyses using the displacement method.

3.2.2 EnergyError

The `EnergyError` option enables the user to specify the convergence tolerance for the energy error. The energy error is the ratio of the absolute values of the external work done by the applied incremental loads and the residual loads for the current increment. The line in the input file following the keyword contains the convergence tolerance, which is a floating point number that is greater than zero but less than or equal to one. Typically, values between 0.001 and 0.005 are satisfactory for most problems. If the `EnergyError` option is omitted from the incremental load block, the energy error tolerance will default to a value of one.

3.2.3 RelResidErr

The `RelResidErr` option enables the user to specify the convergence tolerance for the relative residual error. The relative residual error is ratio of the Euclidean norms of the current and initial residual load vectors. The line in the input file following the keyword contains the convergence tolerance, which is a floating point number that is greater than zero but less than or equal to one. Typically, a value of 0.1 is satisfactory for most problems. If the `RelResidErr` option is omitted from the incremental load block, the relative residual error tolerance will default to a value of one.

3.2.4 AbsResidErr

The `AbsResidErr` option enables the user to specify the convergence tolerance for the absolute residual error. The absolute residual error is the infinity norm of the residual load vector. The line in the input file following the keyword contains the convergence tolerance, which is a floating point number that is greater than zero but significantly less than the value of the maximum reaction force component. Typically, a value that is ten percent of the maximum reaction component is satisfactory for most problems. This error tolerance is most useful in stress analyses where the only loads are due to thermal gradients in the structure. If the `AbsResidErr` is omitted from the incremental load block, the absolute residual error tolerance will default to a large positive value.

3.2.5 DispError

The `DispError` option enables the user to specify the convergence tolerance for the displacement error. The displacement error is ratio of the Euclidean norms of the iterative displacement correction and incremental displacement vectors. The line in the input file following the keyword contains the convergence tolerance, which is a floating point number that is greater than zero but less than or equal to one. Typically, a value of 0.01 is satisfactory for most problems. If the `DispError` option is omitted from the incremental load block, the displacement error tolerance will default to a value of one.

3.3 Solution

A summary of the solution options compatibility is shown in Table 3.3

	Initial Stiffness	Tangent Stiffness	Secant Newton	Line Search
Line Search		√	x	x
Arc Length		√	x	x
Specify COD		√	x	x
Load Scale		√	x	x

Table 3.3: Compatibility of Various Non-Linear Options

3.3.1 Incremental Analyses

The default option is the the initial elastic stiffness where the same initial stiffness of increment one, iteration one is used throughout the analyses for all increments.

The advantage of this method is there is no update of the stiffness matrix, hence there will be no problem associated with unloading (and resulting negative terms along the diagonal). However, if this default option is used, large number of iterations should be specified.

3.3.1.1 TangentStiff

The `TangentStiff` option enables the user to invoke the assembly of the global stiffness matrix using tangent material stiffness matrices at the element level. Currently, the Fictitious Crack Model and the Interface Crack Model support a tangent material stiffness matrix. Specifically, the `TangentStiff` option should be used when the fracture process zone is unloading (i.e., the crack is closing), as the rate of convergence is greatly improved.

This is practically identical to a full Newton-Raphson, in which the stiffness matrix is updated at each iteration. However, when softening (resulting in negative terms) or dilatancy (resulting in unsymmetric terms), then the previous stiffness matrix is kept intact. Otherwise, a rank one update is performed.

If the `TangentStiff` option is omitted from the incremental load block, the global stiffness matrix will be formed using elastic material stiffness matrices.

3.3.1.2 SecantNewton

The `SecantNewton` option enables the user to accelerate the convergence of the modified-Newton algorithm by performing a rank-one update of the stiffness matrix (**?**).

Since nonlinear analyses cannot be performed for heat transfer or seepage flow analyses, the `SecantNewton` option has not been implemented for use with either heat transfer or seepage flow analyses. If the `SecantNewton` option is requested for a heat transfer or seepage flow analysis, the request will be ignored and a warning message will be printed to the formatted output file.

Rank-one updates are especially useful in nonlinear analyses where convergence of the modified-Newton algorithm is relatively slow, as is the case when large step sizes are used or a peak load is being approached. Rank-one updates may be useful when using the mixed-iterative method since the updated stiffness matrix may be much closer to the mixed stiffness matrix than the displacement method stiffness matrix. However, it should not be used in linear analyses with the mixed-iterative method in an attempt to accelerate the rate of convergence for poorly discretized meshes, as the modified-Newton algorithm might fail to converge to the specified tolerances regardless of the number of iterations.

The rank-one update is implemented in such a manner that an explicit update of either the stiffness matrix or its inverse is never performed; an update formula for the iterative displacement corrections involving a small number of vector operations is used (**?**). This approach allows the rank-one update to be used with softening materials in the post-peak regime without having to worry about the update producing a non-positive definite stiffness matrix, provided the initial stiffness matrix for the increment is positive definite. The rank-one update is also implemented such that it can be used together with the `LineSearch` option. The user should be aware that

using the `SecantNewton` and `LineSearch` options together does not guarentee fewer iterations than using either option alone. If the `SecantNewton` option is omitted from the incremental load block, no line searches will be performed for the current increment.

3.3.2 LineSearch

The `LineSearch` option enables the user to accelerate the convergence of the modified-Newton algorithm by performing a line search (**?**). Since nonlinear analyses cannot be performed for heat transfer or seepage flow analyses, the `LineSearch` option has not been implemented for use with either heat transfer or seepage flow analyses. If the `LineSearch` option is requested for a heat transfer or seepage flow analysis, the request will be ignored and a warning message will be printed to the formatted output file.

Line searches are especially useful in nonlinear analyses where convergence of the modified-Newton algorithm is relatively slow, as is the case when large step sizes are used. Line searches should not be used in linear analyses with the mixed-iterative method in an attempt to accelerate the rate of convergence for poorly discretized meshes, as the modified-Newton algorithm might fail to converge to the specified tolerances regardless of the number of iterations. Situations in which the line search is not working properly are easily recognized by observing the values of the error measures printed in the formatted output file: they will appear to be converging and then suddenly diverge only to appear to converge again before suddenly diverging. If enough iterations are requested this pattern will occur repeatedly, with the error measures never decreasing below a particular value (i.e., the error due to the mesh discretization), which is unfortunately greater than the specified tolerances. If the line search should fail in this manner, the applied loads should be changed and, contrary to intuition, increasing the load sometimes achieves better results than decreasing the load.

The line search is implemented in such a manner that it is only performed for iterations where the errors are large; when the errors are relatively small, the request for a line search is ignored by the program. If the `LineSearch` option is omitted from the incremental load block, no line searches will be performed for the current increment.

3.3.3 Indirect Control Methods

One of the following three indirect control methods can be used:

1. Arc-Length
2. Specified COD/CMD (crack opening displacement)
3. Load scale

The default control is the traditional load/displacement control.

For any of the following indirect control methods, MERLIN will output both the incremental and total load scale factors.

3.3.3.1 Arc-Length

The `Arc-Length` option enables the user to control the step size for the modified-Newton algorithm through a spherical arc-length method (**?**, **?**). Since nonlinear analyses cannot be

performed for heat transfer or seepage flow analyses, the `ArcLength` option has not been implemented for use with either heat transfer or seepage flow analyses. If the `ArcLength` option is requested for a heat transfer or seepage flow analysis, the request will be ignored and a warning message will be printed to the formatted output file.

This option can not be used with imposed displacements.

The arc-length is the Euclidean norm (i.e. the "length") of the incremental displacement vector. This norm is printed in the formatted output file as part of the summary of the iterative error measures when the `ArcLength` option is included in the incremental load block. The line in the input file following the keyword contains the prescribed arc-length, which is a floating point number that is greater than zero. The arc-length method is generally used with proportional loadings, but it can also be used with arbitrary loadings by adjusting the arc-lengths such that the changes in the loading occur at the proper instances. A load factor, which indicates the magnitude of a proportional loading, is introduced to scale the applied incremental loads for each iteration such that the Euclidean norm of the incremental displacement vector is equal to the specified arc-length. Incremental load factors are accumulated to obtain a total load factor, which is meaningful only for proportional loadings, and both the incremental and total load factors are printed to the formatted output file at the end of each increment. The `ArcLength` option can be used in the same analysis with either the `SpecifyCOD` or `LoadScale` options to simulate a change in the loading, but not in the same increment as it would be virtually impossible to satisfy all the step size constrains simultaneously.

The arc-length method is especially useful for capturing snap-back or snap-through behavior in nonlinear analyses and may be used in conjunction with the `LineSearch` option but not the `SecantNewton` option. It is also useful for nonlinear analyses with hardening materials such as J_2 plasticity (see Section 2.2.3.4.1), where the arc-length for the nonlinear increments is a fraction of the arc-length required to reach the onset of nonlinearity. For analyses with snap-back or snap-through the arc-length might require adjustment during the analyses to capture this behavior, but for analyses with hardening it may be possible to use the same arc-length for all nonlinear increments. If the `ArcLength` option is omitted from the incremental load block, the incremental loads defined in the input file are applied unscaled.

3.3.3.2 SpecifyCOD

The `SpecifyCOD` option enables the user to control the step size for the modified-Newton algorithm by specifying a relative displacement component between two nodes (**?**, **?**). Since nonlinear analyses cannot be performed for heat transfer or seepage flow analyses, the `SpecifyCOD` option has not been implemented for use with either heat transfer or seepage flow analyses. If the `SpecifyCOD` option is requested for a heat transfer or seepage flow analysis, the request will be ignored and a warning message will be printed to the formatted output file.

The line in the input file following the keyword contains pieces up to six pieces of information:

1. Magnitude of the prescribed relative displacement component, $\Delta u'$,
2. Node number for the reference node,
3. Node number for the "moving" node,
4. First component v_x of the vector $\mathbf{v}$ defining the direction of the relative displacement component,

5. Second component v_y of the vector **v** defining the direction of the relative displacement component, and
6. Third component v_z of the vector **v** defining the direction of the relative displacement component.

The magnitude of the prescribed relative displacement component and the direction vector components are floating point numbers and the node numbers are integers. The relative displacements (in the global coordinate system) are defined by subtracting the displacements for the "moving" node from the displacements for the reference node. The direction vector **v** is not required to be a unit vector; the program utilizes it before computing the relative displacement component. For two-dimensional analyses the third component v_z of the vector **v** is not input; it is only input for three-dimensional analyses.

$\Delta u'$	Reference Node No.	"Moving" Node No.	v_x	v_y	$[v_z]$

Table 3.4: SpecifyCOD Input

This method of step size control is generally used with proportional loadings, but it can also be used with arbitrary loadings by adjusting the relative displacement magnitudes such that the changes in the loading occur at the proper instances. A load factor, which indicates the magnitude of a proportional loading, is introduced to scale the applied incremental loads for each iteration such that the magnitude of the relative displacement component is equal to the prescribed value. Incremental load factors are accumulated to obtain a total load factor, which is meaningful only for proportional loadings, and both the incremental and total load factors are printed to the formatted output file at the end of each increment. The `SpecifyCOD` option can be used in the same analysis with either the `ArcLength` or `LoadScale` options to simulate a change in the loading, but not in the same increment as it would be virtually impossible to satisfy all the step size constrains simultaneously.

This method of step size control is especially useful for capturing snap-back or snap-through behavior in nonlinear analyses and may be used in conjunction with the `LineSearch` option but not the `SecantNewton` option. It is particularly useful for nonlinear analyses of notched structures made of softening materials such as those modeled using Drucker-Prager plasticity (see Section 2.2.3.4.2). It can also be used to simulate displacement controlled unloading and reloading, such as that associated with an unload/reload cycle in an experiment. If this type of behavior is being simulated for a structure with a discrete crack and softening modeled using the Fictitious Crack Model (see Section 2.2.3.6.1), the `TangentStiff` option (see Section 3.3.1.1) should also be used to improve the rate of convergence for those increments. For analyses with snap-back or snap-through the relative displacement magnitude might require adjustment during the analyses to capture this behavior. If the `SpecifyCOD` option is omitted from the incremental load block, the incremental loads defined in the input file are applied unscaled.

3.3.3.3 LoadScale

The `LoadScale` option enables the user to control the step size for the modified-Newton algorithm in FCM analyses based on the value of the normal surface traction on the constrained interface at the crack tip (**?**). Since nonlinear analyses cannot be performed for heat transfer or seepage flow analyses, the `LoadScale` option has not been implemented for use with either

heat transfer or seepage flow analyses. If the `LoadScale` option is requested for a heat transfer or seepage flow analysis, the request will be ignored and a warning message will be printed to the formatted output file. A request for the `LoadScale` option will be ignored and a warning message will be printed to the formatted output file if a stress analysis is being performed and there are no interface elements in the mesh.

This method of step size control is generally used with proportional loadings, but it can also be used with arbitrary loadings by adjusting the relative displacement magnitudes such that the changes in the loading occur at the proper instances. A load factor, which indicates the magnitude of a proportional loading, is introduced to scale the applied incremental loads for each iteration such that the normal stress at the crack tip on the interface is equal to uniaxial tensile strength specified in the FCM material properties (see Section 2.2.3.6.1). Incremental load factors are accumulated to obtain a total load factor, which is meaningful only for proportional loadings, and both the incremental and total load factors are printed to the formatted output file at the end of each increment. The `LoadScale` option can be used in the same analysis with either the `ArcLength` or `SpecifyCOD` options to simulate a change in the loading, but not in the same increment as it would be virtually impossible to satisfy all the step size constrains simultaneously.

When using this algorithm for step size control another interface element should enter the fracture process zone at the beginning of every increment after increment 0. Therefore, the total number of increments is limited by the number of interface elements on the assumed crack path; further loading of the structure should be terminated when the fracture ligament is defined by no less than two interface elements. The algorithm will fail if all interface elements are allowed to become part of the fracture process zone. If the `LoadScale` option is omitted from the incremental load block, the incremental loads defined in the input file are applied unscaled.

3.4 Loading

3.4.1 Stress Analysis

The various type of loads supported by MERLIN are shown in Table 3.5.

Incremental and Total loads are also referred to as Direct and Indirect loads.

Indirect loads are those which are defined as a total load for a given increment only. This feature is essential for proper handling of the uplift/hydrostatic and some types of body forces loads in a nonlinear analysis.

3.4.1.1 Static vs Dynamic Loads

It should be noted that most (but not all) static loads are incremental. However, in the dynamic analysis it is the total load which is defined for each time step. Hence, if there is a constant load on a dynamically loaded structure, the constant load must be defined for each and every time increment.

Load Option	Load Scale Category	Static	Dynamic
`DispBCs`	Displacement Boundary Conditions	Inc.	Tot.
`PointLoads`	Point Loads	Inc.	Tot.
`Tractions`	Tractions	Inc.	Tot.
`UTRACT`	User Defined tractions	Inc.	Tot.
`BodyForces`	Body force applied to entire mesh	Inc.	Tot.
`Temperature`	Temperature	Inc.	Tot.
`Hydrostatic`	Hydrostatic load	Tot.	Tot.
`Westergaard`	pseudo hydrodynamic load	Tot.	Tot.
`Zangar`	pseudo hydrodynamic load	Tot.	Tot.
`Uplift`	Uplift forces	Tot.	Tot.
`BodyFrcList`	Body force of selected materials	Tot.	Tot.

Table 3.5: Incremental and Total Loads

3.4.1.2 NoLoadScale

Because of MERLIN's capability to handle differently hydrostatic loads from other codes, each load is categorized as either *Direct* or *Indirect*.

Thus the NoLoadScale option could have either one of two arguments, Direct or Indirect. The first will imply that none of the direct loads are scalable in a nonlinear analysis, whereas in the second none of the indirect loads are scaled. The default is that all loads (direct and indirect) are scalable.

By default, all loads are scaled. This option is useful if in the same load increment some loads are scalable and others are not.

3.4.1.3 DispBCs

The `DispBCs` option enables the user to define the incremental displacement boundary conditions. Displacement boundary conditions are the essential boundary conditions for stress analyses. For heat transfer and seepage flow analyses, displacement boundary conditions are not valid essential or natural boundary conditions and their presence is considered an error. On encountering this error, a message is printed to the formatted output file and execution of the program is terminated prematurely.

The line in the input file following the keyword contains the number of displacement boundary conditions. The number of displacement boundary conditions must be greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. The number of displacement boundary conditions is the number of displacement boundary conditions prescribed for nodes that appear in the input file. Displacement boundary conditions are automatically prescribed for mid-side nodes by the program if all corner nodes of an element surface have a displacement boundary condition prescribed for the same degree of freedom. Displacements at the mid-side nodes are computed using the shape functions of the lower order element defined by the corner nodes (i.e., the displacements on an element surface are not required to be constant). Following the line containing the number of displacement boundary conditions there must be one line for each displacement boundary condition. Each line includes three pieces of information defining the displacement boundary

condition:

1. Node number,
2. DOF number, and
3. Magnitude of the displacement.

The DOF number is 1 for a displacement in the x_1-direction, 2 for a displacement in the x_2-direction, and 3 for a displacement in the x_3-direction. The x_1-, x_2-, and x_3-directions are defined in Section 2.3.1. In two-dimensional analyses a DOF number of 3 is not allowed. If this situation is encountered the program will print an error message to the formatted output file and terminate execution prematurely. Failure to include a `DispBCs` option indicates that no displacement boundary conditions are to be prescribed for the current increment. Eventually, the lack of displacement boundary conditions will cause the program to terminate execution prematurely, since the solver is not capable of handling singular systems of equations (**?**). However, it is capable of recognizing a singular system and an error message indicating where the singularity occurred is printed to the formatted output file before execution terminates.

Node No.	DOF No.	Displacement

Table 3.6: Displacement Boundary Conditions Input

3.4.1.4 PointLoads

The `PointLoads` option enables the user to define the incremental point loads. Point loads are a natural boundary condition for stress analyses. For heat transfer and seepage flow analyses, point loads are not valid essential or natural boundary conditions and their presence is considered an error. On encountering this error, a message is printed to the formatted output file and execution of the program is terminated prematurely.

