USERS MANUAL for the FEHMN APPLICATION

by

George A. Zyvoloski
Bruce A. Robinson
Zora V. Dash
Lynn L. Trease

RB

Modification date: 5/20/96

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1.0 PURPOSE

This User's Manual documents the use of the FEHMN application.

2.0 DEFINITIONS AND ACRONYMS

2.1 Definitions

FEHM - Finite element heat and mass transfer code (Zyvoloski, et al. 1988)

FEHMN - YMP version of FEHM (Zyvoloski, et al. 1992).

Acronyms

AVS - Advanced Visual Systems.

I/O - Input / Output.

LANL - Los Alamos National Laboratory.

UCD - unstructured cell data.

YMP - Yucca Mountain Site Characterization Project.

3.0 REFERENCES

Carslaw, H. S., and J. C. Jaeger, *Conduction of Heat in Solids*, 2nd Edition, Clarendon Press, 1959.

"Model and Methods Summary for the FEHMN Application," Los Alamos document FEHMN MMS (ECD-22, in progress).

Molloy, M. W., "Geothermal Reservoir Engineering Code Comparison Project," *Proceedings of the Sixth Workshop on Geothermal Reservoir Engineering*, Stanford University, 1980.

Zyvoloski, G. A., Z. V. Dash, and S. Kelkar, "FEHM: Finite Element Heat and Mass Transfer Code," LA-11224-MS (1988).

Zyvoloski, G. A., Z. V. Dash, and S. Kelkar, "FEHMN 1.0: Finite Element Heat and Mass Transfer Code," LA-12062-MS (1991).

Zyvoloski, G. A., and Z. V. Dash, "Software Verification Report FEHMN Version 1.0," LA-UR-91-609 (1991).

4.0 PROGRAM CONSIDERATIONS

4.1 Program Options

The uses and capabilities of FEHMN are summarized in Table I with reference to the macro input structure discussed in Section 6.0.

Table I. Capabilities of FEHMN with Macro Command References

I. Mass, energy balances in porous media
  A. Variable rock properties (**rock**)
  B. Variable permeability (**perm**)
  C. Variable thermal conductivity (**cond**)
  D. Variable fracture properties, dual porosity, dual porosity/dual permeability (**dual, dpdp**)

II. Multiple components available
  A. Air-water isothermal mixture available (**airwater**), fully coupled to heat and mass transfer (**ngas**)
  B. Up to 10 solutes with chemical reactions between each (**trac, rxn**)
  C. Several different capillary pressure models (**cap**)
  D. Several different relative permeability models (**rlp**)

III. Equation of state flexibility inherent in code (**eos**)

IV. Psuedo-stress models available
  A. Linear porosity deformation (**ppor**)
  B. Gangi stress model (**ppor**)

V. Numerics
  A. Finite element with multiple element capabilities (**elem**)
  B. Short form input methods available (**coor, elem**)
  C. Flexible properties assignment (**zone**)
  D. Flexible solution methods
    1. Upwinding, implicit solution available (**ctrl**)
    2. Iteration control adaptive strategy (**iter**)
  E. Finite volume geometry (**finv**)

VI. Flexible time step and stability control (**time**)

4.2 Initialization

The coefficient arrays for the polynomial representations of the density (crl, crv), enthalpy (cel, cev), and viscosity (cvl, cvv) functions are initialized to the values enumerated in Table III, Table IV, and Table V of the FEHMN.MMS.doc. Values for the saturation pressure and temperature function coefficients are found in Table VI of the FEHMN.MMS.doc. All other global array and scalar variables, with the exception of the variables listed in Table II, whether integer or real, are initialized to zero.

Table II. Initial (Default) Values

Variable	Value	Variable	Value	Variable	Value
aiaa	1.0	contim	1.0e+30	daymin	1.0e-05
daymax	30.0	g1	1.0e-06	g2	1.0e-06
g3	1.0e-03	iamx	500	ncntr	10000000
nicg	1	rnmax	1.0e+11	str	1.0
strd	1.0	tmch	1.0e-09	tmelt	-1.0e+12
upwgt	1.0	upwgta	1.0		

4.3 Restart Procedures

FEHMN writes a restart file for each run. The restart output file name may be given in the input control file or as terminal input, or if unspecified will default to *fehmn.fin* (see Section 6.2.1 on page 23). The file is used on a subsequent run by providing the name of the generated file (via control file or terminal) for the restart input file name. It is recommended that the restart input file name be modified to avoid confusion with the restart output file. For example, by changing the suffix to *.ini*, the default restart output file, *fehmn.fin* would be renamed *fehmn.ini,* and that file name placed in the control file or given as terminal input. Values from the restart file will overwrite any variable initialization prescribed in the input file. The initial time of simulation will also be taken from the restart file.

4.4 Error Processing

Due to the nonlinearity of the underlying partial differential equations, it is possible to produce an underflow or overflow condition through an unphysical choice of input parameters. More likely the code will fail to converge or will produce results which are out of bounds for the thermodynamic functions. The code will attempt to decrease the time step until convergence occurs. If the time step drops below a prescribed minimum the code will stop, writing a restart file. The user is encouraged to look at the input check file which contains information regarding maximum and minimum values of key variables in the code. All error and warning messages will be output to an output error file.

Table III provides additional information on errors that will cause FEHMN to terminate.

Table III. Error Conditions Which Result in Program Termination

Error Condition	Error Message
I/O file error	

Table III. Error Conditions Which Result in Program Termination

Error Condition	Error Message

Unable to open I/O file	`**** Error opening file` *`fileid`* `****`
	.
	.
	.
	`****------------------------------****`
	`****        JOB STOPPED          ****`
	`****------------------------------****`

Coefficient storage file not found	`program terminated because coefficient`
	`storage file not found`

Unable to determine file prefix for AVS output files	`FILE ERROR: nmfil2 file:` *`filename`* `unable`
	`to determine contour file prefix`

Input deck errors	

Coordinate data not found	`****    COOR Required Input    ****`
	`****------------------------------****`
	`****        JOB STOPPED          ****`
	`****------------------------------****`

Invalid macro read	`**** error in input deck :` *`char`* `****`

Invalid AVS keyword read for macro **cont**	`ERROR:READ_AVS_IO`
	`unexpected character string (terminate`
	`program execution)`
	`Valid options are shown:`
	.
	.
	.
	`The invalid string was:`
	`string`

Invalid parameter values (macros using loop construct)	`Fatal error - for array number` *`arraynum`*
	`macro -` *`macro`*
	`Group number -` *`groupnum`*
	`Something other than a real or integer`
	`has been specified`
	-or-
	`Line number -` *`line`*
	`Bad input, check this line`

Invalid transport conditions	`Fatal error You specified a Henrys Law`
	`species with initial concentrations`
	`input for the vapor phase (icns = -2),`
	`yet the Henrys Constant is computed as 0`
	`for species number` *`speciesnum`* `and node`
	`number` *`nodenum`*`. If you want to simulate`
	`a vapor-borne species with no interphase`
	`transport, then you must specify a`
	`gaseous species (icns = -1).`

Table III. Error Conditions Which Result in Program Termination

Error Condition	Error Message
Invalid parameters set	
Noncondensible gas	cannot input ngas temp in single phase -or- ngas pressure lt 0 at temp and total press given max allowable temperature *temp* -or- ngas pressure gt total pressure i= *i* -or- ngas pressure lt 0.
Dual porosity	**** check fracture volumes,stopping**** **** check equivalent continuum VGs ****
Insufficient storage for geometric coefficients	program terminated because of insufficient storage
Unable to compute local coordinates	iteration in zone did not converge, izone = *zone* please check icnl in macro CTRL
Singular matrix in LU decomposition	singular matrix in ludcmp
Solution failed to converge	timestep less than daymin *timestep_number current_timestep_size current_simulation_time*

5.0 DATA FILES

5.1 Control file (iocntl)

5.1.1 Content

The control file contains the names of the input and output files needed by the FEHMN code. In addition to listing the I/O file names, terminal (tty) output option and the user subroutine number are given. The control file provides the user an alternate means for inputting file names, terminal output option, and user subroutine number than through the terminal I/O. It is useful when long file names are used or when files are buried in several subdirectories, or for automated program execution.

5.1.2 Use by Program

The control file is an input file which provides the FEHMN application with the names of the input and output files, terminal output units, and user subroutine number to be utilized for a particular run. The default control file name is *fehmn.files*. If the control file is found, it is read prior to problem initialization. If not present, terminal I/O is initiated and the user is prompted for required information. A control file may use a name other than the default. This alternate control file name would be input during terminal I/O. See Section 6.1.1.1.

5.1.3 Auxiliary Processing

N/A

5.2 Input file (inpt)

5.2.1 Content

The input file contains user parameter initialization values and problem control information. The form of the file name is *filen* or *filen.** where "*filen*" is a prefix used by the code to name auxiliary files and ".*" represents an arbitrary file extension. If a file name is not specified when requested during terminal I/O, the file *fehmn.dat* is the default. The organization of the file is described in detail in Section 6.2.

5.2.2 Use by Program

The input file is an input file which provides the FEHMN application with user parameter initialization values and problem control information.The input file is read during problem initialization.

5.2.3 Auxiliary Processing

N/A

5.3 Geometry data file (incoor)

5.3.1 Content

The geometry data file contains the mesh element and coordinate data. This can either be the same as the input file or a separate file.

5.3.2 Use by Program

The geometry data file is an input file which provides the FEHMN application with element and coordinate data. The geometry data file is read during problem initialization.

5.3.3 Auxiliary Processing

N/A

5.4 Zone data file (inzone)

5.4.1 Content

The zone data file contains the zone information (see macro **zone**). This can either be the same as the input file or a separate file.

5.4.2 Use by Program

The zone data file is an input file which provides the FEHMN application with geometric zone descriptions. The zone data file is read during problem initialization.

5.4.3 Auxiliary Processing

N/A

5.5 Output file (iout)

5.5.1 Content

The output file contains the FEHMN output. The file name is provided in the input control file or as terminal input, or may be generated by the code from the name of the input file if terminal I/O is envoked. The generated name is of the form *filen.out* where the "*filen*" prefix is common to the input file.

5.5.2 Use by Program

The ouput file is an output file the FEHMN application uses for general program time step summary information. It is accessed throughout the program as the simulation steps through time.

5.5.3 Auxiliary Processing

N/A

5.6 Read file (iread)

5.6.1 Content

The read file contains the initial values of pressure, temperature, saturation, and simulation time. (the restart or initial state values). The naming convention is similar to that for the output file. The generated name is of the form *filen.ini.*

5.6.2 Use by Program

The read file is an input file the FEHMN application uses for program restarts. The read file is read during problem initialization.

5.6.3 Auxiliary Processing

N/A

5.7 Write file (isave)

5.7.1 Content

The write file contains the final values of pressure, temperature, saturation, and simulation time for the run. This file can in turn be used as the read file in a restart run. The naming convention is similar to that for the output file. The generated name is of the form *filen.fin*.

5.7.2 Use by Program

The write file is an output file the FEHMN application uses for storing state data of the simulation. It is accessed at specified times throughout the program when state data should be stored.

5.7.3 Auxiliary Processing

N/A

5.8 History plot file (ishis)

5.8.1 Content

The history plot file contains data for time history plots of variables. The naming convention is similar to that for the output file. The generated name is of the form *filen.his*.

5.8.2 Use by Program

The history plot file is an output file the FEHMN application uses for storing time history data for pressure, temperature, flow, and energy output. It is accessed throughout the program as the simulation steps through time.

5.8.3 Auxiliary Processing

This file is used to produce time history plots with the Browser (see Section 8.5).

?This file is used by the code FEHPLTR (see Section C of Zyvoloski).

5.9 Solute plot file (istrc)

5.9.1 Content

The solute plot file contains time history data for solute concentrations at specified nodes. The naming convention is similar to that for the output file. The generated name is of the form *filen.trc*.

5.9.2 Use by Program

The solute plot file is an output file the FEHMN application uses for storing time history data for tracer output. It is accessed throughout the program as the simulation steps through time.

5.9.3 Auxiliary Processing

This file is used to produce time history plots of tracers with the Browser (see Section 8.5).

?It is used in the code FEHPLTR.

5.10 Contour plot file (iscon)

5.10.1 Content

The contour plot file contains the contour plot data. The naming convention is similar to that for the output file. The generated name is of the form *filen.con.*

5.10.2 Use by Program

The contour plot file is an output file the FEHMN application uses for storing contour data for pressure, temperature, flow, energy output, and tracer output. It is accessed at specified times throughout the program when contour data should be stored.

5.10.3 Auxiliary Processing

?This file is used by the code FECPLTR.

5.11 Contour plot file for dual or dpdp (iscon1)

5.11.1 Content

The dual or dpdp contour plot file contains the contour plot data for dual porosity or dual porosity / dual permeability problems. The naming convention is similar to that for the output file. The generated name is of the form *filen.dp.*

5.11.2 Use by Program

The dual or dpdp contour plot file is an output file the FEHMN application uses for storing contour data for pressure, temperature, flow, energy output, and tracer output for dual porosity or dual porosity / dual permeability problems. It is accessed at specified times throughout the program when contour data should be stored.

5.11.3 Auxiliary Processing

?This file is used by the code FECPLTR.

5.12 Stiffness matrix data file (isstor)

5.12.1 Content

The stiffness matrix data file contains finite element coefficients calculated by the code. It is useful for repeated calculations that use the same mesh, especially for large problems. The naming convention is similar to that for the output file. The generated name is of the form *filen.stor.*

5.12.2 Use by Program

The stiffness matrix data file is both an input and an output file the FEHMN application uses for storing or reading finite element coefficients calculated by the code. The stiffness matrix data file is read during problem initialization if being used for input. It is accessed after finite element coefficients are calculated if being used for output.

5.12.3 Auxiliary Processing

N/A

5.13 Input check file (ischk)

5.13.1 Content

The input check file contains a summary of coordinate and variable information, suggestions for reducing storage, coordinates where maximum and minimum values occur, and information about input for variables set at each node. The naming convention is similar to that for the output file. The generated name is of the form *filen.chk*.

5.13.2 Use by Program

The input check file is an output file the FEHMN application uses for writing a summary of the data initialization. The input check file is accessed after data initialization has been completed.

5.13.3 Auxiliary Processing

N/A

5.14 Output error file (ierr)

5.14.1 Content

The output error file contains any error or warning messages issued by the code during a run. The file is always named *fehmn.err* and will be found in the directory from which the problem was executed.

5.14.2 Use by Program

The output error file is an output file the FEHMN application uses for writing error or warning messages issued by the code during a run. It may be accessed at any time.

5.14.3 Auxiliary Processing

N/A

5.15 Advanced Visual Systems (AVS) output files

5.15.1 Content

The AVS output files contain geometry based data that can be imported into Advanced Visual Systems (AVS) UCD (unstructured cell data) graphics routines. The AVS output files each have a unique file name indicating the section type, the data type and the time step the files were created. These file names are automatically generated by the code and are of the form *fileprefix.NumberAVS_id*, where *fileprefix* is common to the contour output file prefix if defined, otherwise it is the input file prefix, *Number* is a value between 10001 and 99999, and *AVS_id* is a string denoting file content (see Table IV). In general, _head are header files, _geo is the geometry file, and _node with _mat, _sca, _vec, _con, _mat_dual, _sca_dual, _vec_dual, or _con_dual the data selected for output. Currently all properties are node based rather than cell based.

Table IV. AVS File Content Tag

AVS_id	File purpose
_avs_log	Log file from AVS output routines
_geo	Geometry output file containing coordinates and cell information
_mat_head	AVS UCD header for material properties file.
_mat_dual_head	AVS UCD header for material properties file for dual or dpdp.
_sca_head	AVS UCD header for scalar parameter values file.
_sca_dual_head	AVS UCD header for scalar parameter values file for dual or dpdp.
_vec_head	AVS UCD header for vector parameter values.
_vec_dual_head	AVS UCD header for vector parameter values for dual or dpdp.
_con_head	AVS UCD header for solute concentration file.
_con_dual_head	AVS UCD header for solute concentration file for dual or dpdp.
_mat_node	Data output file with Material properties.
_mat_dual_node	Data output file with Material properties for dual or dpdp.
_sca_node	Data output file with Scalar parameter values (pressure, temperature, saturation).
_sca_dual_node	Data output file with Scalar parameter values (pressure, temperature, saturation) for dual or dpdp.
_vec_node	Data output file with Vector parameter values (velocity).
_vec_dual_node	Data output file with Vector parameter values (velocity) for dual or dpdp.
_con_node	Data output file with Solute concentration.
_con_dual_node	Data output file with Solute concentration for dual or dpdp.