The line in the input file following the keyword contains the number of point loads. The number of point loads must be greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. Following the line containing the number of point loads there must be one line for each point load. Each line includes three pieces of information defining the point load:

1. Node number,
2. DOF number,
3. Magnitude of the point load.

The DOF number is 1 for a point load in the x_1-direction, 2 for a point load in the x_2-direction, and 3 for a point load in the x_3-direction. The x_1-, x_2-, and x_3-directions are defined in Section 2.3.1. In two-dimensional analyses a DOF number of 3 is not allowed. If this situation is encountered the program will print an error message to the formatted output file and terminate execution prematurely. Point loads should be avoided when using the mixed-iterative method, as they tend to cause convergence problems unless the mesh around the point load is very fine;

surface tractions (see Section 3.4.1.5) should be used to model point loads when the mixed-iterative method is used. Failure to include a `PointLoads` option indicates that no point loads are to be applied for the current increment.

Loads are defined as F/L, for axisymmetric analysis, the larger the radius (x), the smaller the stress induced by the point load.

Node No.	DOF No.	Point Load

Table 3.7: Point Load Input

3.4.1.5 Tractions

The `Tractions` option enables the user to define the incremental surface tractions for both 2- and 3-D elements. Surface tractions are a natural boundary condition for stress analyses. For heat transfer and seepage flow analyses, surface tractions are not valid essential or natural boundary conditions and their presence is considered an error. On encountering this error, a message is printed to the formatted output file and execution of the program is terminated prematurely.

The line in the input file following the keyword contains the number of surface tractions. The number of surface tractions must be greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. Following the line containing the number of surface tractions there must be one line for each surface traction. Each line includes up to eight pieces of information which define the surface traction:

1. Element number,
2. Surface number (see Section 2.2.2.2),
3. Magnitude of the normal surface traction, t_1,
4. Magnitude of the tangential surface traction in the local x_2-direction, t_2,
5. Magnitude of the tangential surface traction in the local x_3-direction, t_3,
6. First component v_x of the vector $\mathbf{v}$, defining the local coordinate system on the surface,
7. Second component v_y of the vector $\mathbf{v}$, defining the local coordinate system on the surface, and
8. Third component v_z of the vector $\mathbf{v}$, defining the local coordinate system on the surface.

The traction is always defined as F/L^2.

The direction for the normal surface traction corresponds to the local x_1-direction. A positive sign on the magnitude of the normal surface traction indicates that the traction is directed away from the interior of the element (i.e., produces a tensile stress in the element) and a negative sign indicates that the traction is directed into the interior of the element (i.e., produces a compressive stress in the element). The sign on the magnitude of the tangential surface traction in either the x_2- or x_3-directions indicates whether the traction is applied in the positive or

negative direction. The magnitude of the tangential surface traction in the local x_3-direction is required only for three-dimensional analyses; it is omitted for two-dimensional analyses. For three-dimensional analyses a vector **v**, defining the local coordinate system for the element surface, is also required. The vector **v** lies in the local x_1-x_2 plane. The conventions for the local coordinates systems for two- and three-dimensional elements are defined in Figures 3.1 and 3.2, respectively. Failure to include a `Tractions` option indicates that no surface tractions are to be applied for the current increment.

Element No.	Surface No.	t_1	t_2	$[t_3]$	$[v_x]$	$[v_y]$	$[v_z]$

Table 3.8: Surface Tractions Input

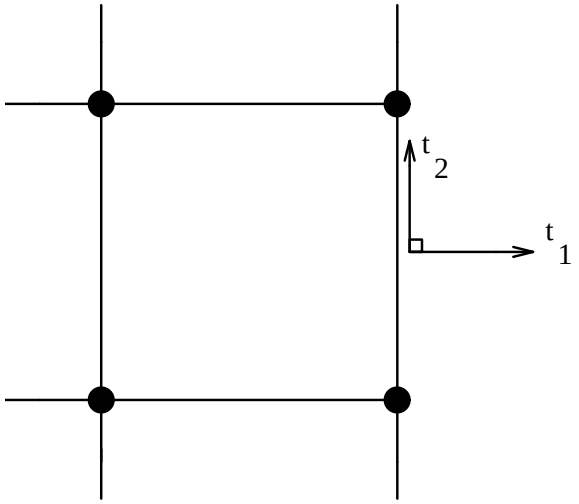

Figure 3.1: Positive local directions of surface tractions in 2D

Note that for 2D problems, the traction is defined for unit width. Hence, MERLIN will internally multiply the user defined traction by the thickness of the element. For 3D problems, the user defined traction is multiplied by the area of the corresponding surface.

To apply non-constant surface tractions for an element surface, the user must modify user subroutine `utract` to define the variation of the surface tractions. The default version of `utract` provides the user with the capability to apply hydrostatic loads by specifying the elevation of the fluid surface and the unit weight of the fluid in place of the magnitudes of the normal and tangential surface tractions. When using this option the mesh must be oriented such that the surface of the fluid is parallel to the global x-axis.

Note that if this option is used along with `utract`, then the arguments will be used by `utract` for internal operations and thus need not conform with the default definition.

3.4.1.5.1 UTRACT The `UTRACT` option enables the user to specify non-constant surface tractions on element surfaces through user subroutine `utract`. User subroutine `utract` is discussed in Section B.2 and an example problem illustrating the use of `utract` can be found in the MERLIN EXAMPLE PROBLEMS MANUAL. This option is used in conjunction with the

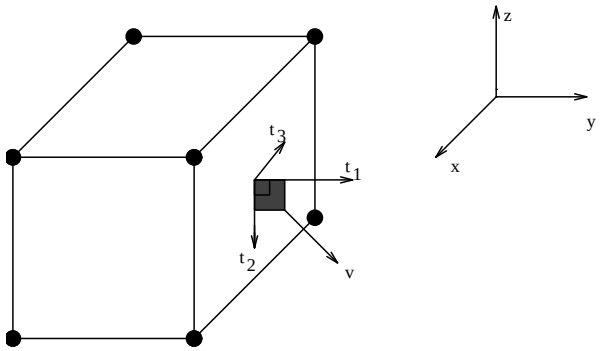

Figure 3.2: Positive local directions of surface tractions in 3D

`Tractions` option, otherwise the `UTRACT` option is ignored by the program. Each element surface to which a non-constant surface traction is to be applied must be defined in the `Tractions` option using fictitious values for the surface traction component magnitudes. The values of the surface traction magnitudes used to compute the equivalent nodal forces are computed in user subroutine `utract`. Failure to include a `UTRACT` option indicates that the applied surface tractions are constant over each element face.

3.4.1.6 AAR: DeltaTime

The `DeltaTime` option specifies the time corresponding to the current increment in which AAR expansion is assumed. The time must be in the same unit as the one adopted in the definition of $\tau_C(\theta_0^{test})$ and $\tau_L(\theta_0^{test})$ in Section 2.2.4.

Note: If *DeltaTime* is set to zero, then there is no expansion during this time increment. This may be useful to apply a load following AAR expansion.

3.4.1.7 AAR: AARMaxMod

The `AARMaxMod` option allows the user to change the maximum volumetric AAR strain from a given increment onward. This would typically be due to a change in relative humidity (it is often assumed that the maximum AAR strain is based on a 100% humidity, and drops through a factor of $RH^{0.8}$). The line following the keyword specifies the number of AAR material groups to be modified, and this is followed by a line with the AAR group ID and the new max AAR from that increment onward.

The new value remains in effect until it is subsequently modified in a later increment.

3.4.1.8 Dam Load

3.4.1.8.1 Hydrostatic The `Hydrostatic` option enables the user to define the hydrostatic loads for both 2- and 3-D elements.

This is the full load (not incremental) applied during a given increment.

Hydrostatic loads are special forms of surface tractions and as such they are natural boundary conditions for stress analyses. For heat transfer and seepage flow analyses, hydrostatic loads are not valid essential or natural boundary conditions and their presence is considered an error. On encountering this error, a message is printed to the formatted output file and execution of the program is terminated prematurely.

The line in the input file following the keyword contains the number of hydrostatic loads. The number of hydrostatic loads must be greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. Following the line containing the number of hydrostatic loads there must be one line for each hydrostatic load. Each line includes five pieces of information which define the hydrostatic load:

1. Element number,
2. Surface number (see Section 2.2.2.2),
3. Axis along which the fluid level is defined, *axis*. Integer number between 1 and 3. The number 1 means x axis, number 2 means y axis and number 3 means z axis.
4. Fluid level in the coordinates of the axis that was defined in the previous item, EL. Note that if the element integration point is higher than the current fluid level, then a zero pressure is assumed,
5. Weight density of the fluid γ_w.

It is assumed that the fluid surface is perpendicular to the specified coordinate system axis. The actual hydrostatic pressure is determined automatically in the program from the depth below the fluid surface. Any point that lies above this surface is considered to have zero hydrostatic pressure. Failure to include a `Hydrostatic` option indicates that no hydrostatic loads are to be applied for the current increment.

Element No.	Surface No.	*axis*	$EL.$	γ_w

Table 3.9: Hydrostatic Load Input

3.4.1.8.2 Westergaard Hydrodynamic The `Westergaard` option enables the user to define the pseudo-hydrodynamic loads formulated by Westergaard for dams for both 2- and 3-D elements. Note this option should be used only in Static analysis. For dynamic analysis, one should use the `NodalMass` option described in 2.3.4.

This is the full load (not incremental) applied during a given increment.

Hydrodynamic loads are similar to hydrostatic loads, and as such are special forms of surface tractions. These are natural boundary conditions for stress analyses and are not valid essential or natural boundary conditions for heat transfer and seepage flow analyses. The presence of hydrodynamic loads for heat transfer or seepage flow analyses is considered an error, and on encountering this error a message is printed to the formatted output file and execution of the program is terminated prematurely.

The line in the input file following the keyword contains the number of Westergaard hydrodynamic loads. The number of hydrodynamic loads must be greater than zero, otherwise the

program will print an error message to the formatted output file and terminate execution prematurely. Following the line containing the number of hydrodynamic loads there must be one line for each hydrodynamic load. Each line contains eleven pieces of information which define the hydrodynamic load:

1. Element number,
2. Surface number (see Section 2.2.2.2),
3. Axis along which the fluid level is defined, $axis$. Integer number between 1 and 3. The number 1 means x axis, number 2 means y axis, and number 3 means z axis.
4. Direction of the applied hydrodynamic traction in relation to the element being loaded, dir. 1 means the load is applied normal to the element surface **towards** the element centroid, while -1 means the load is applied normal to the element surface **away from** the element centroid.
5. Fluid level in the coordinates of the axis that was previously defined, El. Note that if the element integration point is higher than the current fluid level, then a zero pressure is assumed. Units of length.
6. Total height of the dam begin subjected to the hydrodynamic load, h. Note that dam over-topping by the hydrodynamic load is not permissible, so if the fluid level is set higher than the total dam height, the fluid level will be held at the dam height. Units of length.
7. K factor for Westergaard's equation. Defined by Westergaard as 51.0 lb/ft^3 or 8011.4 N/m^3.
8. Weight density of the fluid, γ_w. Units of force/length^3.
9. Acceleration of gravity, g. Units of length/time^2.
10. Modulus of elasticity of volume of the fluid, E_w. This is the fluid pressure divided by the reduction of volume per unit of volume, a measure of the compressibility of the fluid. Units of force/length^2.
11. Period to characterize the seismic acceleration imposed to the dam, t_e. Units of time.
12. Horizontal seismic acceleration coefficient applied at the base of the dam α, expressed in terms of the peak ground acceleration or spectral acceleration (fraction of g). Unit-less.

Failure to include a `Westergaard` option indicates that no hydrodynamic loads are to be applied for the current increment.

Element No.	Surface No.	$axis$	dir	El	h	K	γ_w	g	E_w	t_e	α

Table 3.10: Westergaard Hydrodynamic Load Input

3.4.1.8.3 Zangar Hydrodynamic The `Zangar` option enables the user to define the pseudo-hydrodynamic loads formulated by Zangar for dams for both 2- and 3-D elements. Note this option should be used only in Static analysis. For dynamic analysis, one should use the `NodalMass` option described in 2.3.4.

This is the full load (not incremental) applied during a given increment.

Hydrodynamic loads are similar to hydrostatic loads, and as such are special forms of surface tractions. These are natural boundary conditions for stress analyses and are not valid essential or natural boundary conditions for heat transfer and seepage flow analyses. The presence of hydrodynamic loads for heat transfer or seepage flow analyses is considered an error, and on encountering this error a message is printed to the formatted output file and execution of the program is terminated prematurely.

The line in the input file following the keyword contains the number of Zangar hydrodynamic loads. The number of hydrodynamic loads must be greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. Following the line containing the number of hydrodynamic loads there must be one line for each hydrodynamic load. Each line contains eleven pieces of information which define the hydrodynamic load:

1. Element number
2. Surface number (see Section 2.2.2.2)
3. Axis along which the fluid level is defined, $axis$. Integer number between 1 and 3. The number 1 means x axis, number 2 means y axis, and number 3 means z axis.
4. Fluid level in the coordinates of the axis that was previously defined, El. Note that if the element integration point is higher than the current fluid level, then a zero pressure is assumed. Units of length.
5. Total height of the dam begin subjected to the hydrodynamic load, h. Note that dam over-topping by the hydrodynamic load is not permissible, so if the fluid level is set higher than the total dam height, the fluid level will be held at the dam height, L.
6. Weight density of the fluid, γ_w, (F/L^3).
7. Acceleration coefficient a_h/g.
8. Surface angle with respect to the vertical in radians.

Element No.	Surface No.	$axis$	El	h	γ_w	a_h/g	α

Table 3.11: Zangar Hydrodynamic Load Input

3.4.1.8.4 Uplift/Uplift2 The `Uplift2` option enables the user to define (hydrostatic) uplift pressures on the discrete crack surfaces in 3D.

Note that for the full uplift to occur, the crack must have locally failed, and the crack opening displacement must be positive.

For the uplift to be fully applied with shear failure, user must specify `FulUplNegCod` (see sect. 3.4.1.8.5).

2D problems should use the keyword `Uplift`, careful make sure that the interface elements are numbered counterclockwise.

This is the full load (not incremental) applied during a given increment.

Uplift pressures are a natural boundary condition for stress analyses. For heat transfer and seepage flow analyses, uplift pressures are not valid essential or natural boundary conditions and their presence is considered an error. On encountering this error, a message is printed to the formatted output file and execution of the program is terminated prematurely.

The line in the input file following the keyword contains the number of discrete cracks subjected to uplift pressures. The number of discrete cracks subjected to uplift pressures must be greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. Following the line containing the number of discrete cracks subjected to uplift pressures there must be one or two lines for each discrete crack. The first line includes seven pieces of information which define `uplift2`:

1. Discrete crack number,
2. Uplift model type (1-4), Fig. 3.4
3. Water elevation in global coordinate system
4. Id of the global coordinate axis along which the uplift is applied (x =1, y=2, z=3).
5. Unit weight of water, γ_W.
6. Directions of uplift forces. 0: both sides, -1: lower side, 1: upper side.
7. Direction of crack (from upstream to downstream)
   (a) 2D: User must simply define direction of ± 1 if along $\pm x$, and ± 2 if along $\pm y$.
   (b) 3D: User must define a vector v_x, v_y, v_z from upstream to downstream direction.

The discrete crack number must be greater than zero and less than or equal to the number of discrete cracks specified in the `CrackSurface` option (see Section 2.4.1), otherwise the program will print an error message to the formatted output file and terminate execution prematurely.

For 3D cracks, the uplift pressure profile is calculated based on the selected uplift model and it is interpolated along the specified direction of crack propagation, Fig. **??**. Then it is applied along the true crack path, which may differ from a straight line. The following assumptions are made during the uplift calculation:

1. It is assumed that the crack propagates along the direction of crack propagation.
2. Uplift value is calculated and written into the output file as an average value over the whole element surface.
3. The distance from the crack mouth is calculated as the largest projection from the current point to the most far point along the specified crack propagation direction.

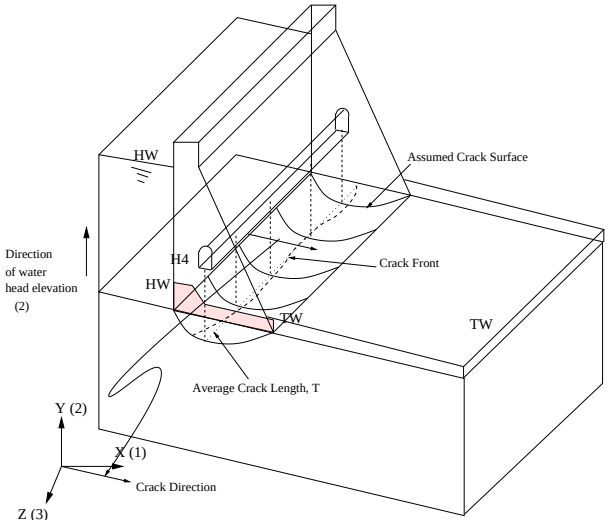

Figure 3.3: Uplift Description in 3D

Crack No.	Uplift Model Type	H_W	axis	γ_W	**Direction**	Direct. of Prop.

Table 3.12: Uplift Pressures Input

4. Crack tip distance is assumed constant over the crack surface and it is calculated as an average from all crack tip positions.

The second line would contain specific information for each of the selected uplift models.

Model 1: Full Uplift There is no need for additional line

Model 2: FERC Model should have the following six data, (**?**), Fig. 3.5.

1. Distance from crack mouth to the drain
2. Drain efficiency e ($0 < e < 1$) where 1. corresponds to full efficiency) before it is intersected by the crack. If there is no drain, specify 0.
3. Drain efficiency after it has been intersected by the crack
4. Vertical distance between the crack subjected to uplift and the drain
5. Tail water elevation (or water elevation at the end of the crack) measured with respect to the crack subjected to uplift
6. Total base length of the dam from upstream to downstream.
   (a) 2D:User must specify the length.
   (b) 3D: If left as 0., then Merlin internally computes this length, if non-zero, then Merlin will consider this value as the length.

   Note that this is not the projected length, but the actual "curvilinear" length of the crack.