5.15.2 Use by Program

The AVS output files are output files the FEHMN application uses for storing geometry based data for material properties, temperature, saturation, pressure, velocities, and solute concentrations in a format readable by AVS graphics. The log output file is created on the first call to the AVS write routines. It includes the code version number, date and problem title. When output for a specified time step has been completed, a line containing the file name prefix,time step,call number (the initial call is 1 and is incremented with each call to write AVS contour data) and time (days) is written. The header files, one for each type of data being stored , and the single geometry file are written during the first call to the AVS output routines. The node data files are written for each call to the AVS write routines, at specified times throughout the program when contour data should be stored using AVS format.

5.15.3 Auxiliary Processing

These files are used for visualization and analysis of data by AVS and to produce contour plots by the Browser (see Section 8.5).

To use with AVS, the appropriate header file, geometry file, and data file for each node must be concatenated into one file of the form *filen.inp* (Fig. 1). This can be done with the script **fehm2avs** for a series of files

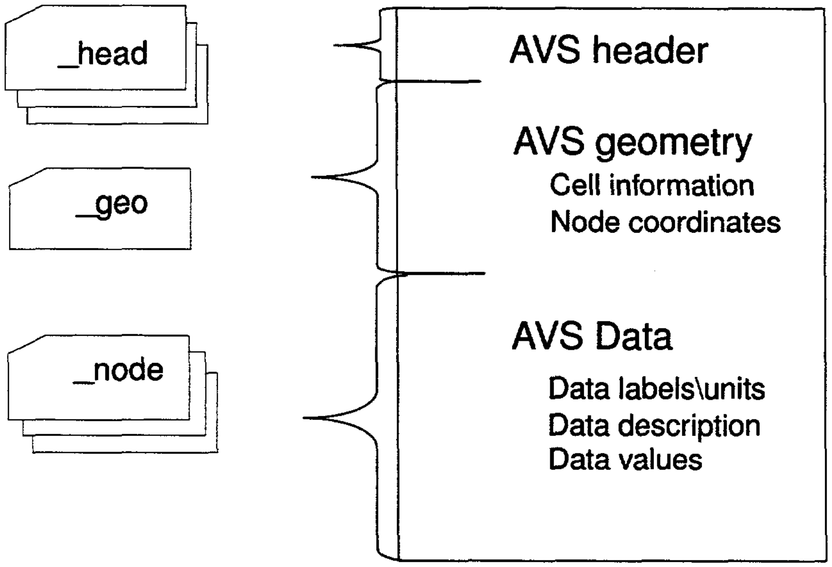


Figure 1. AVS UCD formatted FEHMN output files.

with the same root *filen* or manually, for example:

```
cat filen.10001_head filen.10001_geo filen.10001_mat_node > filen.10001.inp
```

Once header and geometry have been merged with data files into a single AVS file, the data can be imported into AVS using the read_ucd module.

6.0 INPUT DATA

6.1 General Considerations

6.1.1 Techniques

6.1.1.1 Control File or Terminal I/O Startup

The input/output (I/O) file information is provided to the code from an input control file or the terminal. The default control file name is *fehmn.files*. If a control file with the default name is present in the directory from which the code is being executed, no terminal input is required. If the default control file is not present, input prompts are written to the screen preceeded by a short description of the I/O files used by FEHMN. The descriptions of the I/O files are elaborated on in Section 5.0. The initial prompt asks for the name of a control file. If a control file name is entered for that prompt no further terminal input is required. If a control file is not used, the user is then prompted for I/O file names, the tty output flag, and user subroutine number. When the input file name is entered from the terminal the user has the option of letting the code generate the names for the remainder of the auxiliary files using the input file name prefix. The form of the input file name is *filen* or *filen.** where "*filen*" is the prefix used by the code to name the auxiliary files and ".*" represents an arbitrary file extension.

6.1.1.2 Macro Control Structure

The finite element heat and mass transfer code (FEHMN) contains a macro control structure for data input that offers added flexibility to the input process. The macro command structure makes use of a set of control statements recognized by the input module of the program. When a macro control statement is encountered in an input file, a certain set of data with a prescribed format is expected and read from the input file. In this way, the input is divided into separate, unordered blocks of data. The input file is therefore a collection of macro control statements, each followed by its associated data block. Blocks of data can be entered in any order, and any blocks unnecessary to a particular problem need not be entered. The macro control statements must appear in the first four columns of a line. The other entries are free format, which adds flexibility, but requires that values be entered for all input variables (no assumed null values).

As an aid to the user, the capabilities of FEHMN summarized in Table I refer to applicable macro commands. Table V lists the macro control statements with a brief description of the data associated with each. A more detailed description of each macro control statement and its associated input are found in Section 6.2. Macro control statements may be called more than once, if, for example, the user wishes to reset some property values after defining alternate zones. Some statements are required, as indicated in Table V, the others are optional.

Table V. Macro Control Statements for FEHMN

Control Statement	Description
adif	Air-water vapor diffusion
airwater	Isothermal air-water input
alti	Alternate input
cap	Capillary pressure data
coor	Node coordinate data **(required)**
cond	Thermal conductivity data **(required)**
cont	Contour plot data
ctrl	Program control parameters **(required)**
dof	[Not implemented]
dpdp	Double porosity/double permeability model input
dual	Input for dual porosity solution
elem	Element node data **(required)**
eos	Equation of state data
finv	Finite volume flow coefficients
flow	Flow data
flo2	Alternate format for flow data
flxo	Flux printout
hflx	Heat flux input
ice	Ice phase calculations (untested)
init	Initial value data **(required if macro pres or restart file is not used)**
iter	Iteration parameters
itup	Iterations used with upwinding
iupk	Upwind transmissibility including intrinsic permeability
ivfc	Enable volume factor calculations
ngas	Noncondensible gas (air) data
nod2	Node numbers for output and time histories, and alternate nodes for terminal output
node	Node numbers for output and time histories
num	[Not implemented]
perm	Permeability input **(required)**

Table V. Macro Control Statements for FEHMN (Continued)

Control Statement	Description
ppor	Pressure and temperature dependent porosity and permeability
pres	Initial pressure, temperature, and saturation data, boundary conditions specification **(required if macro init or restart file is not used)**
ptrk	Particle tracking simulation input
renm	Renumbers nodes
rflx	Radiation flux input
rlp	Relative permeability input **(required for 2-phase problem, otherwise optional)**
rock	Rock density, specific heat, and porosity input **(required)**
rxn	Chemical reaction rate model input
sol	Solver specifications **(required)**
solv	[Not implemented]
stea	Steady state solution generated for initial variable field
stop	Signals the end of input **(required)**
strs	Initiate stress solution, Not implemented in this version of FEHMN
text	Text input to be written to output file
thic	Variable thickness input for two-dimensional problems
time	Time step and time of simulation data **(required)**
trac	Solute simulation input
user	User subroutine call
vcon	Variable thermal conductivity input
velo	Velocity printout
wlbr	Wellbore style input
zone	Geometric definition of grid for input parameter assignment

Comments may be entered in the input file by beginning a line with a '#' symbol (the '#' symbol must be found in the first column of the line). Comments may precede or follow macro blocks but may not be found within a block.

Many input parameters such as porosity or permeability vary throughout the grid and need to have different values assigned at different nodes. This is accomplished in two ways. The first uses a nodal loop-type definition (which is the default):

JA, **JB**, **JC**, PROP1, PROP2 ...

where

JA - first node to be assigned with the properties PROP1, PROP2 ...

JB - last node to be assigned with the properties PROP1, PROP2 ...

JC - loop increment for assigning properties PROP1, PROP2 ...

PROP1, PROP2 - property values to be assigned to the indicated nodes.

In the input blocks using this structure, one or more properties are manually entered in the above structure. When a blank line is entered, that input block is terminated and the code proceeds to the next group or control statement. (Note that blank input lines are shaded in the examples shown in Section 6.2.) The nodal definition above is useful in simple geometries where the node numbers are easily found. Boundary nodes often come at regular node intervals and the increment counter **JC** can be adjusted so the boundary conditions are easily entered. To set the same property values at every node, the user may set **JA** and **JC** to 1 and **JB** to the total number of nodes, or alternatively set **JA** = 1, and **JB** = **JC** = 0.

For dual porosity problems, which have three sets of parameter values at any nodal position, nodes 1 to N [where N is the total number of nodes in the grid (see macro **coor**)] represent the fracture nodes, nodes N + 1 to 2N are generated for the second set of nodes, the first matrix material, and nodes 2N + 1 to 3N for the third set of nodes, the second matrix material. For double porosity/double permeability problems, which have two sets of parameter values at any nodal position, nodes 1 to N represent the fracture nodes and nodes N + 1 to 2N are generated for the matrix material.

For more complicated geometries, such as 3-D grids, the node numbers are often difficult to determine. Here a geometric description is preferred. To enable the geometric description the **zone** control statement (page 73) is used in the input file before the other property macro statements occur. The input macro **zone** requires the specification of the coordinates of 4-node parallelograms for 2-D problems or 8-node polyhedrons in 3-D. In one usage of the control statement **zone** all the nodes are placed in geometric zones and assigned an identifying number. This number is then addressed in the property input macro commands by specifying a **JA** < 0 in the definition of the loop parameters given above. For example if **JA** = -1, the properties defined on the input line would be assigned to the nodes defined as belonging to geometric Zone 1 (**JB** and **JC** must be input but are ignored in this case). The control statement **zone** may be called multiple times to redefine geometric groupings for subsequent input. The previous zone definitions are not retained between calls. Up to 100 zones may be defined. For dual porosity problems, which have three sets of parameter values at any nodal position, Zone 100+I is the zone number for the second set of nodes defined by Zone I, and Zone 200+I is the zone number for the third set of nodes defined by Zone I. For double porosity/double permeability problems, which have two sets of parameter values at any nodal position, Zone 100+I is the zone number for the second set of nodes defined by Zone I.

6.1.2 Consecutive Cases

The program retains no input data between cases. The values of all variables are reinitialized with each run, either from the input files or a restart file when used.

6.1.3 Defaults

Default values are set during the initialization process if overriding input is not provided by the user.

6.2 Individual Input Records or Parameters

Other than the control file or terminal I/O, the main user input is provided using macro control statements in the input file, geometry data file, or zone data file. Data provided in the input files is entered in free format with the exception of the macro control statements which must appear in the first four columns of a line. Data values may be separated with spaces, commas, or tabs. The primary input file differs from the others in that it begins with a title line (80 characters maximum) followed by input in the form of the macro commands. Each file containing macro commands should be terminated with the **stop** control statement. In the examples provided in the following subsections, blank input lines are depicted with shading.

6.2.1 Control File or Terminal I/O Input

The parameters enumerated below, are entered in order one per line in the control file (excluding the control file name [nmfile(1)]) or as prompted for during terminal input. If there is a control file with the name fehmn.files in your local space, FEHMN will execute using that control file and there will be no prompts. If another name is used for the control file, it can be entered at the first prompt.

A blank line can be entered in the control file for any auxiliary files not required, for the "none" option for tty output, and for the "0" option for the user subroutine number. The code will always write an input check file and a restart file, so if names are not provided by the user the defaults will be used. If an ouput file name is not specified, the generalized output is written to the terminal.

Input Variable	Format	Opt/ Req	Default	Description
nmfil(1)	character*100	Opt	fehmn.files	Control file name (this line is not included in the control file)
nmfil(2)	character*100	Req	fehmn.dat	Main input file name
nmfil(3)	character*100	Opt	not used	Geometry data input file name
nmfil(4)	character*100	Opt	not used	Zone data input file name
nmfil(5)	character*100	Opt	terminal	Main output file name
nmfil(6)	character*100	Opt	not used	Restart input file name
nmfil(7)	character*100	Opt	fehmn.fin	Restart output file name
nmfil(8)	character*100	Opt	not used	Simulation history output file name
nmfil(9)	character*100	Opt	not used	Solute history output file name
nmfil(10)	character*100	Opt	not used	Contour plot output file name. (Required if using avs option in cont macro.)
nmfil(11)	character*100	Opt	not used	Dual porosity or double porosity / double permeability contour plot output file name
nmfil(12)	character*100	Opt	not used	Coefficient storage output file name
nmfil(13)	character*100	Opt	fehmn.chk	Input check output file name
tty_flag	character*4	Opt	none	Terminal output flag: all, some, none
usub_num	integer	Opt	0	User subroutine call number

The following are examples of input control files.

```
tape5.dat
tape5.dat
tape5.dat
tape5.out


tape5.his
tape5.trc
tape 5.con


tape5.chk
some
0
```

```
/groupdir/c14-3
/groupdir/grid-402
/groupdir/c14-3
c14-3.out
/groupdir/c14-3.ini
c14-3.fin
c14-3.his
c14-3.trc
c14-3.con
c14-3.dp
c14-3.stor
c14-3.chk
none
0
```

6.2.2 **Control Statement adif** (optional)

Air-water vapor diffusion.

Group 1- TORT

Input Variable	Format	Description
TORT	real	Tortuosity for air-water vapor diffusion.

6.2.3 **Control Statement airwater** (optional)

Isothermal air-water two-phase simulation.

Several macros are affected if the air module is enabled. These are

pres - Because the air-water formulation is 2-phase at all times,care should be taken to insure that IEOSD is always specified to be 2. Likewise, saturations (not temperatures) are used.

init - This macro should not be used because the saturation values cannot be specified.

flow - A variety of different flow and boundary values are input with this macro when the macro **airwater** is also used. See description of control statement **flow**.

Group 1 - ICO2D

Group 2 - TREF, PREF

Input Variable	Format	Description
ICO2D	integer	Determines the type of air module used. ICO2D = 1, 1 degree of freedom solution to the saturated-unsaturated problem is produced. This formulation is similar to the RIchard's Equation. ICO2D = 2, 1 degree of freedom solution is obtained assuming only gas flow with no liquid present. ICO2D = 3, full 2 degree of freedom solution. All other values are ignored. The default is [3].
TREF	real	Reference temperature for properties (°C).
PREF	real	Reference pressure for properties (MPa).

The following is an example of **airwater**

```
airwater
   3
  20.           0.1
```

6.2.4 **Control Statement alti** (optional)

Alternate element and coordinate input. Not supported in this version.

Group 1 - CC, N

Group 2 - INFL

Input Variable	Format	Description
CC	character*4	Input file type (ment - mentat mesh generator, ptrn - patran mesh generator).
N	integer	Number of nodes in the grid.
INFL	character*100	Name of alternate element and coordinate data input file.

6.2.5 Control statement cap (no longer used, see macro **rlp**)

6.2.6 Control statement cond (required)

Assign thermal conductivities of the rock.

Group 1 - JA, JB, JC, THXD, THYD, THZD (JA, JB, JC - defined on page 22)

Input Variable	Format	Default	Description
THXD	real	1.e-30	Thermal conductivity in the x-direction ($\frac{W}{m \cdot K}$).
THYD	real	1.e-30	Thermal conductivityin the y-direction ($\frac{W}{m \cdot K}$).
THZD	real	1.e-30	Thermal conductivity in the z-direction($\frac{W}{m \cdot K}$).

The following is an example of **cond**

```
cond
   1      140       1        1.00e-00      1.00e-00      0.00e-00
```

6.2.7 Control statement cont (optional)

Contour data output format, output timestep intervals, and time intervals.

Group 1 - NCNTR, CONTIM

An alternative form of input for macro **cont** is possible. This is

Group 1 - ALTC, NCNTR, CONTIM

Group 2 - CHDUM (only input if ALTC is 'avs')

FEHMN will automatically distinguish between the alternative input formats. When keywords are used they must be entered starting in the first column. The contour data will be output whenever either of the interval criteria are satisfied.