Model 3: Brühwiler/Reich Model in which the uplift pressure is a function of the crack opening displacements. No additional data need to be specified for this model. However, if an `LEFM` option (see Section 1.4.1) appears in the program control block, the `p_W0-COD_W0` option (see Section 1.4.2) must also appear in the program control block. The program does not support a default $p_{W0} - COD_{W0}$ relationship (in order to avoid problems with units), so a $p_{W0} - COD_{W0}$ relationship must be defined in the input file for the program to be able to compute uplift pressures. If the program control block includes the `LEFM` option but not the `p_W0-COD_W0` option the program will print an error message to the formatted output file and terminate execution prematurely. Failure to include an `Uplift` option indicates that no uplift pressures are to be applied for the current increment.

Model 4: User Defined There should be four data

1. Distance from crack mouth to the drain
2. Drain efficiency (0.-1.0 where 1.0 corresponds to full efficiency) before it is intersected by the crack (drains are typically 50 to 60% efficient when new), (**?**).
3. Drain efficiency after it has been intersected by the crack
4. Drain elevation
5. Number of data points N specifying the uplift pressure

Following this, there should be N lines, each one containing

1. Distance from crack mouth.
2. Corresponding water elevation (uplift pressure)

Note:

1. When interface elements are used, the uplift pressure is applied when the corresponding Gauss point has failed (i.e. $F > 0$), **and** there is a positive crack opening displacement.
2. In a nonlinear incremental analysis, the uplift water elevations are automatically adjusted.
3. Uplift models 2 and 4 are not yet implemented in 3D analysis.
4. `uplift` relies heavily on the fact that the crack surface definition (not part of the uplift) must be from crack tip to crack mouth and a crack direction is used. Crack direction can be only x-y-z (too restrictive) and relied heavily on the crack surface definition. When we tried to extend to 3d, this information was not sufficient, and not easy, which is why we first introduced the crack direction, but this was not enough, so we introduced `uplift2`.

3.4.1.8.5 FulUplNegCod By default, Merlin will apply the full uplift if there is a positive crack opening displacement and the joint has failed.

`FulUplNegCod` will apply the full uplift if the joint has failed eventhough we may still have a negative crack opening displacement (i.e. shear failure).

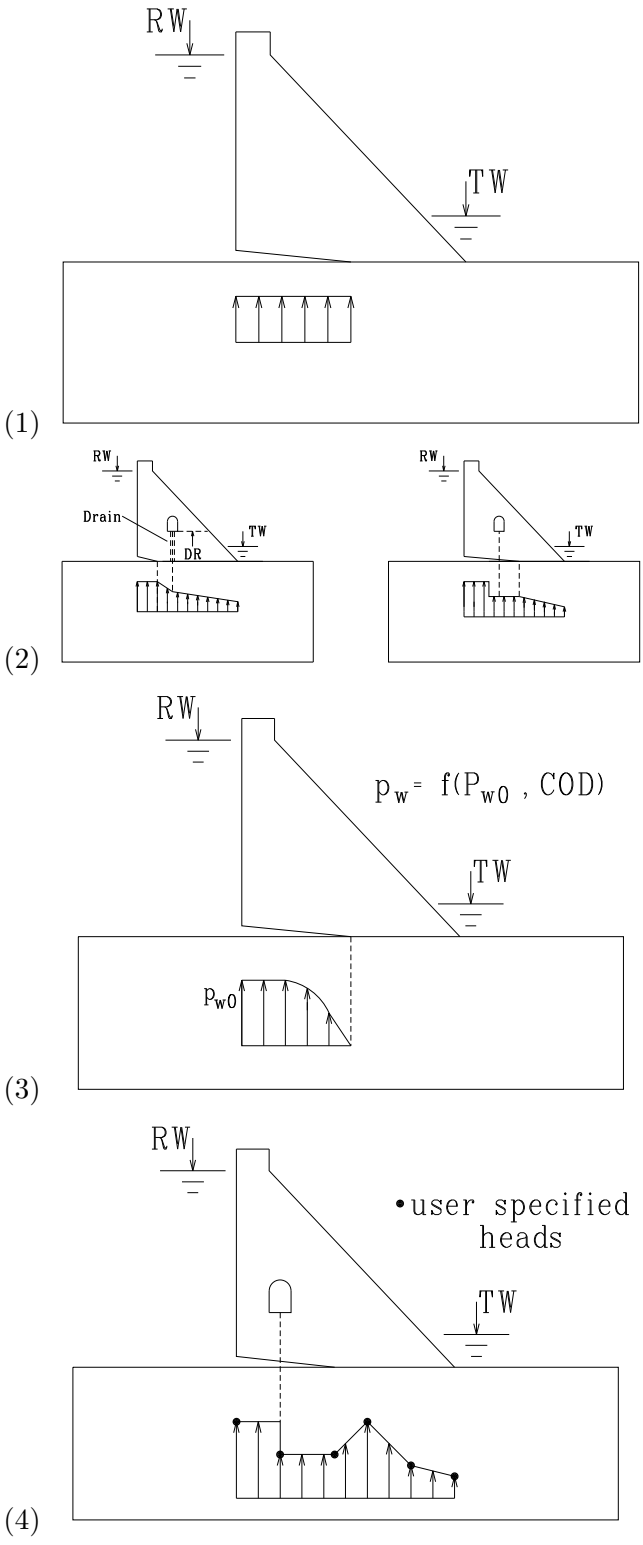

Figure 3.4: Uplift models in MERLIN ((1) Full, (2) FERC, (3) Brühwiller-Reich, (4) User defined).

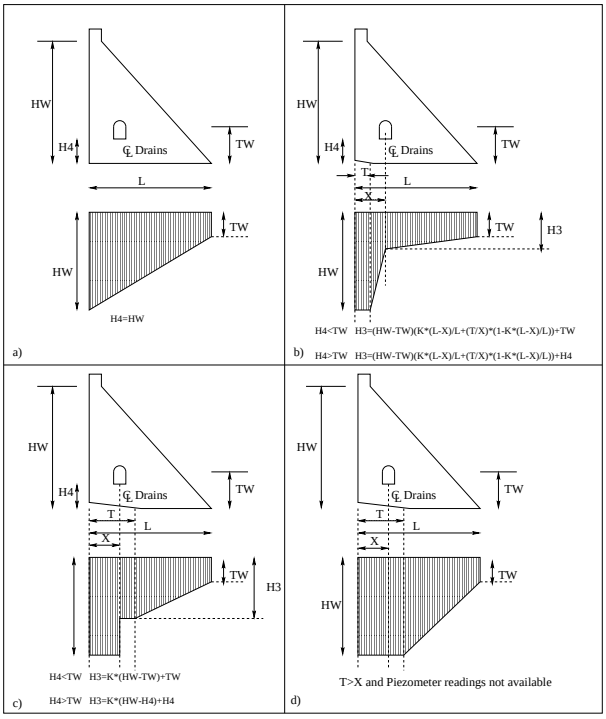

Figure 3.5: FERC Uplift Model as Implemented in Merlin

3.4.1.8.6 DynamUplift `DynamUplift` is used to specify the uplift model to be adopted in a dynamic analysis. Failure to specify this option will result in Merlin maintaining the static uplift during dynamic analysis.

Following the keyword, is an integer which is the number n of cracks to be subjected to a dynamic uplift. Then, are n pairs of integers which are the crack number, and the dynamic uplift model specified as a curve defined in `UserCurves`.

The corresponding `UserCurves` defines the ratio of the dynamic uplift to the static one (y) in terms of the crack opening displacement rate, (x).

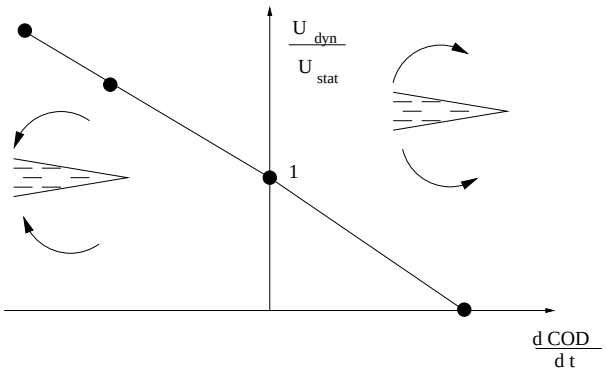

Figure 3.6: Dynamic Uplift Definition

3.4.1.9 Body Forces

3.4.1.9.1 BodyForces The `BodyForces` option enables the user to define the incremental body forces for stress analyses. For heat transfer and seepage flow analyses, body forces are not valid and their presence is considered an error. On encountering this error, a message is printed to the formatted output file and execution of the program is terminated prematurely.

The line in the input file following the keyword contains the magnitude of the acceleration, a, and the vector defining the direction of acceleration, v_i. The direction vector is not required to be a unit vector; the program unitizes the vector before computing the components of the acceleration vector. The number of components required to define the direction vector corresponds to the number of coordinates defining the geometry of the mesh (i.e., two components for a two-dimensional analysis and three components for a three-dimensional analysis). Failure to include a `BodyForces` option indicates that no body forces are to be applied for the current increment.

a	v_1	v_2	$[v_3]$

Table 3.13: Body Forces Input

To apply non-constant body forces, the user must modify user subroutine `ubody` to define the variation of the body forces. The default version of `ubody` simply returns body forces based on the values provided by the user with the `BodyForce` option.

3.4.1.9.1.1 UBODY The `UBODY` option enables the user to specify non-constant body forces through user subroutine `ubody`. User subroutine `ubody` is discussed in Section B.3 and an example problem illustrating the use of `ubody` can be found in the MERLIN EXAMPLE PROBLEMS MANUAL. This option is used in conjunction with the `BodyForces` option, otherwise the `UBODY` option is ignored by the program. The information defined in the `BodyForces` option is available in user subroutine `ubody` and can be used in the computation of the non-constant body forces. Failure to include a `UBODY` option indicates that the applied body forces are constant over the problem domain.

3.4.1.9.2 BodyFrcList The `BodyFrcList` enables the user to specify those material groups subjected to the previously defined `BodyForces` option.

`BodyForces` first applies by default body forces to all the groups, whereas `BodyFrcList` selectively applies the body forces to one or more groups. This should enable the user to partially simulate stage construction and/or excavation.

It should be noted that contrary to `BodyForces` which are direct loads (once defined, they are acive in all subsequent increments), `BodyFrcList` are indirect loads and are active only in the current load increment.

Furthermore, `BodyFrcList` must always be preceded by `BodyForces`.

The line in the input file following the keyword contains the number of material groups subjected to body forces.

Following the line containing the number of materials be one line for each material group id.

3.4.1.10 Eigen Analysis

3.4.1.10.1 EigenModes The `EigenModes` instructs MERLIN to compute the n lowest eigenvalues and corresponding eigenvectors for the current load/time increment. n should have been specified in the control block in conjunction with the `NumEigenVals` keyword.

For a static (linear or nonlinear) analysis, the stiffness matrix eigenvalues are determined on the basis of $|\mathbf{K}_{tan} - \lambda\mathbf{I}| = \mathbf{0}$. Those eigenvalues, in the context of a nonlinear analysis, may provide additional indicators of failure and failure modes for a given increment.

For a transient (linear or nonlinear) analysis, the structure eigenvalues are determined on the basis of $|\mathbf{K}_{tan} - \lambda\mathbf{M}| = \mathbf{0}$.

3.4.1.10.2 EigenFile The `EigenFile` option instructs MERLIN to output the results of an eigenvalue and eigenvector calculations to an ASCII external file whose name is specified as an argument for this keyword.

Note that for this file to be viewed by the accompanying external viewer, the filename should have an extension `.eig`.

3.4.1.11 Time Integration

3.4.1.11.1 Velocities The `Velocities` option enables the user to define the incremental velocity boundary conditions. Velocity boundary conditions are the essential boundary conditions for stress analysis. For heat transfer and seepage flow analysis, velocity boundary conditions are not valid essential or natural boundary conditions and their presence is considered an error. On encountering this error, a message is printed to the formatted output file and execution of the program is terminated prematurely.

The line in the input file following the keyword contains the number of velocity boundary conditions. The number of velocity boundary conditions must be greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. The number of velocity boundary conditions is the number of velocity boundary conditions prescribed for nodes that appear in the input file. Velocity boundary conditions are automatically prescribed for mid-side nodes by the program if all corner nodes of an element surface have a velocity boundary condition prescribed for the same degree of freedom. Velocities at the mid-side nodes are computed using the shape functions of the lower order element defined by the corner nodes (i.e., the velocities on an element surface are not required to be constant). Following the line containing the number of velocity boundary conditions there must be one line for each velocity boundary condition. Each line includes three pieces of information defining the velocity boundary condition:

1. Node number,
2. DOF number, and
3. Magnitude of the velocity.

The DOF number is 1 for a velocity in the x_1-direction, 2 for a velocity in the x_2-direction, and 3 for a velocity in the x_3-direction. The x_1-, x_2-, and x_3-directions are defined in Section 2.3.1. In two-dimensional analysis a DOF number of 3 is not allowed. If this situation is encountered

the program will print an error message to the formatted output file and terminate execution prematurely. Failure to include a `Velocities` option indicates that no velocity boundary conditions are to be prescribed for the current increment.

Node No.	DOF No.	Velocity

Table 3.14: Velocity Boundary Conditions Input

3.4.1.11.2 Acceleration The `Acceleration` option enables the user to define the incremental acceleration boundary conditions. Acceleration boundary conditions are the essential boundary conditions for stress analysis. For heat transfer and seepage flow analysis, acceleration boundary conditions are not valid essential or natural boundary conditions and their presence is considered an error. On encountering this error, a message is printed to the formatted output file and execution of the program is terminated prematurely.

The line in the input file following the keyword contains the number of acceleration boundary conditions. The number of acceleration boundary conditions must be greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. The number of acceleration boundary conditions is the number of acceleration boundary conditions prescribed for nodes that appear in the input file. Acceleration boundary conditions are automatically prescribed for mid-side nodes by the program if all corner nodes of an element surface have an acceleration boundary condition prescribed for the same degree of freedom. Accelerations at the mid-side nodes are computed using the shape functions of the lower order element defined by the corner nodes (i.e., the accelerations on an element surface are not required to be constant).

Merlin will automatically dump the acceleration boundary condition to the post file. Hence, the accelerogram could be viewed by Spider.

3.4.1.11.2.1 Flag=0 in `Transient` Option of Control Block This is the most general alternative to specify both Velocities and Accelerations at each time increment.

Within each time increment, we can specify boundary values for velocities or accelerations.

For Accelerations, following the `Acceleration` keyword, user must specify:

1. Total number of degrees of freedom with prescribed accelerations at the current time increment.
2. For each degree of freedom, enter:
   (a) Node number.
   (b) DOF number.
   (c) Magnitude of the acceleration.

   The DOF number is 1 for an acceleration in the x_1-direction, 2 for an acceleration in the x_2-direction, and 3 for an acceleration in the x_3-direction. The x_1-, x_2-, and x_3-directions are defined in Section 2.3.1. In two-dimensional analysis a DOF number of 3 is not allowed. If this situation is encountered the program will print an error message to the formatted

output file and terminate execution prematurely. Failure to include a `Acceleration` option indicates that no acceleration boundary conditions are to be prescribed for the current increment.

Node No.	DOF No.	Acceleration

Table 3.15: Acceleration Boundary Conditions Input

The same process is repeated for velocities (if any). Following the `Velocities` keyword, user must specify:

1. Total number of degrees of freedom with prescribed accelerations at the current time increment.
2. For each degree of freedom, enter:
   (a) Node number.
   (b) DOF number.
   (c) Magnitude of the velocity.

   The DOF number is 1 for an velocity in the x_1-direction, 2 for a velocity in the x_2-direction, and 3 for a velocity in the x_3-direction. The x_1-, x_2-, and x_3-directions are defined in Section 2.3.1. In two-dimensional analysis a DOF number of 3 is not allowed. If this situation is encountered the program will print an error message to the formatted output file and terminate execution prematurely. Failure to include a `Velocities` option indicates that no velocity boundary conditions are to be prescribed for the current increment.

Node No.	DOF No.	Velocity

Table 3.16: Velocity Boundary Conditions Input

3.4.1.11.2.2 Flag=1 in `Transient` Option of Control Block This is a more compact way of defining the dynamic load. First we specify the nodes which will be subjected to the accelerations, and then we specify the accelerations for each time t. Hence, all the nodes will be subjected to the same acceleration. Note in this case, only one "Increment" is written.

Following the **Acceleration** keyword, user must specify:

1. Total number of degrees of freedom subjected to imposed acceleration.
2. For each excited degree of freedom, enter: node number, dof No., followed by a zero (0.0)
3. Total number of time increments.
4. For each time increment, enter t, flag for pst file output (1=yes, 0=no), followed by the two or three nodal accelerations.

Node	dof	0.0

Table 3.17: Degrees of Freedom Subjected to Imposed Acceleration Boundary Conditions

t_i	Flag	a_x	a_y	$[a_z]$

Table 3.18: Acceleration Boundary Conditions Input

Then a line containing the number of acceleration boundary conditions and the following set of information must be supplied

where t_i is the time at increment i, The flag is set to 0 if results of this increment are not to be included in the post file, 1 otherwise. a_x, a_y and a_z are the corresponding accelerations to all the restrained DOF. For two dimensional analysis, a_z is not specified.

For seismic analysis it is advisable to use Flag=0 in the `Transient` option. Note that if $\Delta t^* = t_{i+1} - t_i$ is different from the Δt specified in the `Transient` option, then MERLIN will use Δt^*.

3.4.1.12 Centrifugal

The `Centrifugal` option enables the user to define the incremental centrifugal forces for stress analyses. For heat transfer and seepage flow analyses, centrifugal forces are not valid and their presence is considered an error. On encountering this error, a message is printed to the formatted output file and execution of the program is terminated prematurely.

This option is to be used when the centrifugal (or gravity) forces varies along a structure (such as in tests performed inside a centrifuge, or electromagnetic forces).

The line in the input file following the keyword contains the magnitude of the angular velocity in radians per second, ω, a point on the axis of rotation, x_i, and a vector specifying the direction of the axis of rotation, v_i. The direction vector is not required to be a unit vector; the program utilizes the vector before computing the components of the radius vector and, eventually, the components of the acceleration vector. The number of coordinates required to define the point on the rotation axis and the number of components required to define the direction vector corresponds to the number of coordinates defining the geometry of the mesh (i.e., two components for a two-dimensional analysis and three components for a three-dimensional analysis). Failure to include a `Centrifugal` option indicates that no centrifugal forces are to be applied for the current increment.

ω	x_1	x_2	$[x_3]$	v_1	v_2	$[v_3]$

Table 3.19: Centrifugal Forces Input

It should be recalled that $F = m\omega^2 r$, where $r = x_i$ and that $\omega = \sqrt{\frac{g}{r}}$.

3.4.1.13 Pressures

The `Pressures` option enables the user to specify nodal hydrostatic pressures inside the structure for stress analyses. For heat transfer and seepage flow analyses, internal hydrostatic pressures are not valid and their presence is considered an error. On encountering this error, a message is printed to the formatted output file and execution of the program is terminated prematurely.

The internal hydrostatic pressures could be either pore pressures or pressures due to seepage flow. The pressures are interpolated to the integration points for each element and assembled into stress vectors. The product of the transpose of the strain-displacement matrix and the stresses are integrated over the element domains to compute element nodal force vectors. The element nodal force vectors are added to the applied load vector for the structure (**?**, **?**). When internal hydrostatic pressures are specified for the structure, the stresses printed in the formatted and unformatted output files are effective stresses. The effective stresses are used when evaluating the contour integrals (see Sections 1.4.1.1 and 1.4.1.2) and the pressure gradients are treated as body forces for the elements within the contour paths (**?**).

The line in the input file following the keyword contains the number of nodes for which pressures are specified. The number of nodes for which pressures are specified must be greater than zero, otherwise the program will print an error message to the formatted output file and execution will be terminated prematurely. Following the line containing the number of nodes for which pressures are specified there must be one line for each node and its specified pressure. Each line includes two pieces of information defining the nodal pressure:

1. Node number and
2. Pressure.

At those nodes for which no pressure is defined, a pressure of zero is assumed. Failure to include a `Pressures` option indicates that hydrostatic pressures are not present in the structure for the current increment.

3.4.1.14 Temperatures

3.4.1.14.1 NodalTemperatures The `Temperatures` option enables the user to specify **incremental** nodal temperatures for stress analysis. In an incremental analysis at increment i the temperatures correspond to the ΔT_i with respect to the previously accumulated differential temperatures $\sum_{j=1}^{j=i-1} T_j$.

If no temperature is specified for a node it is assumed to be at the stress free temperature. The temperature differentials are used to compute initial strains and, through the constitutive law, thermal stresses. The product of the transpose of the strain-displacement matrix and the thermal stresses are integrated over the element domains to compute element nodal force vectors. The element nodal force vectors are added to the applied initial load vector for the structure (**?**).

The line in the input file following the keyword contains the number of nodes for which temperatures are specified. The number of nodes for which temperatures are specified must be greater than zero, otherwise the program will print an error message to the formatted output file and execution will be terminated prematurely. Following the line containing the number of

nodes for which temperatures are specified there must be one line for each node and its specified temperature. Each line includes two pieces of information defining the nodal temperature:

1. Node number and
2. Temperature.

Failure to include a `Temperatures` option for a stress analysis indicates that temperature effects are not to be considered for the current increment.

3.4.1.14.2 PointTemps The `PointTemps` option enables the user to import a spatial temperature distribution (most likely obtained from a separate analysis with a different mesh) and then Merlin will map it into nodes of user specified element groups.

1. Number of element groups which nodes are to be assigned a temperature from the subsequent set of data.
2. Element group(s).
3. Number of data points.
4. x_i, y_i, $[z_i]$, T_i

Note that no internal interpolation is performed. Merlin will simply loop on each node inside the selected group, identify the closest point with a user supplied temperature, and assigns this temperature to that node.

3.4.1.14.3 TempVarElev The `TempVarElev` option enables the user to define two nodal temperatures at a node, a coordinate axis, and a reference elevation.

If the selected coordinate is lower than the specified one, one temperature is assigned, otherwise the other is assigned.

This feature is useful when a dam upstream face is subjected to a variable pool elevation, and air and water have different temperatures.

Following the `TempVarElev` keyword, the number of nodes to be assigned a temperature is specified, followed by where node_id is the node number, axis_id is the axis (1, 2 or 3) corresponding

node_id	axis_id	T_1	T_2	Elev.

Table 3.20: Variable Nodal Temperature Definition

to the elevation definition, T_1 and T_2 are the temperatures, and is the elevation. If x_{axis_id} is below Elev. than the node is assigned temperture T_1, otherwise it is assigned temperature T_2.

3.4.2 Field Problems

3.4.2.1 Temperatures

The `Temperatures` option enables the user to specify nodal temperatures for heat transfer analyses.

In transient analysis, the temperature is the total absolute temperature.

In steady state analysis, the temperature is the incremental one (**CHECK**).

The line in the input file following the keyword contains the number of nodes for which temperatures are specified. The number of nodes for which temperatures are specified must be greater than zero, otherwise the program will print an error message to the formatted output file and execution will be terminated prematurely. Following the line containing the number of nodes for which temperatures are specified there must be one line for each node and its specified temperature. Each line includes two pieces of information defining the nodal temperature:

1. Node number and
2. Temperature.

If the `Temperatures` option is omitted for a heat transfer analysis, the program will eventually terminate prematurely as the solver is not capable of handling singular systems of equations (**?**). However, the solver is capable of recognizing that a singularity exists and an error message indicating where the singularity occurred is printed to the formatted output file before execution terminates.

3.4.2.2 PointFluxes

The `PointFluxes` option enables the user to define the incremental point fluxes. Point fluxes are a natural boundary condition for heat transfer and seepage flow analyses. For stress analyses, point fluxes are not valid essential or natural boundary conditions and their presence is considered an error. On encountering this error, a message is printed to the formatted output file and execution of the program is terminated prematurely.

The line in the input file following the keyword contains the number of point fluxes. The number of point fluxes must be greater than zero, otherwise the program will print an error message to the formatted output file and terminate execution prematurely. Following the line containing the number of point fluxs there must be one line for each point flux. Each line includes two pieces of information defining the point flux:

1. Node number and
2. Magnitude of the point flux.

Point fluxes should be avoided when using the mixed-iterative method (**after it has been implemented**), as they tend to cause convergence problems unless the mesh around the point flux is very fine; surface fluxes (see Section 3.4.2.3) should be used to model point fluxes when the mixed-iterative method is used. Failure to include a `PointFluxes` option indicates that no point fluxes are to be applied for the current increment.

Node No.	Point Flux

Table 3.21: Point Flux Input

3.4.2.3 Fluxes

The `Fluxes` option enables the user to specify incremental surface fluxes. Surface fluxes are a natural boundary condition for heat transfer and seepage flow analyses. For stress analyses, surface fluxes are not valid essential or natural boundary conditions and their presence is considered an error. On encountering this error, a message is printed to the formatted output file and execution of the program is terminated prematurely.

The line in the input file following the keyword contains the number of element surfaces for which fluxes are specified. The number of element surfaces for which fluxes are specified must be greater than zero, otherwise the program will print an error message to the formatted output file and execution will be terminated prematurely. Following the line containing the number of element surfaces for which fluxes are specified there must be one line for each element surface and its specified flux. Each line includes three pieces of information defining the surface flux:

1. Element number,
2. Element surface number, and
3. Flux.

The flux is applied normal to the element surface. A positive sign for the magnitude of the flux indicates that the heat flow is directed away from the interior of the element and a negative sign indicates that the flow is directed into the interior of the element. Failure to include a `Fluxes` option indicates that no surface fluxes are to be applied for the current increment.

To apply non-constant surface fluxes for an element surface, the user must modify the user subroutine `uflux` to define the variation of the surface fluxes. The default version of `uflux` simply returns the surface fluxes defined by the user in the input file.

3.4.2.3.1 FluxVarElev The `FluxVarElev` option enables the user to define two fluxes for an element surface, a coordinate axis, and a reference elevation.

If the selected coordinate is lower than the specified one, one flux value is assigned, otherwise the other is assigned.

This feature is useful when a dam upstream face is subjected to a variable pool elevation, and air and water have different fluxes.

Following the `FluxVarElev` keyword, the number of elements to be assigned a flux is specified, followed by where element_id is the node number, surface # corresponds to the element surface

element_id	surface #	axis_id	q_1	q_2	Elev.

Table 3.22: Variable Element FLux Definition

number defined in Sect. 2.2.2.2, axis_id is the axis (1, 2 or 3) corresponding to the elevation definition, q_1 and q_2 are the fluxes, and Elev. is the elevation. If x_{axis_id} is below Elev. than the node is assigned flux q_1, otherwise it is assigned flux q_2.

3.4.2.3.2 UFLUX The `UFLUX` option enables the user to specify non-constant fluxes on element surfaces for heat transfer and seepage flow analyses. User subroutine `uflux` is discussed

in Section B.4 and an example problem illustrating the use of `uflux` can be found in the MERLIN EXAMPLE PROBLEMS MANUAL. This option is used in conjunction with the `Fluxes` option, otherwise the `UFLUX` option is ignored by the program. Each element surface to which a non-constant flux is to be applied must be specified in the `Fluxes` option using arbitrary values for the flux magnitudes. The values of the flux magnitudes used to compute the equivalent nodal "loads" are determined in user subroutine `uflux`. If the `UFLUX` option is omitted from the incremental load block but the `Fluxes` option is not, the resulting fluxes will be constant over the element surfaces.

3.4.2.4 Films

The `Films` option enables the user to specify incremental convection heat fluxes (i.e., film coefficients and fluid temperatures) on element surfaces. Convection heat fluxes are a natural boundary condition for heat transfer. For stress analyses and seepage flow analyses, convection heat fluxes are not valid essential or natural boundary conditions and their presence is considered an error. On encountering this error, a message is printed to the formatted output file and execution of the program is terminated prematurely.

The line in the input file following the keyword contains the number of element surfaces for which convection heat fluxes are specified. The number of element surfaces for which convection fluxes are specified must be greater than zero, otherwise the program will print an error message to the formatted output file and execution will be terminated prematurely. Following the line containing the number of element surfaces for which convection heat fluxes are specified there must be one line for each element surface and its specified film coefficient and fluid temperature. Each line includes four pieces of information defining the convection heat flux:

1. Element number,
2. Element surface number,
3. Film coefficient, and
4. Fluid temperature.

The convection heat flux is applied normal to the element surface.

Note In steady state analysis, user must specify incremental temperature. For transient analysis it should correspond to the total temperature.

To apply non-constant fluid temperatures and film coefficients for an element surface, the user must modify the user subroutine `ufilm` to define the variation of the film coefficients and fluid temperatures. The default version of `ufilm` simply returns the film coefficients and fluid temperatures defined by the user in the input file.

3.4.2.4.1 UFILM The `UFILM` option enables the user to specify non-constant convection heat fluxes on element surfaces for heat transfer analyses. User subroutine `ufilm` is discussed in Section B.4 and an example problem illustrating the use of `ufilm` can be found in the MERLIN EXAMPLE PROBLEMS MANUAL. This option is used in conjunction with the `Films` option, otherwise the `UFILM` option is ignored by the program. Each element surface to which a non-constant convection heat flux is to be applied must be specified in the `Films` option using

arbitrary values for the film coefficient and fluid temperature magnitudes. The values of the film coefficient and fluid temperature magnitudes used to compute the equivalent nodal "loads" are determined in user subroutine `ufilm`. If the `UFILM` option is omitted from the incremental load block but the `Films` option is not, the resulting convective heat fluxes will be constant over the element surfaces.

3.4.2.5 BodyHeat

The `BodyHeat` option enables the user to define the incremental internally generated heat (i.e., body heat) on an element-by-element basis. For stress analyses and seepage flow analyses, internally generated heat is not valid and its presence is considered an error. On encountering this error, a message is printed to the formatted output file and execution of the program is terminated prematurely.

The line in the input file following the keyword contains the number of elements for which internally generated heat is specified. The number of elements for which internally generated heat is specified must be greater than zero, otherwise the program will print an error message to the formatted output file and execution will be terminated prematurely. Following the line containing the number of elements for which internally generated heat is specified there must be one line for each element and its prescribed internally generated heat. Each line includes two pieces of information defining the internally generated heat:

1. Element number and
2. Internally generated heat.

A positive sign for the magnitude of the internally generated heat indicates that the heat flow is out of the element and negative sign indicates that the flow is into the element. Failure to include a `BodyHeat` option indicates that no internally generated heat is to be applied for the current increment.

Element No.	Body Heat

Table 3.23: Body Heat Input

To apply non-constant internally generated heat, the user must modify user subroutine `uheat` to define the variation of the internally generated heat. The default version of `uheat` simply returns the internally generated heat defined by the user in the input file.

3.4.2.5.1 UHEAT The `UHEAT` option enables the user to specify non-constant internally generated heat through user subroutine `uheat`. User subroutine `uheat` is discussed in Section B.6 and an example problem illustrating the use of `uheat` can be found in the MERLIN EXAMPLE PROBLEMS MANUAL. This option is used in conjunction with the `BodyHeat` option, otherwise the `UHEAT` option is ignored by the program. The information defined in the `BodyHeat` option is available in user subroutine `uheat` and can be used in the computation of the non-constant internally generated heat. Failure to include a `UHEAT` option indicates that the internally generated heat is constant over each element.

3.4.2.6 Heads

The `Heads` option enables the user to specify nodal heads that are used to define essential boundary conditions in seepage flow analyses. For stress and heat transfer analyses, the heads are not valid essential or natural boundary conditions and their presence is considered an error. On encountering this error, a message is printed to the formatted file and execution of the program is terminated prematurely.

The line in the input file following the keyword contains the number of nodes for which heads are specified. The number of nodes for which heads are specified must be greater than zero, otherwise the program will print an error message to the formatted output file and execution will be terminated prematurely. Following the line containing the number of nodes for which heads are specified there must be one line for each node and its specified head. Each line includes two pieces of information defining the nodal head:

1. Node number and
2. Head.

At those nodes for which no head is defined, the head is a primary unknown. If the `Heads` option is omitted from the incremental load block, execution of the program will eventually terminate prematurely as the solver is not capable of handling singular systems of equations (**?**). However, the solver is capable of recognizing that a singularity exists and an error message indicating where the singularity occurred is printed to the formatted output file before execution terminates.

Chapter 4

OUTPUT FILES

MERLIN produces two output files: one formatted and one unformatted. The formatted output file is an ASCII file that contains an echo print of the input data, informational messages, and incremental results. The unformatted output file is a binary file that contains incremental results and information required for program restart and post-processing. The contents of each output file are discussed in detail in this chapter. Users should make a point of reading this chapter so that they are fully aware of the contents of the output files and the tasks for which they were designed.

4.1 Formatted Output File

The formatted output file contains an echo print of the information read from the input file; informational messages generated by the program; and incremental analysis results. Each piece of information appearing in the input file is written to the formatted output file so that the user can locate and correct input errors more quickly. Informational messages report the status of the iterative solution and indicate that an actual or potential error has been encountered. Incremental analysis results are included in the formatted output file only when the user specifically requests their presence (see Sections 1.7.4 through 1.7.8).

Information from the input file that is echoed in the formatted output file is written in the format prescribed in Chapters 1 through 3 of this manual. Keywords are echoed with the appropriate combination of upper and lower case letters. Integer values are printed using a field width of five and floating point values are printed in exponential form using a field width of ten. The logical line structure defined for the input options discussed in Chapters 1 through 3 is strictly adhered to.

An informational message reporting the status of the iterative solution is printed for each iteration. This message includes the computed value for each error measure and the corresponding tolerance used to determine convergence. When convergence is reached, a message is written indicating that this is the case. Informational messages reporting the status of the line search acceleration and the load factors for the arc-length and load scaling step size control methods are also printed for each iteration when the **LineSearch**, **ArcLength**, and **LoadScale** options have been included in the incremental load block (see Sections 3.3.2, 3.3.3.1, and 3.3.3.3).

When MERLIN encounters what it considers to be an actual error or a potential error, a message is written indicating what the error is. An actual error is an error that would adversely affect

the execution of the program, eventually causing the program to crash or produce meaningless results. An example of an actual error would be the use of a node in the connectivity definition that does not appear in the coordinate definition. Program execution is terminated prematurely when an actual error is encountered. A potential error is an error that may or may not adversely effect the execution of the program. An example of a potential error would be a node belonging to two or more element groups, which is perfectly valid if both groups have identical material constants but different thicknesses. Messages for actual and potential errors begin with a line containing a message indicator and a message number. The message indicator is ERROR for an actual error and WARNING for a potential error. The message number is a unique four digit integer including leading zeroes. A summary of the actual and potential errors, listed according to their indicators and numbers, is included in Appendix D.

Incremental results written to the formatted output file are reported as nodal quantities only. It is not possible to print integration point quantities, even when the **DispMethod** option (see Section 1.3.1) has been included in the program control block of the input file. Nodal quantities for secondary unknowns such as strain, stress, or flux are computed using the variational projection techniques of the mixed-iterative method regardless of whether it is being used. Incremental results for each unknown field variable, primary, intermediate, or flux-like, are printed in tabular form with each line containing the information for one node. The table is preceded by an informational message indentifying the unknown; a legend for the table column labels; and the table column labels. The table column labels are intentionally cryptic as they must be ten characters or less in length, so the legend is included to provide a more explicit description of the quantities in the table. Each line in the table begins with the node number; the remaining columns contain floating point values. The node number is an integer value that is printed using a field width of five. Floating point values in the table are printed in exponential form using a field width of ten.

4.2 Unformatted Output File

The unformatted output file contains all the information necessary to restart an analysis that has been suspended in progress (see Section 1.1.3). Regardless of the analysis type, the results for each increment always appear in the unformatted output file. For nonlinear fracture mechanics analyses, the unformatted output file also contains the updated mesh topology for each increment. If MERLIN was to be used in conjunction with a program performing adaptive mesh refinement, the updated mesh topology generated by this program would need to be included in the unformatted output file. Although the unformatted output file was designed to facilitate analysis restart, its primary function is data transfer for graphical post-processing.