For AVS output, if the *material* keyword is selected, the following material property values will be written for each node: permeability in the x, y, and z directions, thermal conductivity in the x, y, and z directions, porosity,

Input Variable	Format	Description
ALTC	character*4	Keyword specifying the type of contour output wanted (avs, fehm, free, ment, ptrn): 'avs' produces contour plot files compatible with the AVS postprocessor. 'fehm' produces a binary output file. The same contour plot file is produced using the first form of Group1 input. 'free' produces a free format contour plot file. 'ment' produces a contour plot file compatible with the MENTAT postprocessor. 'ptrn' produces a contour plot file compatible with the PATRAN postprocessor.
NCNTR	integer	Time step interval for contour plots (number of timesteps). Output contour information each NCNTR timesteps.
CONTIM	real	Time interval for contour plots (days). In addition to output each NCNTR timesteps, output contour information each CONTIM days.
CHDUM	character*72	Keyword specifying type of AVS contour plot data files to be created in AVS UCD format, either formatted (ASCII) or unformatted (binary). Keywords are entered one per line and terminated with 'endavs'. Valid keywords (case insensitive) are: (m)aterial - output contour values for material properites. (l)iquid - output contour values for liquid phase. (va)por - output contour values for vapor phase. (ve)locity - output velocity values. (dp)dp - output contour values for dual permeability nodes. (p)ressure - output pressure values. (t)emperature - output temperature values. (s)aturation - output saturation values. (c)oncentration - output solute concentration values. (f)ormatted - output data in ASCII format. (u)nformatted - output data in binary format. (e)ndavs - last keyword entered. If a format keyword is not entered, the default is 'formatted'. The default for data keywords is "off". The letters given in ( ) are sufficient to identfy the keyword.

rock specific heat, capillary pressure, relative permeability model being used, and capillary pressure model being used. If *vapor* and/or *liquid* are selected, *pressure* or *velocity* must also be defined (otherwise, no data for these values will be written). *velocity* will result in vector values, *pressure* values will be scalar. If *concentration* is selected, values will be output only if nspeci is defined for tracer solutions. See the control statement **trac** for a description of nspeci for solutes.

The following are examples of **cont**. For the first example, FEHMN binary format contour output files will be written every 100 timesteps and for each 1.e20 days. The second example invokes AVS contour output. AVS UCD binary files will be written for every 100 time steps and 1.e20 days. The resulting files will include a log file, geometry file, plus header and data

```
cont
 100          1.e20
```

```
cont
avs             100           1.e20
liquid
velocity
con
pressure
temp
mat
unformatted
end
```

files for the following: material properties, solute concentrations, liquid velocities, pressures and temperatures.

6.2.8 Control statement **coor** (required)

Node coordinate data. These data are usually created by a mesh generation program, then cut and copied into the input file or a separate geometry data input file. The mesh must be a right handed coordinate system.

Group 1 - N

Group 2 - MB, CORD1, CORD2, CORD3

To end the control section a blank line is entered.

Input Variable	Format	Description
N	integer	Number of nodes in the grid
MB	integer	Node number. If MB < 0 then the difference between the absolute value of MB and the previously read absolute value of MB is used to generate intermediate values by interpolation.
CORD1	real	X-coordinate of node MB (m).
CORD2	real	Y-coordinate of node MB (m).
CORD3	real	Z-coordinate of node MB (m).

The following is an example of **coor**

```
coor
 140
   1      0.00000    200.00000      0.00000
   2     12.50000    200.00000      0.00000
   .         .           .             .
   .         .           .             .
   .         .           .             .
  10    212.50000    200.00000      0.00000
   .         .           .             .
   .         .           .             .
   .         .           .             .
 140    300.00000      0.00000      0.00000
```

6.2.9 Control statement ctrl (required)

Assign various control parameters needed for equation solvers and matrix solver routines.

Group 1 - MAXIT, EPM, NORTH

Group 2 - JA, JB, JC, IGAUS (JA, JB, JC - defined on page 22)

Group 3 - AS, GRAV, UPWGT

Group 4 - IAMM, AIAA, DAYMIN, DAYMAX

Group 5 - ICNL, LDA

Input Variable	Format	Default	Description
MAXIT	integer		Maximum number of iterations allowed in either the overall Newton cycle or the inner cycle to solve for the corrections at each iteration. If MAXIT < 0 then the maximum number of iterations is ABS(MAXIT) but the minimum number of iterations is set to 2. {10}
EPM	real		Tolerance for Newton cycle (nonlinear equation tolerance). {1.e-5}
NORTH	integer		Number of orthogonalizations in the linear equation solver. {8}
IGAUS	integer	1	The order of partial Gauss elimination {1 or 2 is recommended}. Larger values increase memory utilization but may be necessary for convergence.
AS	real		Implicitness factor. {1} AS $\leq$ 1, use standard pure implicit formulation. AS > 1, use second-order implicit method.

GRAV	integer		Direction of gravity
			GRAV = 0, no gravity is used.
			GRAV = 1, X-direction.
			GRAV = 2, Y-direction.
			GRAV = 3, Z-direction.
			A value for gravity of 9.81 m/s^2 is used in the code when GRAV $\neq$ 0. If GRAV > 3, GRAV is set equal to 3.

UPWGT	real		Value of upstream weighting {0.5 $\leq$ UPWGT $\leq$ 1.0}. If UPWGT < 0.5, UPWGT is set to 0.5 If UPWGT > 1.0, UPWGT is set to 1.0
IAMM	integer		Maximum number of iterations for which the code will multiply the time step size. If this number of time steps is exceeded at any time, the time step will not be increased for the next time {7-10}.
AIAA	real	1	Time step multiplier {1.2-2.0}

Input Variable	Format	Default	Description
DAYMIN	real		Minimum time step size (days)
DAYMAX	real		Maximum time step size (days)

ICNL	integer		Parameter that specifies the geometry
			ICNL = 0, three-dimensional
			ICNL = 1, X - Y plane
			ICNL = 2, X - Z plane
			ICNL = 3, Y - Z plane
			ICNL = 4, X - Y radial plane, (radius is X)
			ICNL = 5, X - Z radial plane, (radius is X)
			ICNL = 6, Y - Z radial plane, (radius is Y)

LDA	integer	0	Parameter that specifies the external storage of geometric coefficients LDA = +1, element coefficients are read from file *filen.stor* and no coefficients are calculated in the code LDA = 0, element coefficients are calculated in the code and not saved LDA = -1, element coefficients are calculated in the code and saved on file *filen.stor* It should be noted that if the coefficients are read from a file (LDA = 1) then the macro **finv** is ignored as well as information read from macros **elem** and **coor**.

The following is an example of **ctrl**

```
ctrl
  40        1.e-7      8
   1      140          1        1

   1.0      0.0        1.0
  40        1.2        0.1      60.0
   1        0
```

6.2.10 Control statement dof (Not implemented)

6.2.11 Control statement dpdp (optional)

Double porosity / double permeability formulation. There are two sets of parameter values at any nodal position, for which property values must be defined. Nodes 1 to N (see macro **coor** for definition of N) represent the fracture nodes and nodes N + 1 to 2N the matrix material. When zones are used with the **dpdp** macro, additional zones are automatically generated. See instructions for the macro **zone** for a more detailed description.The **dpdp** parameters are only defined for the first N nodes.

Group 1 - IDPDP

Group 2 - JA, JB, JC, VOLFD1 (JA, JB, JC - defined on page 22)

Group 3 - JA, JB, JC, APUV1 (JA, JB, JC - defined on page 22)

Input Variable	Format	Default	Description
IDPDP	integer		Solution descriptor for double porosity/double permeability solution. IDPDP = 0, information is read but not used. IDPDP ≠ 0, **dpdp** solution is implemented.
VOLFD1	real	1.	Volume fraction for fracture node.
APUV1	real	10.	Length scale for matrix nodes (m).

The volume fraction VOLFD1 is related to the total volume by

$$VOLFD1 + VOLFD2 = 1.0$$

where VOLFD2 is the volume fraction of the matrix node. If permeability model IRLP = 4 is selected in control statement **rlp**, VOLFD1 is calculated from RP15 (fracture porosity) in that control statement.

The following is an example of **dpdp**

```
dpdp
  1
  1        140         1          0.005

  1        140         1          0.10
```

6.2.12 Control statement dual (optional)

Dual porosity formulation. There are three sets of parameter values at any nodal position, for which property values must be defined. Nodes 1 to N (see macro **coor** for definition of N) represent the fracture nodes, nodes N + 1 to 2N the first matrix material, and nodes 2N + 1 to 3N the second matrix material. When zones are used with the **dual** macro, additional zones are automatically generated. See instructions for the macro **zone** for a more detailed description. The **dual** parameters are only defined for the first N nodes.