The unformatted output file is written and read using the buffered I/O utilities provided in the C programming language. As MERLIN is written in FORTRAN, these utilities are accessed through a library of FORTRAN callable C functions. This function library is included in the MERLIN `utils` object library and discussions for each of the functions in the library are included in the MERLIN PROGRAMMER'S MANUAL.

The use of a binary file for analysis restart has four benefits: machine precision of all quantities is maintained; minimum disk space is required; disk access is decreased; and random positioning within the file is enabled. The availability of machine precision quantities is critical to the success of a restarted analysis. Anything less than machine precision would tend to cause the solution of a restarted analysis to "drift" away from the solution of an analysis that was not

restarted. In a binary file, the space required to store a number in the file is identical to the space required to store it in the program. However, in an ASCII file, the space required to store a number in the file is usually greater than the space required to store it in the program. Since fewer words are required to store information in a binary file than in an ASCII file, disk access is decreased with a binary file in comparison to an equivalent ASCII file. In addition, numbers in an ASCII file must be converted from character strings to numerical values while numbers in a binary file do not require conversion, decreasing the time spent performing disk I/O even further. Using the buffered I/O utility functions provided in the C programing language it is possible to move about in a file without reading information from the file. This capability is invaluable for analysis restarts and post-processing the results of multi-increment analyses. In either case, it would be necessary to read a large amount of useless information to arrive at the appropriate location in the file if the ability to move about the file without reading was not available.

Like the input file, the unformatted output file is divided into a number of blocks, each containing a specific type of information. There are three blocks in all: the file header block, the generic finite element information block, and the incremental results block. An unformatted output file contains only one file header block, appearing at the beginning of the file. The file header block is primarily a list of the post variables appearing in the incremental results block. For each increment, there can be an arbitrary number of generic finite element information blocks, but only one incremental result block. The generic finite element information block can be used to store updated mesh definition information or non-nodal analysis results such as stress intensity factors. If no information corresponding to the general description of generic finite element information is required for analysis restart, the generic finite element information block can be omitted from the unformatted file altogether. The incremental analysis block is assembled in a tabular format with the columns corresponding to post variables and the rows corresponding to the nodes. The order of the information in a row of the table is defined in the file header.

Appendix A

PROGRAM EXECUTION

A.1 Merlin

A.1.1 Static Analysis

Merlin can be executed from:

1. The Merlin Tool box.
2. Inside a DOS window and typing

   `merlin fn.inp fn.out fn.pst`

   where

   - `fn.inp` is a text file containing the input data for the finite element analysis. Default name for the input file is `input.dat`.
   - `fn.out` is a text file, where the results of the finite element analysis will be stored. Default name of the output file is `output.dat`.
   - `fn.pst` is a binary file containing the results of the finite element analysis in a format that can be used either to restart the finite element analysis or to graphically visualize the results using the post-processing program Post-MERLIN. Default name for post file is `post.dat`.
3. From a `run.bat` (batch file) which contains the appropriate DOS commands.

A.1.1.1 Dynamic Analysis

A dynamic analysis requires a sequence of two analysis, a static one, and then a dynamic one.

Following the static analysis of `fn1.in`, `fn1.pst`must be copied into a new file `fn2.pst` where `fn2.in` is the dynamic input file. A sample batch file which performs such an operation is listed below:

```
c:\merlin fn1.inp fn1.out fn1.pst
copy fn1.pst fn2.pst
c:\merlin fn2.inp fn2.out fn2.pst
```

A.2 Merlin-MPI

The parallel version of merlin is called `merlin-mpi`. It can be run on a single CPU computer, multiple-CPU computer or a computer network.

Great care should be exercised in generating the input files for Merlin-mpi through `kumonosu`.

A.2.1 Initial Setup (done only once)

1. On each computer create identical set of directories
   (a) For Merlin executable (such as `c:\Merlin`) where `Merlin.exe` and various `mpd` files will have to reside.
   (b) For data files (such as `c:\temp`)
2. All computers must be in the same workgroup;
   (a) Click **Start**, point to **Settings**, click **Control Panel**, and then double-click **System**.
   (b) On the **Computer Name** tab, click **Change**.
   (c) In Computer **name**, type your computer name. The computer name must be unique. You cannot use a name already in use on the network.
   (d) In **Workgroup**, type the workgroup name.
3. All computers must have the same user configuration (not sure about password).
4. Optional
   (a) You may check that the computers are on the same workgroup by: Select **Start**, **Search**, **Files or Folders**, **Other Search Options**, **Computer or People**, **A Computer on the network**, **Search**. No need to put any text in the box. Search will identify computers on the local network sharing the same Workgroup.
   (b) We can run `mpiconfig` and press select. It will autodetect the machines that are on the network. We should be able to then quit `mpiconfig` and run `guiMPIrun` (next step) and have all the machines listed on the right.

A.2.2 Execution

1. Open a DOS window inside the Merlin directory, and type `mpd -d` this will place the computer in "wait" mode; Repeat this operation on each of the computers which will be used.
2. Open another DOS window inside the Merlin directory and type

   ```
   mpirun -np 2 -dir c:\Temp c:\merlin\merlin-mpi.exe file.inp file.out file.pst
   ```

   where:
   (a) `-np n` is the number of processors.

(b) `-dir` is the directory containing the input data files.

(c) `File.inp` is the merlin input file generated by Kumo. Note that this particular file does not exist, however `file.inp.001` `file.inp.002` etc. should be placed inside all computers.

Appendix B

USER SUBROUTINES

Many of MERLIN's capabilities have been intentionally implemented using the most elementary approaches. For example, the input data for surface tractions, conduction heat fluxes, and convection heat fluxes is based on the assumption that these quantities are constants for a given element surface. This assumption may simplify the definition of input data, but it also excludes a large number of common "load" conditions such as surface tractions due to hydrostatic loads. The intent of the user subroutines is to provide general capabilities without complicating the input for the most basic cases. The user subroutines for "loads" typically would compute the "load" at a point based on the coordinates of the point (i.e., the "load" is a function of the coordinates) and perhaps the "load" definitions in the input file. In many cases, the user subroutine would replace a separate computer program used to generate the general data that normally would appear in the input file, eliminating a particularly unwieldy step in the preparation of the input file.

In this chapter, each of the user subroutines provided in MERLIN will be discussed separately. The discussions will include a description of the intended capabilities; a description of the default capabilities; a description of the variables in the argument list; and guidelines for proper usage. The user subroutines delivered with MERLIN include code that provides certain basic capabilities. These capabilities may be useful as is, but the primary purpose for including the code is to illustrate the proper usage of the user subroutines. The importance of following the guidelines for proper usage cannot be overemphasized. Failure to follow these guidelines could lead to the computation of erroneous results or cause the program to crash. Situations in which failure to follow these guidelines could cause the program to crash will be identified specifically.

B.1 User Subroutine UFCMLW

User subroutine `ufcmlw` computes the resisting stresses and tangent stiffness matrix for the Fictitious Crack Model (FCM) given the material properties and the crack displacements. The resisting stresses and the crack displacements are related through softening laws. The tangent stiffness matrix is simply the first derivative of the softening laws with respect to the crack displacements. The classical implementation of the FCM (**?**) has only one softening law, relating the normal closing stress to the crack opening displacement (COD) in a linear or bilinear fashion. The code provided with `ufcmlw` is for the Continuous Function Model (**?**), which uses an exponential softening law to define the relationship between the normal closing

stress and the COD and assumes that there are no resisting stresses in the tangential directions. The Continuous Function Model also allows for cyclic loading, including hysteretic behavior. The displacements tangent to the crack surface are passed into `ufcmlw` along with the COD, allowing for additional softening laws defining the resisting stresses tangent to the crack surfaces in terms of crack displacements. The user is not prohibited from using more than one crack displacement component in the definition of a softening law defining a particular resisting stress component.

The argument list for `ufcmlw` includes twelve arguments: eight arguments for passing input data, three arguments for passing output data, and one argument for passing input/output data. The arguments for passing input data appear at the beginning of the argument list, followed by the arguments for passing output data. The eight arguments for passing input data, listed in order of appearance in the argument list, are as follows:

`nelprp`: The number of material properties.

`nelstr`: The number of crack displacement and resisting stress components.

`melstr`: The maximum number of crack displacement and resisting stress components.

`nstvar`: The number of state variables.

`matprp`: Material properties.

`epstot`: Total crack displacements (excluding the current increment).

`epsinc`: Incremental crack displacements.

`sigtot`: Total resisting stresses (excluding the current increment).

Arguments `nelprp`, `nelstr`, `melstr`, and `nstvar` are scalar integers and arguments `matprp`, `epstot`, `epsinc`, and `sigtot` are floating point vectors. The value of `nelprp` defines the length of `matprp` and the value of `nelstr` defines the length of `epstot`, `epsinc`, and `sigtot`. Arguments for input data should never be modified by the user; modification of these variables could lead to a program crash. The argument for passing input/output data is as follows:

`state`: Current/updated state variables.

Argument `state` is a floating point vector. The value of `nstvar` defines the length of `state`. On input argument `state` contains the current state variables and on output argument `state` contains the updated state variables. The three arguments for passing output data, listed in order of appearance in the argument list, are as follows:

`stfflg`: A flag indicating a change of stiffness associated with unloading or reloading.

`sigtrl`: Updated total resisting stresses.

`dtngnt`: Tangent stiffness matrix.

Argument `stfflg` is an integer scalar; argument `sigtrl` is a floating point vector; and argument `dtngnt` is a two-dimensional floating point matrix. The value of `nelstr` defines the length of `sigtrl` and the value of `melstr` defines both dimensions of `dtngnt`. The value of `stfflg` should

be set to `-1` if unloading from the softening law is detected; `1` if reloading after unloading is detected; and `0` there is no change from the current loading condition. By default, `sigtrl` and `dtngnt` are assigned values based on the material properties `matprp` and the current total crack displacements (i.e. the sum of vectors `epstot` and `epsinc`) using the Continuous Function Model (**?**).

When using `ufcmlw`, the user should pay close attention to the functions used to define softening laws; the shape functions defining the stress fields for the interface elements in the fictitious crack; and the mesh discretization near the fictitious crack. If a softening law should include a region of extreme nonlinearity, great care should be taken when using lower order interface elements in the fictitious crack. Too coarse a mesh near the fictitious crack will adversely effect the accuracy of the computed results regardless of the softening law or assumed stress fields in the interface elements.

B.2 User Subroutine UTRACT

User subroutine `utract` computes the normal and tangential tractions on an element surface given the element nodal coordinates, the element surface definition, a point on the element surface, and the prescribed surface tractions. Typically, the surface tractions are a function of the coordinates of the point on the element surface. However, the prescribed surface tractions can be used as variables in the definition of the surface tractions. The code provided with `utract` defines a hydrostatic loading using the first two components of the prescribed surface tractions to define the elevation of the fluid and the unit weight of the fluid.

The argument list for `utract` includes eight arguments: seven arguments for passing input data and one argument for passing output data. The arguments for passing input data appear at the beginning of the argument list, followed by the arguments for passing output data. The seven arguments for passing input data, listed in order of appearance in the argument list, are as follows:

`nelcrd`: The number of coordinates per node.

`nelnod`: The number of nodes defining the element.

`nelsnd`: The number of nodes defining the element surface.

`elmcrd`: Element nodal coordinates.

`elmsrf`: Element node numbers defining the element surface.

`pntcrd`: Coordinates of a point on the element surface.

`tracts`: Prescribed surface tractions.

Arguments `nelcrd`, `nelnod`, and `nelsnd` are integer scalars; argument `elmcrd` is a floating point matrix; argument `elmsrf` is an integer vector; and arguments `pntcrd` and `tracts` are floating point vectors. The values of `nelcrd` and `nelnod` define the dimensions of `elmcrd`; the value of `nelsnd` defines the length of `elmsrf`; and the value of `nelcrd` defines the length of `pntcrd` and `tracts`. Arguments for input data should never be modified by the user; modification of these variables could lead to a program crash. The argument for passing output data is as follows:

`usrtrc`: Computed surface tractions.

Argument `usrtrc` is a floating point vector and the value of `nelcrd` defines the length of `usrtrc`. By default, the normal component of `usrtrc` is assigned a value equal to the product of the fluid unit weight and the difference between the fluid surface elevation and the y-coordinate of the point, provided the point is not above the fluid surface; all other components of `usrtrc` are zero. The normal component of `usrtrc` is also zero if the point is above the fluid surface.

When using `utract`, the user should pay close attention to the functions used to define the surface tractions; the shape functions defining the stress field over the element domains; and the mesh discretization. If the functions defining the surface tractions are of a higher order than the shape functions defining the stress field over the element domains, great care should be taken when constructing the mesh. Too coarse a mesh in a region of extreme nonlinearity of the surface tractions will adversely effect the accuracy of the computed results regardless of the order of the shape functions defining the stress field over the element domains.

Note that when using `utract`, the user can access two user defined arguments for 2D through the `Tractions` options, and six in 3D.

B.3 User Subroutine UBODY

User subroutine `ubody` computes the body forces given the mass density, the prescribed acceleration vector, and a point within the element. The prescribed acceleration vector is that defined in the input file and the point within the element corresponds to a node. Typically, the body forces are a function of the coordinates of the point within the element. However, the mass density and the prescribed acceleration vector can be used as variables in the definition of the body forces. The code provided with `ubody` computes a body force vector assuming that the mass density and the prescribed acceleration are constant over the element and the direction of the acceleration vector does not change. Using `ubody` without modifying the source code is the same as using the standard body force capabilities provided in the program.

The argument list for `ubody` includes five arguments: four arguments for passing input data and one argument for passing output data. The arguments for passing input data appear at the beginning of the argument list, followed by the arguments for passing output data. The seven arguments for passing input data, listed in order of appearance in the argument list, are as follows:

`nelcrd`: The number of coordinates per node.

`masden`: Mass density.

`accel`: Prescribed acceleration vector.

`pntcrd`: Coordinates of a point within the element.

Arguments `nelcrd` is an integer scalar; argument `masden` is a floating point scalar; and arguments `accel` and `pntcrd` are floating point vectors. The value of `nelcrd` defines the length of `accel` and `pntcrd`. Arguments for input data should never be modified by the user; modification of these variables could lead to a program crash. The argument for passing output data is as follows:

`bdyfrc`: Computed body forces.

Argument `bdyfrc` is a floating point vector and the value of `nelcrd` defines the length of `bdyfrc`. By default, `bdyfrc` is assigned the value of the scalar-vector product of `masden` and `accel`.

When using `ubody`, the user should pay close attention to the functions used to define the body forces; the shape functions defining the stress field over the element domains; and the mesh discretization. If the functions defining the body forces are of a higher order than the shape functions defining the stress field over the element domains, great care should be taken when constructing the mesh. Too coarse a mesh in a region of extreme nonlinearity of the body forces will adversely effect the accuracy of the computed results regardless of the order of the shape functions defining the stress field over the element domains.

B.4 User Subroutine UFLUX

User subroutine `uflux` computes the conduction heat flux on an element surface given the element nodal coordinates, the element surface definition, a point on the element surface, and the prescribed conduction heat flux. The point on the element surface generally corresponds to an integration point and the prescribed conduction heat flux is that defined in the input file. Typically, the conduction heat flux is a function of the coordinates of the point on the element surface. However, the prescribed conduction heat flux can be used as variables in the definition of the conduction heat flux. The code provided with `uflux` simply returns the prescribed conduction heat flux as the user defined value. Using `uflux` without modifying the source code is the same as using the standard conduction heat flux capabilities provided in the program.

The argument list for `uflux` includes eight arguments: seven arguments for passing input data and one argument for passing output data. The arguments for passing input data appear at the beginning of the argument list, followed by the arguments for passing output data. The seven arguments for passing input data, listed in order of appearance in the argument list, are as follows:

`nelcrd`: The number of coordinates per node.

`nelnod`: The number of nodes defining the element.

`nelsnd`: The number of nodes defining the element surface.

`elmcrd`: Element nodal coordinates.

`elmsrf`: Element node numbers defining the element surface.

`pntcrd`: Coordinates of a point on the element surface.

`flux`: Prescribed conduction heat flux.

Arguments `nelcrd`, `nelnod`, and `nelsnd` are integer scalars; argument `elmcrd` is a floating point matrix; argument `elmsrf` is an integer vector; argument `pntcrd` is a floating point vector; and argument `flux` is a floating point scalar. The values of `nelcrd` and `nelnod` define the dimensions of `elmcrd`; the value of `nelsnd` defines the length of `elmsrf`; and the value of `nelcrd` defines the length of `pntcrd`. Arguments for input data should never be modified by the user; modification

of these variables could lead to a program crash. The argument for passing output data is as follows:

`usrflx`: Computed conduction heat flux.

Argument `usrflx` is a floating point scalar. By default, `usrflx` is assigned the value of `flux`.

When using `uflux`, the user should pay close attention to the functions used to define the conduction heat flux; the shape functions defining the flux field over the element domains; and the mesh discretization. If the function defining the conduction heat flux is of a higher order than the shape functions defining the flux field over the element domains, great care should be taken when constructing the mesh. Too coarse a mesh in a region of extreme nonlinearity of the conduction heat flux will adversely effect the accuracy of the computed results regardless of the order of the shape functions defining the flux field over the element domains.

B.5 User Subroutine UFILM

User subroutine `ufilm` computes the fluid temperature and film coefficient defining a convection heat flux on an element surface given the element nodal coordinates, the element surface definition, a point on the element surface, the prescribed fluid temperature, and the prescribed film coefficient. The point on the element surface generally corresponds to an integration point and the prescribed fluid temperature and film coefficient are those defined in the input file. Typically, the convection heat flux is a function of the coordinates of the point on the element surface. However, the prescribed fluid temperature and film coefficient can be used as variables in the definition of the convection heat flux. The code provided with `ufilm` simply returns the prescribed fluid temperature and film coefficient as the user defined values. Using `ufilm` without modifying the source code is the same as using the standard convection heat flux capabilities provided in the program.

The argument list for `ufilm` includes ten arguments: eight arguments for passing input data and two arguments for passing output data. The arguments for passing input data appear at the beginning of the argument list, followed by the arguments for passing output data. The eight arguments for passing input data, listed in order of appearance in the argument list, are as follows:

`nelcrd`: The number of coordinates per node.

`nelnod`: The number of nodes defining the element.

`nelsnd`: The number of nodes defining the element surface.

`elmcrd`: Element nodal coordinates.

`elmsrf`: Element node numbers defining the element surface.

`pntcrd`: Coordinates of a point on the element surface.

`temp`: Prescribed fluid temperature.

`film`: Prescribed film coefficient.

Arguments `nelcrd`, `nelnod`, and `nelsnd` are integer scalars; argument `elmcrd` is a floating point matrix; argument `elmsrf` is an integer vector; argument `pntcrd` is a floating point vector; and arguments `temp` and `film` are floating point scalars. The values of `nelcrd` and `nelnod` define the dimensions of `elmcrd`; the value of `nelsnd` defines the length of `elmsrf`; and the value of `nelcrd` defines the length of `pntcrd`. Arguments for input data should never be modified by the user; modification of these variables could lead to a program crash. The arguments for passing output data, listed in order of appearance in the argument list, are as follows:

`usrtmp:` Computed fluid temperature.

`usrflm:` Computed film coefficient.

Arguments `usrtmp` and `usrflm` are floating point scalars. By default, `usrtmp` is assigned the value of `temp` and `usrflm` is assigned the value of `film`.

When using `ufilm`, the user should pay close attention to the functions used to define the fluid temperature and film coefficient; the shape functions defining the flux field over the element domains; and the mesh discretization. If the functions defining the fluid temperature and film coefficient are of a higher order than the shape functions defining the flux field over the element domains, great care should be taken when constructing the mesh. Too coarse a mesh in a region of extreme nonlinearity of the fluid temperature and film coefficient will adversely effect the accuracy of the computed results regardless of the order of the shape functions defining the flux field over the element domains.

B.6 User Subroutine UHEAT

User subroutine `uheat` computes the internally generated heat (i.e., body heat) given the prescribed internally generated heat and a point within the element. The prescribed internally generated heat is that defined in the input file and the point within the element corresponds to a node. Typically, the internally generated heat is a function of the coordinates of the point within the element. However, the prescribed internally generated heat can be used as a variable in the definition of the internally generated heat. The code provided with `uheat` simply returns the prescribed internally generated heat as the user defined value. Using `uheat` without modifying the source code is the same as using the standard internally generated heat capabilities provided in the program.

The argument list for `uheat` includes four arguments: three arguments for passing input data and one argument for passing output data. The arguments for passing input data appear at the beginning of the argument list, followed by the arguments for passing output data. The seven arguments for passing input data, listed in order of appearance in the argument list, are as follows:

`nelcrd:` The number of coordinates per node.

`heat:` Prescribed internally generated heat.

`pntcrd:` Coordinates of a point within the element.

Arguments `nelcrd` is an integer scalar; argument `heat` is a floating point scalar; and argument `pntcrd` is a floating point vector. The value of `nelcrd` defines the length of `pntcrd`. Arguments

for input data should never be modified by the user; modification of these variables could lead to a program crash. The argument for passing output data is as follows:

`usrhet:` Computed internally generated heat.

Argument `usrhet` is a floating point scalar. By default, `usrhet` is assigned the value of `heat`.

When using `uheat`, the user should pay close attention to the functions used to define the internally generated heat; the shape functions defining the flux field over the element domains; and the mesh discretization. If the functions defining the internally generated heat are of a higher order than the shape functions defining the flux field over the element domains, great care should be taken when constructing the mesh. Too coarse a mesh in a region of extreme nonlinearity of the internally generated heat will adversely effect the accuracy of the computed results regardless of the order of the shape functions defining the flux field over the element domains.

Appendix C

GLOSSARY

Displacement correlation method: A method for computing stress intensity factors in which the crack tip is surrounded by 6-node triangles or 8-node quadrilaterals with mid-side nodes on element edges radiating from the crack tip node moved toward the crack tip node to the quarter-point position (i.e., singular elements). The displacements from the nodes on the crack surface for the singular elements are inserted in the analytical solution for the displacement field in the neighborhood of the crack tip to compute the stress intensity factors.

Volume integral method: A method for computing stress intensity factors in which an energy quantity is computed using a volume integral in the neighborhood of the crack tip. Auxiliary solutions or functions with special characteristics may appear in the integrand. The stress intensity factors can be computed from the energy quantities or an energy criterion can be used to detect crack propagation.

Contour integral method: A method for computing stress intensity factors in which an energy quantity is computed using a line integral that begins on one crack surface and ends on the opposite crack surface surrounding the crack tip. This line integral is supplemented by an additional line integral on the crack surfaces when the crack surfaces are loaded and a volume integral over the area inside the line integral when body forces are present. Auxiliary solutions or functions with special characteristics may appear in the integrands. The stress intensity factors can be computed directly from the integral or indirectly from the energy quantities. Either the stress intensity factors or an energy criterion can be used to detect crack propagation.

Mixed-iterative methods: Mixed finite element methods result in a system of equations involving two or more field variables as unknowns. There is limited coupling between field variables in these mixed systems of equations, resulting in very sparse coefficient matrices. Unfortunately, if direct solution methods are applied to these sparse coefficient matrices a significant amount of terms are added, thus destroying the sparsity and making the solution prohibitively expensive. If an iterative solution strategy in which the mixed system of equations is solved in a staggered fashion the cost of obtaining a solution is greatly reduced, making it possible to obtain the accuracy of a mixed without much of the associated overhead.

Variational principles: A variational principle is an integral equation defining a scalar value

in terms of one or more independent variables (i.e. the field variables). The integral equation is called a functional and for the majority of variational principles the scalar value defined by the functional is an energy quantity. For the variational principle to be useful the functional must possess a stationary point in the solution space where small changes in the independent variables (i.e. variations of the field variables) produce no change in the scalar value defined by the functional. Stationary points are located by taking the first variation of the functional with respect to each of the independent variables and setting this scalar quantity equal to zero; the first variation of the functional is called a variational statement or weak variational form. If a stationary point exists for the functional, there are well established techniques available for obtaining approximate solutions using the finite element method. A useful feature of variational principles is that the discrete form yields a symmetric system of equations.

Primary field variable: The field variable that is differentiated in the field equations is the primary field variable. For stress analyses the primary field variable is the displacement; for heat transfer analyses the primary field variable is the temperature; and for seepage flow analyses the primary field variable is the head.

Intermediate field variable: The differentiated primary field variable in the field equations is the intermediate field variable. For stress analyses the intermediate field variable is the strain; for heat transfer analyses the intermediate field variable is the temperature gradient; and for seepage flow analyses the intermediate field variable is the head gradient.

Flux-like field variable: The field variable that is the product of an operator matrix defined by the material properties and the intermediate field variable is the flux-like field variable. For stress analyses the flux-like field variable is the stress; for heat transfer analyses the flux-like field variable is the heat flux; and for seepage flow analyses the flux-like field variable is the flux.

Essential boundary conditions: Boundary conditions in which the value of the primary field variable are prescribed are essential boundary conditions. In finite element analysis, essential boundary conditions are required to obtain a non-singular global stiffness matrix.

Natural boundary conditions: Boundary conditions in which the value of the flux-like field variable are prescribed are natural boundary conditions. In finite element analysis, natural boundary conditions are not required, but the computed primary field variable will be zero at all nodes in the mesh if non-zero essential boundary conditions or some form of body "loads" are not prescribed.

Reactions: The result of integrating the product of the transpose of the differential operator matrix used to obtain the intermediate field variable and the flux-like field variable over the element domain. For stress analyses, the reaction vector for an element is defined as

$$\mathbf{p}_e \;=\; \int_{\Omega_e} \mathbf{B}^T \boldsymbol{\sigma}\, d\Omega \tag{3.1}$$

and for heat transfer and seepage flow analyses the reaction vector is defined as

$$\mathbf{p}_e \;=\; \int_{\Omega_e} \mathbf{B}^T \mathbf{q}\, d\Omega. \tag{3.2}$$

Element load vectors are summed to create a global reaction vector, which should have non-zero elements for only those degrees of freedom with prescribed essential or natural boundary conditions in the absence of body forces.

Residual loads: The difference between the applied loads and the reactions are the residual loads. The residual loads are a measure of the error in the computed solution. For linear elastic, displacement method analyses there should be no difference between the applied loads and the reactions, which means that the residual load vector would be a null vector. However, for linear elastic analyses performed using the mixed-iterative method or nonlinear analyses there are rarely differences between the applied loads and the reactions. In fact, the object of these analyses is to minimize, within prescribed tolerances, the residual loads.

Constraint variables: When interfaces between materials are treated using constraint equations on the primary field variable, a system of mixed equations with two field variables results. In this system of mixed equations the primary field variable is called the constrained variable and the other field variable is the constraint variable. The constraint variable is the product of the flux-like field variable and a normal vector. For stress analyses the constraint variables are surface tractions and for heat transfer and seepage flow the constraint variables are normal fluxes.

State variables: Nonlinear material models often require parameters other than the material properties and the intermediate field variable to define the flux-like field variable. These extra parameters are called state variables. An example of a state variable would be the equivalent plastic strain.

Preconditioner: A matrix used to compute an estimate of an unknown vector in iterative solution methods. The preconditioner should be less complex than the coefficient matrix for the system of equations whose solution it is estimating while retaining the most dominant of its characteristics. The goal is to balance the cost of obtaining the estimate with the quality of the estimate.

Conjugate gradient: An iterative solution method for linear systems of equations in which the solution is updated in the direction of the steepest gradient in the solution space. For well conditioned coefficient matrices, the cost of obtaining a solution using this method can be much less than that of a direct solution method, especially if the specified convergence tolerance is "loose". Typically, preconditioners are introduced to accelerate convergence, with the form of the preconditioner being dependent on the characteristics of the coefficient matrix.

Euclidean norm: The length of a vector, computed as the square root of the sum of the vector components squared

$$\| \mathbf{v} \|_2 = \sqrt{\sum_{i=1}^{N} v_i^2} \tag{3.3}$$

is called the Euclidean norm of the vector. In two- and three-dimensions this concept is quite simple to visualize, but for vectors defined in an N-dimensional space it is not, even though the operation is the same. The quotient of the Euclidean norms of the residual loads and the reactions is a measure of the overall error in the solution.

Infinity norm: The maximum absolute value of the vector components

$$\| \mathbf{v} \|_\infty = \max(\sum_{i=1}^{N} \text{abs}(v_i)) \tag{3.4}$$

is called the infinity norm of the vector. The quotient of the infinity norms of the residual loads and the reactions is a measure of the component error in the solution.

Spectral radius: The maximum eigenvalue of a matrix is called the spectral radius. The spectral radius is used to compare the convergence characteristics of the iterative strain projection algorithms. The spectral radius must be less than one to insure convergence and the smaller the spectral radius the faster the convergence.

Appendix D

ERRORS, WARNING AND HELP

The error and warning messages written into the formatted output file and the error file include a number that allows for easy identification. In this appendix the situations that would cause each message to be printed and the corrective measures that can be taken by the user to prevent those situations are discussed. In this discussion error and warning messages will be treated separately.

D.1 Error Messages

All error messages written by MERLIN indicate fatal errors that would either cause the program to crash or the computed results to be erroneous if execution were allowed to continue. Error messages are identified by a four digit number with leading zeroes. The error message numbers begin at 0001 and increase sequentially.

Error Message 0001

Error message 0001

```
aaaaaaaaaaaa was not found in the keyword list for this input block.
```

occurs when a keyword read from the input file does not match any of the keywords in the keyword list for the current input block. Misspelling and omitted input data for the previous input option are the most common culprits for this error.

Error Message 0002

Error message 0002

```
Heat transfer analysis cannot be performed concurrently with seepage
flow analysis.
```

occurs when the **HeatTransfer** option appears in the program control block along with the **SeepageFlow** option, but following the **SeepageFlow** option. The program interprets this as a request to perform both a heat transfer and a seepage flow analysis concurrently. Either the **HeatTransfer** option or the **SeepageFlow** option must be removed from the program control block.

Error Message 0003

Error message 0003

```
Seepage flow analsysis cannot be performed concurrently with heat
transfer analysis.
```

occurs when the **SeepageFlow** option appears in the program control block along with the **HeatTransfer** option, but following the **HeatTransfer** option. The program interprets this as a request to perform both a heat transfer and a seepage flow analysis concurrently. Either the **HeatTransfer** option or the **SeepageFlow** option must be removed from the program control block.

Error Message 0004

Error message 0004

```
The number of increments must be greater than zero.
```

occurs when the number of load or time increments specified for the **Increments** option is less than or equal to zero. The number of increments must be positive and non-zero.

Error Message 0005

Error message 0005

```
The increment number for program restart must be greater than zero.
```

occurs when the increment number at which an analysis is to be resumed using the **Restart** option is less than or equal to zero. This increment number must be positive and non-zero.

Error Message 0006

Error message 0006

```
The extraction method must be specified by a number between one and
three, inclusive.
```

occurs when the value for the specified extraction method for the **B-integral** option is either less than 1 or greater than 3. There are only three extraction methods available and they are numbered 1, 2, and 3.

Error Message 0007

Error message 0007

```
The number of discs must be a positive, non-zero number less than 5.
```

occurs when the value for the **B-integral** option that specifies the number of discs (i.e. volumes of integration) to be evaluated is either less than 1 or greater than 5. At least one disc is required, but the program is not capable of handling more than 5 discs.

Error Message 0008

Error message 0008

```
The alpha coeficient for transient analyses must be in the interval
(0.0,1.0).
```

occurs when the specified α-factor for the **Transient** option, which indicates when the equations of motion for transient analyses will be evaluted, is either less that zero or greater than one. Since this value is a fraction, it must be defined on the open interval (0.0,1.0).

Error Message 0009

Error message 0009

```
The number of element groups must be greater than zero.
```

occurs when the number of element groups specified for the **ElementGroup** option is less than or equal to zero. This value must positive and non-zero.

Error Message 0010

Error message 0010

```
Element type iiiiiiiiii is not valid.
```

occurs when the specified element type for an element group is less than one or greater than the number of element types in the element library. Tables 2.6, 2.7 and 2.8 in Section 2.2.2 describe the element types in the element library.

Error Message 0011

Error message 0011

```
Material type iiiiiiiiii is not valid.
```

occurs when the specified material type for an element group is less than one or greater than the number of material models supported by the program. Table 2.2 in Section 2.2 descibes the material models supported by the program.

Error Message 0012

Error message 0012

```
Material type ii cannot be used with element type jj.
```

occurs when the specified material type for an element group cannot be used with the specified element type. Tables 2.3 and 2.4 in Section 2.2 indicates which material models are supported for each element in the element library.

Error Message 0013

Error message 0013

```
The material identifier must be specified as either 0, 1, or 2.
```

occurs when the value for the specified material identifier for an element group is either less than 0 or greater than 2. A value of 0 implies that the material around the crack tip is homogeneous and values of 1 and 2 identify materials on either side of a crack on a bi-material interface.

Error Message 0014

Error message 0014

```
Element type ii is not valid for a heat transfer analysis.
```

occurs when an element type for stress analysis is specified for a heat transfer analysis. Consult Table 2.8 in Section 2.2.2 to determine the appropriate element type.

Error Message 0015

Error message 0015

```
Element type ii is not valid for a seepage flow analysis.
```

occurs when an element type for stress analysis is specified for a seepage flow analysis. Consult Table 2.8 in Section 2.2.2 to determine the appropriate element type.

Error Message 0016

Error message 0016

```
Element type ii is not valid for a stress analysis.
```

occurs when an element type for heat transfer or seepage flow analysis is specified for a stress analysis. Consult Tables 2.6 and 2.7 in Section 2.2.2 to determine the appropriate element type.

Error Message 0017

Error message 0017

```
Element type ii cannot be used with the Mixed-Iterative Method.
```

occurs when the **DispMethod** option does not appear in the program control block for a stress analysis and an element type for which the mixed-iterative method is not supported appears in the element group data. The support status of the mixed-iterative method is given for each element type in Table 2.6 and 2.7.

Error Message 0018

Error message 0018

```
Element type ii appearing in element group jjjjjj cannot be used
with the C-splitting option.
```

occurs when the **C-splitting** option appears in the program control block and an element type for which **C-splitting** option is not supported appears in the element group data.

Error Message 0019

Error message 0019

```
The ElementGroup option must appear prior to the Connectivity option.
```

occurs when the **Connectivity** option appears before the **ElementGroup** option in the mesh definition block. The **ElementGroup** option must appear before the **Connectivity** option because a number of parameters affecting the memory requirements of the program, such as the maximum number of nodes per element, are determined by the element group data.

Error Message 0020

Error message 0020

```
The number of elements must be greater than zero.
```

occurs when the number of elements specified for the **Connectivity** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0021

Error message 0021

```
Element iiiii appears in the position where element jjjjj should
appear.
```

occurs when elements are input out of order in the **Connectivity** option. Elements must be numbered from one to the specified number of elements, with the elements appearing in ascending order.

Error Message 0022

Error message 0022

```
Element group iiiii used with element jjjjj is not valid.
```

occurs when the specified element group number for an element is less than one or greater than the number of element groups specified in the **ElementGroups** option.

Error Message 0023

Error message 0023

```
Inconsistent definition of interface element connectivity detected:
node iiiii appears on surface 1 of element jjjjj and on surface 2 of
element kkkkk.
```

occurs when a node appears on surface 1 of one interface element and on surface 2 of another element. Interface elements must be defined such that surface 1 for all elements is on one surface of the interface and surface 2 for all elements is on the other surface of the interface.

Error Message 0024

Error message 0024

```
The ElementGroup option must appear prior to the Coordinates option.
```

occurs when the **Coordinates** option appears before the **ElementGroup** option in the mesh definition block. The **ElementGroup** option must appear before the **Coordinates** option because a number of parameters affecting the memory requirements of the program, such as the maximum number of coordinates per node, are determined by the element group data.