Group 1 - IDUALP

Group 2 - JA, JB, JC, VOLFD1 (JA, JB, JC - defined on page 22)

Group 3 - JA, JB, JC, VOLFD2 (JA, JB, JC - defined on page 22)

Group 4 - JA, JB, JC, APUVD (JA, JB, JC - defined on page 22)

Input Variable	Format	Default	Description
IDUALP	integer		Solution descriptor for dual porosity solution. IDUALP = 0, information is read but not used IDUALP ≠ 0, dual porosity solution is implemented
VOLFD1	real	0.001	Volume fraction for fracture portion of the continuum.

Input Variable	Format	Default	Description
VOLFD2	real	0.5	Volume fraction for the first matrix portion of the continuum.
APUVD	real	5.	Length scale for the matrix nodes (m).

The volume fractions VOLFD1 and VOLFD2 are related to the total volume by

$$VOLFD1 + VOLFD2 + VOLFD3 = 1.0$$

where VOLFD3 is the volume fraction of the second matrix node. If permeability model IRLP = 4 is selected in control statement **rlp**, VOLFD1 is calculated from RP15 (fracture porosity) in that control statement.

The following is an example of **dual**

```
dual
   1
   1          140           1           0.006711409
   1          140           1           0.335570470
   1          140           1           0.10
```

6.2.13 Control statement elem (required).

Element connectivity data. These data are created by a mesh generation program, then cut and copied into the input file or a separate geometry data input file.

Group 1 - NS, NEI

Group 2 - MB, NELM (1), NELM (2), . . ., NELM (NS)

IF $NS < 0$ then ABS(NS) is interpreted as the number of nodes per element. $NS < 0$ signals the code to make rectangles (or bricks in three dimensions) a sum of triangles (or tetrahedrals).This provides more stability in nonlinear problems with a distorted mesh. Figure 2 shows available element types and the nodal numbering convention. To end the control section a blank line is entered.

Input Variable	Format	Description
NS	integer	Number of nodes per element.
NEI	integer	Number of elements
MB	integer	Element number. If MB < 0 then the difference between the absolute value of MB and the previous absolute value of MB is used to generate intermediate values by interpolation in the code.
NELM (1)	integer	First node of element MB
NELM (2)	integer	Second node of element MB

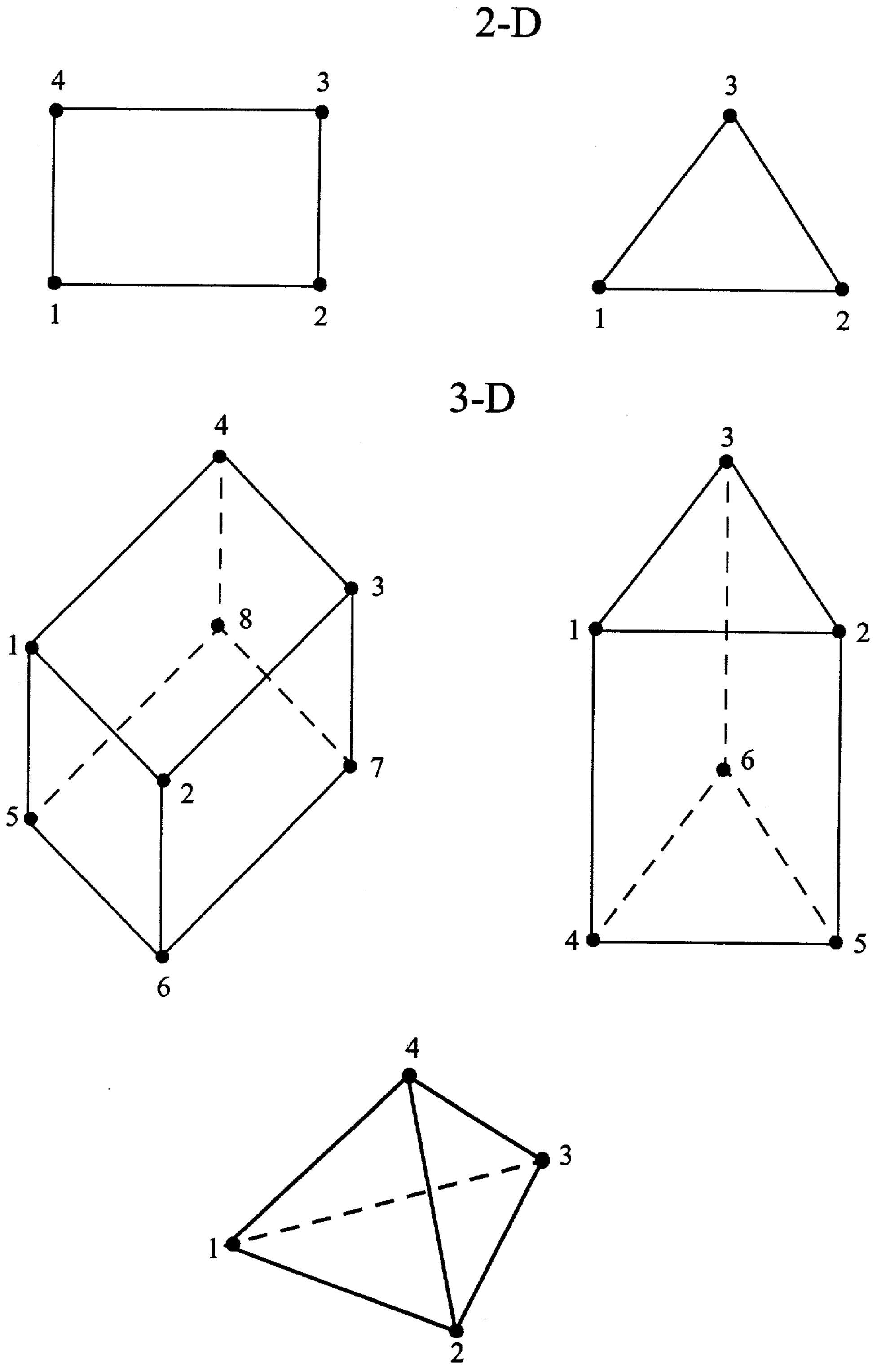


Figure 2. Elements available with FEHMN in 2-D and 3-D problems showing nodal numbering convention.

Input Variable	Format	Description
.	.	.
.	.	.
.	.	.
NELM (NS)	integer	Last node of element MB

The following is an example of **elem**

```
elem
   4      117
   1       15       16        2        1
   2       16       17        3        2
   .        .        .        .        .
   .        .        .        .        .
   .        .        .        .        .
  10       24       25       11       10
  11       25       26       12       11
  12       26       27       13       12
   .        .        .        .        .
   .        .        .        .        .
   .        .        .        .        .
 116      138      139      125      124
 117      139      140      126      125
```

6.2.14 Control Statement eos (optional)

Equation of State. Provide the code with alternate thermodynamic properties for the liquid and/or vapor phases. (This is one way in which the code may be instructed to simulate nonisothermal, single phase air. It may also be used to make comparisons between the code and analytical solutions that use different equations of state.)

Group 1 - IIEOSD, IPSAT, ITSAT

Group 2 - EWI, EW2, EW3, EW4, EW5, EW6, EW7, EW8, EW9, EW10, EW11

Group 3 - EVI, EV2, EV3, EV4, EV5, EV6, EV7, EV8, EV9, EV10, EV11

For the calculation of vapor density and its derivatives, the ideal gas law is used instead of a linear relationship. Thus, EV4 and EV5 are not used, but are included so the format is the same as that for the liquid parameters in Group 2.

Input Variable	Format	Description
IIEOSD	integer	Equation of state reference number. When IIEOSD = 1 or 2 are used, they refer to the high and low pressure data sets, respectively, in FEHMN. For these values the input in Group 2 and Group 3 will be ignored after it is entered. When any value other than 1 or 2 are used, the user-defined equation of state is used with Groups 2 and 3 for input.

Input Variable	Format	Description
IPSAT	integer	Parameter to set vapor pressure to zero. If IPSAT ≠ 0 the vapor pressure is set to zero, otherwise the vapor pressure is calculated in the code.
ITSAT	integer	Parameter to adjust the saturation temperature. If ITSAT < 0, the saturation temperature is set to -1000°C. If ITSAT > 0, the saturation temperature is set to 1000°C. If ITSAT = 0, the calculated value is used.
EW1	real	Liquid reference pressure (MPa).
EW2	real	Liquid reference temperature (°C).
EW3	real	Liquid reference density (kg/m^3).
EW4	real	Derivative of liquid density with respect to pressure at reference conditions.
EW5	real	Derivative of liquid density with respect to temperature at reference conditions.
EW6	real	Liquid reference enthalpy (MJ/kg).
EW7	real	Derivative of liquid enthalpy with respect to pressure at reference conditions.
EW8	real	Derivative of liquid enthalpy with respect to temperature at reference conditions.
EW9	real	Liquid reference viscosity.
EW10	real	Derivative of liquid viscosity with respect to pressure at reference conditions.
EW11	real	Derivative of liquid viscosity with respect to temperature at reference conditions.
EV1	real	Vapor reference pressure (MPa).
EV2	real	Vapor reference temperature (°C).
EV3	real	Vapor reference density (kg/m^3).
EV4	real	Not used, included only to maintain a similar format to Group 2.
EV5	real	Not used, included only to maintain a similar format to Group 2.
EV6	real	Vapor reference enthalpy (MJ/kg).
EV7	real	Derivative of vapor enthalpy with respect to pressure at reference conditions.
EV8	real	Derivative of vapor enthalpy with respect to temperature at reference conditions.
EV9	real	Vapor reference viscosity.

Input Variable	Format	Description
EV10	real	Derivative of vapor viscosity with respect to pressure at reference conditions.
EV11	real	Derivative of vapor viscosity with respect to temperature at reference conditions.

6.2.15 Control Statement finv (optional)

No input is associated with this macro. When invoked, the code will perform finite volume calculations instead of finite element calculations for flow terms -- this may improve accuracy on non-orthogonal grid systems. Anisotropic properties (permeability, conductivity) are not supported with this macro. In this case, the values for permeability in the x-direction from control statement **perm** are used.

6.2.16 Control statement flow (required for flow problem)

Flow data. Source and sink parameters are input and may be used to apply boundary conditions. Note that the alternative definitions (isothermal conditions) apply when control statement **airwater** is used.

Group 1 -JA, JB, JC, SKD, EFLOW, AIPED (JA, JB, JC - defined on page 22)

If the porosity of the node is zero, then there is only a temperature solution, and the code forms a source proportional to the enthalpy difference. The source term is given by $Q = AIPED \cdot (E - EFLOW)$, where E is the in-place enthalpy and EFLOW is a specified enthalpy.

The following is an example of **flow**

```
flow
  88       88       1       0.050      -25.0        0.
  14       14       1       3.600     -160.0        1.
```

Input Variable	Format	Default	Description
Non-Isothermal			
SKD	real	0.	Heat and mass source strength (kg/s), heat only (MJ/s). Negative value indicates injection into the rock mass.
EFLOW	real	0.	Enthalpy of fluid injected (MJ/kg). If the fluid is flowing from the reservoir, then the in-place enthalpy is used. If EFLOW < 0, then ABS(EFLOW) is interpreted as a temperature (°C) and the enthalpy (assuming water only) calculated accordingly. In heat only problems with EFLOW < 0, the node is in contact with a large heat pipe that supplies heat to the node through an impedance AIPED so as to maintain its temperature near ABS (EFLOW). Large values (approximately 1000) of AIPED are recommended.

Input Variable	Format	Default	Description
AIPED	real	0.	Impedance parameter. If AIPED is nonzero, the code interprets SKD as a flowing wellbore pressure (MPa) with an impedance ABS(AIPED). If AIPED < 0, flow is only allowed out of the well. For heat only, AIPED is the thermal resistance. If AIPED = 0, SKD is flow rate. If AIPED $\neq$ 0 and SKD = 0 the initial value of pressure will be used for the flowing pressure.
Isothermal air-water			
Case 1: AIPED = 0 (Constant Mass Rate, 1- or 2-Phase Source or Sink)			
SKD	real	0.	Mass source strength (kg/s). Negative value indicates injection into the rock mass.
EFLOW	real	0.	a) EFLOW $\geq$ 0, EFLOW is source liquid saturation, $Q_w = SKD \cdot EFLOW$ (kg/s), $Q_a = SKD \cdot (1 - EFLOW)$ (kg/s) b) EFLOW < 0, ABS(EFLOW) is the source air pressure (MPa) $Q_w = SKD$ (kg/s) $Q_a = 1.0 \cdot (P_a - ABS(EFLOW))$ (kg/s) In the above and following relations, Q_w is the source term for water, Q_a is the source term for air, and P_a is the in-place air pressure. The second case works well in situations where inflow is specified and it is desired to hold the air pressure at a constant value.
Case 2: AIPED > 0 (Constant Pressure, Constant Liquid Saturation Source or Sink)			
SKD	real	0.	Specified source air pressure (MPa).
EFLOW	real	0.	a) EFLOW < 0, air only source . $Q_a = AIPED \cdot (P_a - SKD)$ (kg/s) b) $0 < EFLOW \leq 1$, EFLOW is specified source liquid saturation, for SKD $\geq$ 0, 2-phase source, $Q_a = AIPED \cdot (P_a - SKD)$ (kg/s) $Q_w = AIPED \cdot (S_l - EFLOW)$ (kg/s) when SKD < 0, water only source $Q_a = 0$. In the above relation S_l is the in-place liquid saturation.
AIPED	real		Impedance parameter. A large value is recommended (10^2 - 10^6), in order to create a flow term large enough to maintain constant pressure.

Input Variable	Format	Default	Description
Case 3: AIPED < 0 (Outflow only , if $P_l > SKD$)			
SKD	real	0.	Pressure above which outflow occurs (MPa)
EFLOW	real	0.	Not used.
AIPED	real	0.	Impedance parameter. $Q_w = ABS(AIPED) \cdot R_l/\mu_l(P_l - SKD)$ (kg/s) where R_l is the water relative permeability and μ_l is the water viscosity.

6.2.17 Control statement flo2 (optional)

Group 1 - JA, JB, JC,JD, SKD, EFLOW, AIPED (SKD, EFLOW, AIPED - defined on page 36 under control statement **flow**)

Multiple lines of input may be used, terminated by a blank line.

Input Variable	Format	Description
JA	integer	Indices used to define planes in a 3-D simulation with a regular numbering pattern. The flow rates are defined within the inner loop of the do loops: DO JK = JA, JB KL = JK - JA DO IJ = JA + KL, JC + KL, KD ... ENDDO ENDDO
JB	integer	
JC	integer	
JD	integer	

6.2.18 Control statement flxo (optional)

Mass flux between two nodes is output by choosing this control statement.

Group 1 - NFLX

Group 2 - IFLX1, IFLX2 (repeated NFLX times)

Group 3 - X1, Y1, Z1 (as needed)

Group 4- X2, Y2, Z2 (as needed)

Input Variable	Format	Description
NFLX	integer	Number of internode fluxes to be calculated.
IFLX1	integer	First node to be used in flux calculation.
IFLX2	integer	Second node to be used in flux calculation.

Input Variable	Format	Description
X1	real	Coordinates of the first node to be used in flux calculation. Used only for those nodes where IFLX1 < 0.
Y1	real	
Z1	real	
X2	real	Coordinates of the second node to be used in flux calculation. Used only for those nodes where IFLX2 < 0.
Y2	real	
Z2	real	

If IFLX1 < 0, then after all IFLX1 and IFLX2 values are read, coordinates X1, Y1, and Z1 are read and the node nearest to these coordinates is used. If IFLX2 < 0, coordinates for the second node are read in on another line. The code cycles through each IFLX1 and IFLX2 in this manner, reading coordinates when needed. Results are written to the screen if tty output is enabled and to the output file **iout**.

6.2.19 Control Statement hflx (optional)

Group 1 - JA, JB, JC, QFLUX, QFLXM (JA, JB, JC - defined on page 22)

A negative heat flux indicates heat flow into the reservoir.

Input Variable	Format	Default	Description
QFLUX	real	0.	If QFLXM = 0, then QFLUX is the heat flux (MW). If QFLXM ≠ 0, then QFLUX is a temperature (°C) and the heat flux is calculated according to the formula: $Q_H = QFLXM(TL - QFLUX)$ (MW).
QFLXM	real	0.	If QFLXM ≠ 0, multiplier for heat flux equation given in QFLUX description (MW/°C). This must be large for large temperature gradients, or when a constant temperature must be maintained.

The following is an example of **hflx**

```
hflx
 401        410        1        -0.001        0.0
```

6.2.20 Control Statement ice (optional)

Ice phase calculations, not tested.