Error Message 0025

Error message 0025

```
The number of nodes must be greater than zero.
```

occurs when the number of nodes specified for the **Coordinates** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0026

Error message 0026

```
Node iiiii appears in the position where node jjjjj should appear.
```

occurs when nodes are input out of order in the **Coordinates** option. Nodes must be numbered from one to the specified number of nodes, with the nodes appearing in ascending order.

Error Message 0027

Error message 0027

```
The number of duplicate node pairs must be greater than zero.
```

occurs when the number of duplicate node pairs specified for the **Master/Slave** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0028

Error message 0028

```
Node iiiii was used as a master node in duplicate node pair jjjjj.
```

occurs when the node number for the master node of a duplicate node pair is less than one or greater than the number of nodes specified in the **Coordinates** option.

Error Message 0029

Error message 0029

```
Node iiiii was used as a slave node in duplicate node pair jjjjj.
```

occurs when the node number for the slave node of a duplicate node pair is less than one or greater than the number of nodes specified in the **Coordinates** option.

Error Message 0030

Error message 0030

```
The ElementGroup option must appear prior to the CrackSurface option.
```

occurs when the **CrackSurface** option appears before the **ElementGroup** option in the mesh definition block. The **ElementGroup** option must appear before the **CrackSurface** option because a number of parameters effecting the memory requirements of the program, such as the maximum number of nodes per element surface, are determined by the element group data.

Error Message 0031

Error message 0031

```
The Connectivity option must appear prior to the CrackSurface option.
```

occurs when the **CrackSurface** option appears before the **Connectivity** option in the mesh definition block. The **Connectivity** option must appear before the **CrackSurface** option to allow for consistency checks to be made on the element numbers and element surface numbers defining the crack surfaces.

Error Message 0032

Error message 0032

```
The number of discrete cracks must be greater than zero.
```

occurs when the number of discrete cracks specified for the **CrackSurface** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0033

Error message 0033

```
The number of element pairs on the surface of crack iiiii must be
greater than zero.
```

occurs when the specified number of elements pairs defining the surface of a discrete crack is less than or equal to zero. This value must be positive and non-zero.

Error Message 0034

Error message 0034

```
Element iiiii used in crack surface pair jjjjj of crack kkkkk is
invalid.
```

occurs when an element number appearing in a crack surface definition is less than one or greater than the number of elements specified in the **Connectivity** option.

Error Message 0035

Error message 0035

```
Surface iiiii of element jjjjj used in crack kkkkk surface pair lllll
of crack mmmmm is invalid.
```

occurs when an element surface number appearing in a crack surface definition is less than one or greater than the number of surfaces for the element.

Error Message 0036

Error message 0036

```
Surface iiiii of element jjjjj and surface kkkkk of element lllll
used in crack surface pair mmmmm of crack nnnnn are not compatible.
```

occurs when the element surfaces appearing in a crack surface definition do not have the same number of nodes. Elements opposite one another on the crack surface must have the same number of nodes on the element surfaces defining the crack surface.

Error Message 0037

Error message 0037

```
The number of discrete crack fronts must be greater than zero.
```

occurs when the number of discrete cracks specified for the **CrackFronts** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0038

Error message 0038

```
The number of nodes defining crack front must be greater than zero.
```

occurs when the specified number of nodes defining the crack front of a discrete crack is less than or equal to zero. This value must be positive and non-zero.

Error Message 0039

Error message 0039

```
The number of contour paths must be greater than zero.
```

occurs when the number of contour paths specified for the **ContourPath** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0040

Error message 0040

```
Crack iiiii used in contour path jjjjj is invalid.
```

occurs when the crack number specifying about which discrete crack front the contour path is defined is less than one or greater than the number of discrete crack fronts defined in the **Crack Surface** and **CrackFronts** options. This value must be positive and non-zero.

Error Message 0041

Error message 0041

```
The number of element surfaces defining contour path jjjjj must be
greater than zero.
```

occurs when the specified number of element surfaces defining the contour path is less than zero. This value must be positive and non-zero.

Error Message 0042

Error message 0042

```
The mesh definition block must contain an ElementGroup option.
```

occurs when the **ElementGroup** option is not included in the mesh definition block. It is not possible to perform an analysis without this information.

Error Message 0043

Error message 0043

```
The mesh definition block must contain a Connectivity option.
```

occurs when the **Connectivity** option is not included in the mesh definition block. It is not possible to perform an analysis without this information.

Error Message 0044

Error message 0044

```
Node iiiii appearing in the connectivity of element jjjjj is not valid.
```

occurs when a node number appearing in an element connectivity definition is less than one or greater than the number of nodes specified in the **Coordinates** option.

Error Message 0045

Error message 0045

```
The mesh definition block must contain an Coordinates option.
```

occurs when the **Coordinates** option is not included in the mesh definition block. It is not possible to perform an analysis without this information.

Error Message 0046

Error message 0046

```
Node iiiii appearing in duplicate node pair jjjjj is not valid.
```

occurs when a node number appearing in a duplicate node pair definition is less than one or greater than the number of nodes specified in the **Coordinates** option.

Error Message 0047

Error message 0047

```
Node iiiii appearing in the connectivity of (interface) element jjjjj
is  a master node in duplicate node pair kkkkk.
```

occurs when a node defining the connectivity of an interface element appears as a master node in a duplicate node pair. Nodes defining the connectivity of interface elements must always appear as slave nodes in a duplicate node pair.

Error Message 0048

Error message 0048

```
The mesh definition block must contain a CrackSurface option because
the program control block contained an LEFM option.
```

occurs when the **LEFM** option appears in the program control block but the **CrackSurface** option does not appear in the mesh definition block. All of the methods for computing stress intensity factors that are implemented in the program require the topology of the crack surface adjacent to the crack tip to be known.

Error Message 0049

Error message 0049

```
The mesh definition block must contain a CrackFronts option because
the program control block contained an LEFM option.
```

occurs when the **LEFM** option appears in the program control block but the **CrackFronts** option does not appear in the mesh definition block. All of the methods for computing stress intensity factors that are inplemented in the program require the coordinates of the crack tip to be known.

Error Message 0050

Error message 0050

```
The mesh definition block must contain a ContourPath option because
the program control block contained an LEFM option and a J-integral
option.
```

occurs when the **LEFM** and **J-integral** options appear in the program control block but the **ContourPath** option does not appear in the mesh definition block. Without contour paths defined through the mesh it is impossible to compute the stress intensity factors.

Error Message 0051

Error message 0051

```
The mesh definition block must contain a ContourPath option because
the program control block contained an LEFM option and a S-integral
option.
```

occurs when the **LEFM** and **S-integral** options appear in the program control block but the **ContourPath** option does not appear in the mesh definition block. Without contour paths defined through the mesh it is impossible to compute the stress intensity factors.

Error Message 0052

Error message 0052

```
Incremental data separator does not appear prior to increment iiiii.
```

occurs when the incremental data separator does not appear in the unformatted output (i.e., post) file preceding the information for increment `iiiii`. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0053

Error message 0053

```
The mesh is not defined in increment 0 of the post file.
```

occurs when the mesh definition (i.e., the coordinates and connectivity) is not included in the unformatted output (i.e., post) for increment 0, making it impossible to resume an analysis in progress using the **Restart** option in the program control block. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0054

Error message 0054

```
Element group parameters are not defined in the post file.
```

occurs when element group parameters (i.e., the element type, the material model identifier, and the bi-material identifier) are not included in the unformatted output (i.e., post) file, making it impossible to resume an analysis in progress using the **Restart** option in the program control block. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0055

Error message 0055

```
Material properties are not defined in the post file.
```

occurs when the material properties for the element groups are not included in the unformatted output (i.e., post) file, making it impossible to resume an analysis in progress using the **Restart** option in the program control block. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0056

Error message 0056

```
Element connectivity is not defined in the post file.
```

occurs when the element connectivity is not included in the unformatted output (i.e., post) file, making it impossible to resume an analysis in progress using the **Restart** option in the program control block. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0057

Error message 0057

```
Nodal coordinates are not defined in the post file.
```

occurs when the nodal coordinates are not included in the unformatted output (i.e., post) file, making it impossible to resume an analysis in progress using the **Restart** option in the program control block. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0058

Error message 0058

```
Cannot read the mesh definition from the post file.
```

occurs when there is information missing from the unformatted output (i.e., post) file required for a program restart or an I/O error occurs while attempting to read information from the unformatted output file for a program restart. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0059

Error message 0059

```
Node iiiii in duplicate node pair jjjjj is not connected to an element.
```

occurs when a "dangling" node appears as either a master or slave node in a duplicate node pair. All nodes appearing in duplicate pairs must be connected to an element.

Error Message 0060

Error message 0060

```
The maximum number of equilibrium iterations must be greater than
or equal to zero.
```

occurs when the specified maximum number of iterations to be performed in the **Iterations** option is less than zero.

Error Message 0061

Error message 0061

```
Displacement boundary conditions cannot be prescribed for a heat
transfer analysis.
```

occurs when the **DispBCs** option appears in the incremental load block for a heat transfer analysis. The **DispBCs** option is used to prescribe essential boundary conditions for stress analyses.

Error Message 0062

Error message 0062

```
Displacement boundary conditions cannot be prescribed for a seepage
flow analysis.
```

occurs when the **DispBCs** option appears in the incremental load block for a seepage flow analysis. The **DispBCs** option is used to prescribe essential boundary conditions for stress analyses.

Error Message 0063

Error message 0063

```
The number of displacement boundary conditions must be greater than
zero.
```

occurs when the number of displacement boundary conditions specified for the **DispBCs** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0064

Error message 0064

```
Node iiiii is invalid; cannot prescribe displacement boundary condition.
```

occurs when a node number appearing in a displacement boundary condition definition is less than one or greater than the number of nodes specified in the **Coordinates** option.

Error Message 0065

Error message 0065

```
DOF iiiii is invalid; cannot prescribe displacement boundary condition.
```

occurs when a DOF number appearing in a displacement boundary condition definition is less than one or greater than the dimensionality of the mesh. For two-dimensional analyses 1 and 2 are valid DOF numbers and for three-dimensional analyses 1, 2, and 3 are valid DOF numbers.

Error Message 0066

Error message 0066

```
Point loads cannot be prescribed for a heat transfer analysis.
```

occurs when the **PointLoads** option appears in the incremental load block for a heat transfer analysis. The **PointLoads** option is used to prescribe natural boundary conditions for stress analyses.

Error Message 0067

Error message 0067

```
Point loads cannot be prescribed for a seepage flow analysis.
```

occurs when the **PointLoads** option appears in the incremental load block for a seepage flow analysis. The **PointLoads** option is used to prescribe natural boundary conditions for stress analyses.

Error Message 0068

Error message 0068

```
The number of point loads must be greater than zero.
```

occurs when the number of point loads specified for the **PointLoads** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0069

Error message 0069

```
Node iiiii is invalid; cannot prescribe point load.
```

occurs when a node number appearing in a point load definition is less than one or greater than the number of nodes specified in the **Coordinates** option.

Error Message 0070

Error message 0070

```
DOF iiiii is invalid; cannot prescribe point load.
```

occurs when a DOF number appearing in a displacement boundary condition definition is less than one or greater than the dimensionality of the mesh. For two-dimensional analyses 1 and 2 are valid DOF numbers and for three-dimensional analyses 1, 2, and 3 are valid DOF numbers.

Error Message 0071

Error message 0071

```
Surface tractions cannot be prescribed for a heat transfer analysis.
```

occurs when the **Tractions** option appears in the incremental load block for a heat transfer analysis. The **Tractions** option is used to prescribe natural boundary conditions for stress analyses.

Error Message 0072

Error message 0072

```
Surface tractions cannot be prescribed for a seepage flow analysis.
```

occurs when the **Tractions** option appears in the incremental load block for a seepage flow analysis. The **Tractions** option is used to prescribe natural boundary conditions for stress analyses.

Error Message 0073

Error message 0073

```
The number of surface tractions must be greater than zero.
```

occurs when the number of surface tractions specified for the **Tractions** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0074

Error message 0074

```
Element iiiii is invalid; cannot prescribe surface tractions.
```

occurs when an element number appearing in a surface tractions definition is less than one or greater than the number of elements specified in the **Connectivity** option.

Error Message 0075

Error message 0075

```
Vector defining the local coordinate system for the element surface
must have a non-zero length.
```

occurs when the vector defining the local coordinate system on an element surface for the specification of surface tractions for three-dimensional elements is of zero length (i.e., the values for all three components are zeros). At least one of the vector components must be non-zero.

Error Message 0076

Error message 0076

```
Surface iiiii is not defined for element type jjjjj.
```

occurs when the surface number appearing in a surface tractions definition is less than one or greater than the number of element surfaces. Consult Section 2.2.2.2 to determine the number of surfaces for the various element geometries.

Error Message 0077

Error message 0077

```
Surface tractions are not supported for surface iiiii of element
type jjjjj.
```

occurs when surface tractions are not supported for the specified surface of a given element type. Currently, surface tractions are supported for each face of the elements described in Table 2.6 and 2.7 in Section 2.2.2.

Error Message 0078

Error message 0078

```
Uplift pressures cannot be prescribed for a heat transfer analysis.
```

occurs when the **Uplift** option appears in the incremental load block for a heat transfer analysis. The **Uplift** option is used to prescribe natural boundary conditions on crack surfaces due to non-constant hydrostatic pressure for stress analyses.

Error Message 0079

Error message 0079

```
Uplift pressures cannot be prescribed for a seepage flow analysis.
```

occurs when the **Uplift** option appears in the incremental load block for a seepage flow analysis. The **Uplift** option is used to prescribe natural boundary conditions on crack surfaces due to non-constant hydrostatic pressure for stress analyses.

Error Message 0080

Error message 0080

```
The p_W0-COD_W0 option was not included in the program control block.
```

occurs when the **LEFM** option appears in the program control block but the **p_W0-COD_W0** option does not. Without the relationship between p_{W0} and COD_{W0} it is impossible for the program to prescribe the non-constant hydrostatic pressure in terms of the COD.

Error Message 0081

Error message 0081

```
The number of crack surfaces subject to nonlinear uplift pressures must
be greater than zero.
```

occurs when the number of discrete cracks specified for the **Uplift** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0082

Error message 0082

```
Crack iiiii is invalid; cannot prescribe uplift pressure.
```

occurs when a discrete crack number appearing in a uplift pressure definition is less than one or greater than the number of discrete cracks specified in the **CrackSurface** option.

Error Message 0083

Error message 0083

```
Body forces cannot be prescribed for a heat transfer analysis.
```

occurs when the **BodyForce** option appears in the incremental load block for a heat transfer analysis. The **BodyForce** option is used to prescribe body forces due to uniform acceleration for stress analyses.

Error Message 0084

Error message 0084

```
Body forces cannot be prescribed for a seepage flow analysis.
```

occurs when the **BodyForce** option appears in the incremental load block for a seepage flow analysis. The **BodyForce** option is used to prescribe body forces due to uniform acceleration for stress analyses.

Error Message 0085

Error message 0085

```
Centrifugal forces cannot be prescribed for a heat transfer analysis.
```

occurs when the **Centrifugal** option appears in the incremental load block for a heat transfer analysis. The **Centrifugal** option is used to prescribe body forces due to centrifugal loading for stress analyses.

Error Message 0086

Error message 0086

```
Centrifugal forces cannot be prescribed for a seepage flow analysis.
```

occurs when the **Centrifugal** option appears in the incremental load block for a seepage flow analysis. The **Centrifugal** option is used to prescribe body forces due to centrifugal loading for stress analyses.

Error Message 0087

Error message 0087

```
Nodal pressures cannot be prescribed for a heat transfer analysis.
```

occurs when the **Pressures** option appears in the incremental load block for a heat transfer analysis. The **Pressures** option is used to prescribe initial stresses for stress analyses.

Error Message 0088

Error message 0088

```
Nodal pressures cannot be prescribed for a seepage flow analysis.
```

occurs when the **Pressures** option appears in the incremental load block for a seepage flow analysis. The **Pressures** option is used to prescribe initial stresses for stress analyses.

Error Message 0089

Error message 0089

```
The number of nodal pressures must be greater than zero.
```

occurs when the number of nodes specified for the **Pressures** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0090

Error message 0090

```
Node iiiii is invalid; cannot prescribe hydrostatic pressures.
```

occurs when a node number appearing in a hydrostatic pressure definition is less than one or greater than the number of nodes specified in the **Coordinates** option.

Error Message 0091

Error message 0091

```
Nodal temperatures cannot be prescribed for a seepage flow analysis.
```

occurs when the **Temperatures** option appears in the incremental load block for a seepage flow analysis. The **Temperatures** option is used to prescribe initial strains for stress analyses and essential boundary conditions for heat transfer analyses.

Error Message 0092

Error message 0092

```
The number of nodal temperatures must be greater than zero.
```

occurs when the number of nodes specified for the **Temperatures** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0093

Error message 0093

```
Node iiiii is invalid; cannot prescribe temperatures.
```

occurs when a node number appearing in a temperature definition is less than one or greater than the number of nodes specified in the **Coordinates** option.

Error Message 0094

Error message 0094

```
Point fluxes cannot be prescribed for a stress analysis.
```

occurs when the **PointFluxes** option appears in the incremental load block for a stress analysis. The **PointFluxes** option is used to prescribe natural boundary conditions for heat transfer and seepage flow analyses.

Error Message 0095

Error message 0095

```
The number of point fluxes must be greater than zero.
```

occurs when the number of point fluxes specified for the **PointFluxes** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0096

Error message 0096

```
Node iiiii is invalid; cannot prescribe point flux.
```

occurs when a node number appearing in a point flux definition is less than one or greater than the number of nodes specified in the **Coordinates** option.

Error Message 0097

Error message 0097

```
Surface fluxes cannot be prescribed for a stress analysis.
```

occurs when the **Fluxes** option appears in the incremental load block for a stress analysis. The **Fluxes** option is used to prescribe natural boundary conditions for heat transfer and seepage flow analyses.

Error Message 0098

Error message 0098

```
The number of surface fluxes must be greater than zero.
```

occurs when the number of surface fluxes specified for the **Fluxes** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0099

Error message 0099

```
Element iiiii is invalid; cannot prescribe surface flux.
```

occurs when an element number appearing in a surface flux definition is less than one or greater than the number of elements specified in the **Connectivity** option.

Error Message 0100

Error message 0100

```
Surface fluxes are not supported for surface iiiii of element
type jjjjj.
```

occurs when surface fluxes are not supported for the specified surface of a given element type. Currently, surface tractions are supported for each face of the elements described in Table 2.8 in Section 2.2.2.

Error Message 0101

Error message 0101

```
Surface films cannot be prescribed for a seepage flow analysis.
```

occurs when the **Films** option appears in the incremental load block for a seepage flow analysis. The **Films** option is used to prescribe natural boundary conditions for heat transfer analyses.

Error Message 0102

Error message 0102

```
Surface films cannot be prescribed for a stress analysis.
```

occurs when the **Films** option appears in the incremental load block for a stress analysis. The **Films** option is used to prescribe natural boundary conditions for heat transfer analyses.

Error Message 0103

Error message 0103

```
The number of surface films must be greater than zero.
```

occurs when the number of surface films specified for the **Films** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0104

Error message 0104

```
Element iiiii is invalid; cannot prescribe surface film.
```

occurs when an element number appearing in a surface film definition is less than one or greater than the number of elements specified in the **Connectivity** option.

Error Message 0105

Error message 0105

```
Surface films are not supported for surface iiiii of element
type jjjjj.
```

occurs when surface films are not supported for the specified surface of a given element type. Currently, surface tractions are supported for each face of the elements described in Table 2.8 in Section 2.2.2.

Error Message 0106

Error message 0106

```
Body heat cannot be prescribed for a seepage flow analysis.
```

occurs when the **BodyHeat** option appears in the incremental load block for a seepage flow analysis. The **BodyHeat** option is used to prescribe internally generated heat (i.e. body "forces") for heat transfer analyses.

Error Message 0107

Error message 0107

```
Body heat cannot be prescribed for a stress analysis.
```

occurs when the **BodyHeat** option appears in the incremental load block for a stress analysis. The **BodyHeat** option is used to prescribe internally generated heat (i.e. body "forces") for heat transfer analyses.

Error Message 0108

Error message 0108

```
The number of elements with body heat must be greater than zero.
```

occurs when the number of elements specified for the **BodyHeat** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0109

Error message 0109

```
Element iiiii is invalid; cannot prescribe body heat.
```

occurs when an element number appearing in a body heat definition is less than one or greater than the number of elements specified in the **Connectivity** option.

Error Message 0110

Error message 0110

```
Heads cannot be prescribed for a heat transfer analysis.
```

occurs when the **Heads** option appears in the incremental load block for a heat transfer analysis. The **Heads** option is used to prescribe essential boundary conditions for seepage flow analyses.

Error Message 0111

Error message 0111

```
Heads cannot be prescribed for a stress analysis.
```

occurs when the **Heads** option appears in the incremental load block for a stress analysis. The **Heads** option is used to prescribe essential boundary conditions for seepage flow analyses.

Error Message 0112

Error message 0112

```
The number of heads must be greater than zero.
```

occurs when the number of nodes specified for the **Heads** option is less than or equal to zero. This value must be positive and non-zero.

Error Message 0113

Error message 0113

```
Node iiiii is invalid; cannot prescribe head.
```

occurs when a node number appearing in a head definition is less than one or greater than the number of nodes specified in the **Coordinates** option.

Error Message 0114

Error message 0114

```
Indefinite stiffness matrix; factorization failed at equation iiiiii.
```

occurs when a negative number is encountered on the diagonal of the global stiffness matrix during factorization. Once this occurs, program execution is terminated as the solution for a non-positive definite stiffness matrix may not be unique.

Error Message 0115

Error message 0115

```
Singular stiffness matrix; factorization failed at equation iiiiii.
```

occurs when a zero is encountered on the diagonal of the global stiffness matrix during factorization. Once this occurs, program execution is terminated as the solver is equipped to deal with singular matrices. This error is typically the result of missing or improper essential boundary conditions.

Error Message 0116

Error message 0116

```
Arc-length method failed: negative operand for square root detected.
```

occurs when a value for the iterative load factor correction cannot be computed when the **ArcLength** option has been used to specify the indirect displacement control method.

Error Message 0117

Error message 0117

```
Attempt to compute iterative load scale factor correction failed.
```

occurs when a value for the iterative load factor correction cannot be computed when either the **SpecifyCOD** or **LoadScale** options have been used to specify the indirect displacement control method.

Error Message 0118

Error message 0118

J_1 is less than J_2 for contour path ii. Cannot compute K_I and K_{II}.

occurs when the computed value of J_1 is less than the computed value of J_2, making it impossible to obtain a real valued solution for the stress intensity factors.

Error Message 0119

Error message 0119

Cannot compute K_I and K_{II} for contour path ii.

occurs when the pair of nonlinear equations defining the stress intensity factors does not yield a real valued solution.

Error Message 0120

Error message 0120 J_1 `is equal to zero for contour path ii.` occurs when the computed value of J_1 is zero, which indicates that the contour path is not defined around a crack tip or that the structure has not been loaded.

Error Message 0121

Error message 0121

```
Character string read from binary is not null terminated. The string
is aaaaaaaaaaa.
```

occurs when the last character of a character string read from a binary file (i.e., the unformatted output file) is not the NULL character. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0122

Error message 0122

```
Insufficient program memory available to allocate array aaaaaa:
     Number of words required  = iiiiiiiiii
     Number of words available = jjjjjjjjjj
```

occurs when the memory requirements of a problem exceed the available memory in the program.

Error Message 0123

Error message 0123

```
Insufficient program memory available to reallocate array aaaaaa:
     Number of words required  = iiiiiiiiii
     Number of words available = jjjjjjjjjj
```

occurs when the memory requirements of a problem exceed the available memory in the program.

Error Message 0124

Error message 0124

```
Attempt to locate array aaaaaa for purpose of reallocation failed.
```

occurs when an attempt is made to resize an array not stored in program memory. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0125

Error message 0125

```
Reduction of array size is not yet supported; array in question is aaaaaa.
```

occurs when an attempt is made to decrease the size an array stored in program memory. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0126

Error message 0126

```
Attempt to locate array aaaaaa for purpose of returning attributes failed.
```

occurs when an attempt is made to query the attributes of an array not stored in program memory. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0127

Error message 0127

```
Attempt to locate array aaaaaa failed.
```

occurs when an attempt is made to locate an array not stored in program memory. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0128

Error message 0128

```
Row dimension mismatch between arrays aaaaaa and bbbbbb; copy failed.
```

occurs when an attempt is made to copy information from between arrays stored in program memory that do not have the same number of rows. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0129

Error message 0129

```
Column dimension mismatch between arrays aaaaaa and bbbbbb; copy failed.
```

occurs when an attempt is made to copy information from between arrays stored in program memory that do not have the same number of columns. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0130

Error message 0130

```
Page dimension mismatch between arrays aaaaaa and bbbbbb; copy failed.
```

occurs when an attempt is made to copy information from between arrays stored in program memory that do not have the same number of pages (i.e., the third dimension of three-dimensional array). This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0131

Error message 0131

```
Cannot copy to or from a nonexistent memory block.
```

occurs when an attempt is made to copy information to or from an array not stored in program memory. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

Error Message 0132

Error message 0132

```
Attempt to locate array aaaaaa for purpose of deleting it from program
memory failed.
```

occurs when an attempt is made to delete an array not stored in program memory. This error would be the result of a program error (i.e., a bug in the program) rather than a user error (i.e., improper input).

D.2 Warning Messages

Warning messages written by MERLIN indicate either potenial errors or situations in which an option has been disabled or will be ignored. Like error messages, warning messages are identified by a four digit number with leading zeroes. The warning message numbers also begin at 0001 and increase sequentially.

Warning Message 0001

Warning message 0001

```
The number of iterations for strain smoothing must be greater than
zero.  Strain smoothing has been turned off for this analysis.
```

occurs when the maximum number of iterations specified for the **StrainSmooth** option is less than zero. In this situation the **StrainSmooth** option is ignored and the default strain projection algorithm is used.

Warning Message 0002

Warning message 0002

```
The number of iterations for C-splitting must be greater than zero.
Strain smoothing has been turned off for this analysis.
```

occurs when the maximum number of iterations specified for the **C-splitting** option is less than zero. In this situation the **C-splitting** option is ignored and the default strain projection algorithm is used.

Warning Message 0003

Warning message 0003

```
The J-integral option was used without the LEFM option.  The J-integral
option has been turned off for this analysis.
```

occurs when the **J-integral** option appears in the program control block but the **LEFM** option does not. Stress intensity factors are not computed by the program if the **LEFM** option is not present in program control block.

Warning Message 0004

Warning message 0004

```
The S-integral option was used without the LEFM option.  The S-integral
option has been turned off for this analysis.
```

occurs when the **S-integral** option appears in the program control block but the **LEFM** option does not. Stress intensity factors are not computed by the program if the **LEFM** option is not present in program control block.

Warning Message 0005

Warning message 0005

```
The B-integral option was used without the LEFM option.  The B-integral
option has been turned off for this analysis.
```

occurs when the **B-integral** option appears in the program control block but the **LEFM** option does not. Stress intensity factors are not computed by the program if the **LEFM** option is not present in program control block.

Warning Message 0006

Warning message 0006

```
The UFCMLW option was used without the FCM option.  The UFCMLW option
has been turned off for this analysis.
```

occurs when the **UFCMLW** option appears in the program control block but the **FCM** option does not. The ficititious crack model (i.e., material model 6) is not accessable if the **FCM** option is not present in program control block.

Warning Message 0007

Warning message 0007

```
PrintDisp is not valid for heat transfer analyses; it will be ignored.
```

occurs when a **PrintDisp** option appears in the control block for a heat transfer analysis. Displacements can only be printed for stress analyses.

Warning Message 0008

Warning message 0008

```
PrintDisp is not valid for seepage flow analyses; it will be ignored.
```

occurs when a **PrintDisp** option appears in the control block for a seepage flow analysis. Displacements can only be printed for stress analyses.

Warning Message 0009

Warning message 0009

```
PrintStrain is not valid for heat transfer analyses; it will be ignored.
```

occurs when a **PrintStrain** option appears in the control block for a heat transfer analysis. Strains can only be printed for stress analyses.

Warning Message 0010

Warning message 0010

```
PrintStrain is not valid for seepage flow analyses; it will be ignored.
```

occurs when a **PrintStrain** option appears in the control block for a seepage flow analysis. Strains can only be printed for stress analyses.

Warning Message 0011

Warning message 0011

```
PrintStress is not valid for heat transfer analyses; it will be ignored.
```

occurs when a **PrintStress** option appears in the control block for a heat transfer analysis. Stresses can only be printed for stress analyses.

Warning Message 0012

Warning message 0012

```
PrintStress is not valid for seepage flow analyses; it will be ignored.
```

occurs when a **PrintStress** option appears in the control block for a seepage flow analysis. Stresses can only be printed for stress analyses.

Warning Message 0013

Warning message 0013

```
PrintState is not valid for heat transfer analyses; it will be ignored.
```

occurs when a **PrintState** option appears in the control block for a heat transfer analysis. State variables can only be printed for stress analyses.

Warning Message 0014

Warning message 0014

```
PrintState is not valid for seepage flow analyses; it will be ignored.
```

occurs when a **PrintState** option appears in the control block for a seepage flow analysis. State variables can only be printed for stress analyses.

Warning Message 0015

Warning message 0015

```
PrintTemp is not valid for stress analyses; it will be ignored.
```

occurs when a **PrintTemp** option appears in the control block for a stress analysis. Temperatures can only be printed for heat transfer analyses.

Warning Message 0016

Warning message 0016

```
PrintTemp is not valid for seepage flow analyses; it will be ignored.
```

occurs when a **PrintTemp** option appears in the control block for a seepage flow analysis. Temperatures can only be printed for heat transfer analyses.

Warning Message 0017

Warning message 0017

```
PrintFlux is not valid for stress analyses; it will be ignored.
```

occurs when a **PrintFlux** option appears in the control block for a stress analysis. Fluxes can only be printed for heat transfer and seepage flow analyses.

Warning Message 0018

Warning message 0018

```
PrintHead is not valid for stress analyses; it will be ignored.
```

occurs when a **PrintHead** option appears in the control block for a stress analysis. Heads can only be printed for seepage flow analyses.

Warning Message 0019

Warning message 0019

```
PrintHead is not valid for heat transfer analyses; it will be ignored.
```

occurs when a **PrintHead** option appears in the control block for a heat transfer analysis. Heads can only be printed for seepage flow analyses.

Warning Message 0020

Warning message 0020

```
PrintConVar is not valid for heat transfer analyses; it will be ignored.
```

occurs when a **PrintConVar** option appears in the control block for a heat transfer analysis. Constraint variables can only be printed for stress analyses with constrained interface elements.

Warning Message 0021

Warning message 0021

```
PrintConVar is not valid for seepage flow analyses; it will be ignored.
```

occurs when a **PrintConVar** option appears in the control block for a seepage flow analysis. Constraint variables can only be printed for stress analyses with constrained interface elements.

Warning Message 0022

Warning message 0022

```
PrintErrEst is not valid for heat transfer analyses; it will be ignored.
```

occurs when a **PrintErrEst** option appears in the control block for a heat transfer analysis. Error estimates currently can only be printed for stress analyses; they will be made available for heat transfer analyses when the mixed-iterative is extended to support heat transfer analyses.

Warning Message 0023

Warning message 0023

```
PrintErrEst is not valid for seepage flow analyses; it will be ignored.
```

occurs when a **PrintErrEst** option appears in the control block for a seepage flow analysis. Error estimates currently can only be printed for stress analyses; they will be made available for seepage flow analyses when the mixed-iterative is extended to support seepage flow analyses.

Warning Message 0024

Warning message 0024

```
The program control block contained an FCM option but the mesh
definition block did not contain a CrackSurface option.  A summary
of the crack lengths cannot be reported without this information.
```

occurs when the **FCM** option appears in the program control block and the mesh is two-dimensional, but the **CrackSurface** option does not appear in the mesh definition block. Without crack surface definitions the program cannot determine the lengths of the true crack, the fracture process zone, and the fracture ligament or stress, uplift pressure, and relative displacement profiles along the crack surfaces.

Warning Message 0025

Warning message 0025

```
There is an element type conflict at elements iiiii and jjjjj causing
displacement continuity to be violated.  The two elements are not of
the same order.
```

occurs when two elements adjacent to one another in the mesh are not of the same order (e.g. a 4-node quadrilateral and a 6-node triangle). The midside node for the higher order element is free to separate from or penetrate the surface of the neighboring lower order element, which causes a discontinuity in the displacements along the shared element surface.

Warning Message 0026

Warning message 0026

```
There is an element group conflict at node iiiii between groups jjj
and kkk; group jjj controls.
```

occurs when two elements connected to a node do not belong to the same element group. This is acceptable if the elements are of different types but share the same element properties, but in most other situations it is not acceptable.

Warning Message 0027

Warning message 0027

```
TangentStiff option is not functional for heat transfer or seepage flow
analyses.  Your request has been ignored.
```

occurs when the **TangentStiff** option is included in the incremental loads block for a heat transfer or seepage flow analyses. Since nonlinear analyses are not supported for these analyses, this option is currently inactive.

Warning Message 0028

Warning message 0028

```
LineSearch option is not functional for heat transfer or seepage flow
analyses.  Your request has been ignored.
```

occurs when the **LineSearch** option is included in the incremental loads block for a heat transfer or seepage flow analyses. Since nonlinear analyses are not supported for these analyses, this option is currently inactive.

Warning Message 0029

Warning message 0029

```
SecantNewton option is not functional for heat transfer or seepage flow
analyses.  Your request has been ignored.
```

occurs when the **SecantNewton** option is included in the incremental loads block for a heat transfer or seepage flow analyses. Since nonlinear analyses are not supported for these analyses, this option is currently inactive.

Warning Message 0030

Warning message 0030

```
ArcLength option is not functional for heat transfer or seepage flow
analyses.  Your request has been ignored.
```

occurs when the **ArcLength** option is included in the incremental loads block for a heat transfer or seepage flow analyses. Since nonlinear analyses are not supported for these analyses, this option is currently inactive.

Warning Message 0031

Warning message 0031

```
SpecifyCOD option is not functional for heat transfer or seepage flow
analyses.  Your request has been ignored.
```

occurs when the **SpecifyCOD** option is included in the incremental loads block for a heat transfer or seepage flow analyses. Since nonlinear analyses are not supported for these analyses, this option is currently inactive.

Warning Message 0032

Warning message 0032

```
LoadScale option is only valid when interface elements are included in
the mesh.  Your request has been ignored.
```

occurs when the **LoadScale** option is included in the incremental loads block for a heat transfer or seepage flow analyses. Since nonlinear analyses are not supported for these analyses, this option is currently inactive.

Warning Message 0033

Warning message 0033

```
Cannot use secant-Newton update with indirect displacement control methods;
secant-Newton update will not be performed for this increment.
```

occurs when the **SecantNewton** option is included in the incremental loads block for stress analyses along with either the **ArcLength**, **SpecifyCOD**, or **LoadScale** options. The secant-Newton update and the indirect displacement control methods are mutually exclusive operations and, therefore, cannot be used together. The **LineSearch** option should be used to accelerate convergence when using the indirect displacement control methods for step size control.

D.3 Help

Following is a list of errors/problems which are likely to occur, and corresponding remedies:

Osciallating ± stresses in a nonlinear analysis with ICM Option: is an indication that a lack of convergence is caused by **unstable** crack growth.

MERLIN ABORT: (on Unix) is usually caused by **ill-conditioning** of the triangular elements (usually close to the interface elements). Use the `SMOOTHING` option in preMERLIN to improve the characteristics of the elements.

Interface elements are not inserted if the `NLFM` option is not specified in preMERLIN.

Bus ERROR (Unix) occurs if the crack front, surface or master/slave nodes are not correctly defined in the input file.
This error occasionally occurs in preMERLIN for inclined cracks.