Group 1 - ICE, SIIN, TMELT

Group 2 - JA, JB, JC, SII (JA, JB, JC - defined on page 22)

Input Variable	Format	Description
ICE	integer	Solution descriptor for ice solution. ICE = 0, information is read but not used. ICE ≠ 0, **ice** solution is implemented.
SIIN	real	Default value for ice saturation (used when ice saturation SII in Group 2 is set to 0 at any node).
TMELT	real	Freezing temperature of water (°C).
SII	real	Ice saturation. The default value is [0].

6.2.21 Control Statement init (required if macro **pres** not used)

Set initial pressure and temperature at all nodes.

Group 1 - PEIN, TIN, TIN1, GRAD1, DEPTH, TIN2, GRAD2, QUAD

Note that the macro **pres** may overwrite some of the values that are set by mactro **init**.

Input Variable	Format	Description
PEIN	real	Initial value of pressure (MPa). If initial values are read from the read file (iread), then this value is ignored. If gravity is present, this is the value of the pressure at node 1, and the other nodal pressures are adjusted by applying the hydraulic head. Absolute pressures are used. Pressure as a function of depth is calculated with TIN < 0.
TIN	real	Initial value of temperature (°C). If TIN ≤ 0, then the initial temperatures are calculated using the temperature gradient formulas given below.
TIN1	real	Defined in formulas below (°C)
GRAD1	real	Defined in formulas below (°C/m)
DEPTH	real	Defined in formulas below (m)
TIN2	real	Defined in formulas below (°C)
GRAD2	real	Defined in formulas below (°C/m)
QUAD	real	Defined in formulas below (°C/m^2)

$$T = TIN1 + GRAD1 \times Z \qquad 0 \le Z \le \text{DEPTH}$$

$$T = TIN1 + GRAD2 \times Z + QUAD \times Z^2 \qquad Z > \text{DEPTH}$$

The following is an example of **init**

```
init
   3.6      0.0     240.      0.       0.      240.      0.       0.
```

6.2.22 Control statement iter (optional)

If the user is not familiar with the linear equation solver routines in FEHMN control statement **iter** should not be used.

Group 1 - G1, G2, G3, TMCH, OVERF

Group 2 - IRDOF, ISLORD, IBACK, ICOUPL, RNMAX

The following is an example of **iter**

```
iter
    1.e-5         1.e-5         1.e-5         1.e-9          1.2
    1             0             0             2            200.0
```

Input Variable	Format	Default	Description
G1	real	1.e-6	Multiplier for the linear convergence region of the Newton-Raphson iteration.
G2	real	1.e-6	Multiplier for the quadratic convergence region of the Newton-Raphson iteration.
G3	real	1.e-3	Tolerance for the adaptive implicit method (multiplying factor for Newton-Raphson tolerance).
TMCH	real	1.e-9	Machine tolerance. If satisfied by the residual norm, the Newton iteration is assumed to be complete. If TMCH is < 0 the ABS(TMCH) is used as a tolerance for each equation at each node. Convergence is achieved if the residual of every equation at every node is < ABS(TMCH).
OVERF	real	1.1	Over relaxation factor for passive nodes in adaptive implicit method.
IRDOF	integer	0	Enables the reduced degree of freedom method {0}. Set to 0 if reduced degrees of freedom are not required. When IRDOF = 1, a reduced degree of freedom from 3 to 2 or 3 to 1 is used. When IRDOF = 2, a reduced degree of freedom from 3 to 2 is used. If IRDOF=11, then an air only solution is found for the isothermal air-water process model.

<table>
<tr><th>Input Variable</th><th>Format</th><th>Default</th><th>Description</th></tr>
<tr><td>ISLORD</td><td>integer</td><td>0</td><td>Reordering parameter. The ordering can be understood by labeling the mass equation as 1, the heat equation as 2, and the noncondensible gas equation (if it exists) as 3. The value of ISLORD and the corresponding equation order is given below. The ordering has an effect on the speed of convergence of several solution algorithms, but will not affect most users.
<table>
<tr><th>ISLORD</th><th>2 Degrees of Freedom</th><th>3 Degrees of Freedom</th><th>4 Degrees of Freedom</th></tr>
<tr><td>0</td><td>1, 2</td><td>1, 2, 3</td><td>1, 2, 3, 4</td></tr>
<tr><td>1</td><td>2, 1</td><td>1, 3, 2</td><td>1, 3, 2, 4</td></tr>
<tr><td>2</td><td></td><td>2, 1, 3</td><td></td></tr>
<tr><td>3</td><td></td><td>2, 3, 1</td><td></td></tr>
<tr><td>4</td><td></td><td>3, 1, 2</td><td></td></tr>
<tr><td>5</td><td></td><td>3, 2, 1</td><td></td></tr>
</table></td></tr>
<tr><td>IBACK</td><td>integer</td><td></td><td>IRDOF parameter. {0} If IBACK = 1, SOR iterations are performed before call to solver. If IBACK = 2, SOR iterations are performed before call to SOLVER, and SOLVER is called twice.</td></tr>
<tr><td>ICOUPL</td><td>integer</td><td></td><td>Number of SOR iterations used in reduced degree of freedom methods. {0}</td></tr>
<tr><td>RNMAX</td><td>real</td><td></td><td>Maximum running time for problem before the solution is stopped (cpu minutes) {very large if not set with control statement <b>iter</b>).</td></tr>
</table>

6.2.23 Control Statement **itup** (optional)

Group 1 - IAD_UP

Input Variable	Format	Default	Description
IAD_UP	integer	100	Number of iterations after which the upwind directions are held constant. {2}

6.2.24 Control Statement **iupk** (optional)

No input is associated with this control statement. If enabled the full transmissibility term will be upwinded (including the intrinsic permeability). Otherwise the fluid and relative permeability part of the transmissibility will be upwinded and the intrinsic permeability will be harmonically averaged.

6.2.25 Control Statement **ivfc** (optional)

Enables volume control subroutine. Not supported in this version.

6.2.26 Control Statement mdnode (optional)

Enables extra connections to be made to nodes. This is useful for simulating wellbore connections, faults, and flow across internal boundaries.

Group 1 - NUM_MD, MAX_CON

Group 2 - NODE, IPAR, NPAR (repeated NUM_MD times)

Input Variable	Format	Default	Description
NUM_MD	integer	0	Number of new connections to be entered.
MAX_CON	integer	0	Maximum number of new connections to a given node. This does not include old connections. Thus, if a node was already connected to 5 neighboring nodes and two new connections were added to this node in this macro statement and this was the maximum number of connections added in this macro statement, then MAX_CON=2.
NODE	integer	0	Node to which new connection is established.
IPAR	integer	0	IPAR is not used at present. Its value is ignored. However the entered number must be an integer.
NPAR	integer	0	NPAR is the new connected node. If NPAR=NODE, no new connection is established.

The following are examples of **mdnode**

```
mdnode
   3        2
  10        0        15
 100        0       106
  10        0       320
```

```
mdnode
   4        3
   1        0        16
   2        0         1
   4        0         1
  10        0       203
```

6.2.27 Control Statement ngas (optional)

Noncondensible gas transport.

Group 1 - ICO2D

Group 2 - JA, JB, JC, PCO2 (JA, JB, JC - defined on page 22)

Group 3 - JA, JB, JC, CPNK (JA, JB, JC - defined on page 22)

Group 4 - JA, JB, JC, QCD (JA, JB, JC - defined on page 21)

Input Variable	Format	Default	Description
ICO2D	integer	3	Solution descriptor for noncondensible gas transport. ICO2D = 1, the 3 degree of freedom solution will be reduced to a 1 degree of freedom problem. (See macro **iter**, the parameter ICOUPL is also set to 5 if ICO2D = 1.) ICO2D = 2, the 3 degree of freedom solution will be reduced to a 2 degree of freedom problem. (See macro **iter**, the parameter ICOUPL is also set to 5 if ICO2D = 2.) ICO2D = 3, full 3 degree of freedom.
PCO2	real	0.	Initial partial pressure of noncondensible gas. If PCO2 < 0 then ABS (PCO2) is interpreted as a temperature and the partial pressure of the noncondensible gas is calculated according to the formula: $PCO2 = P_T - P_{SAT}(T)$ where P_T is the total pressure and $P_{SAT}(T)$ is the water saturation pressure and is a function of temperature only.
CPNK	real	0.	If CPNK < 0, then ABS (CPNK) is the specified noncondensible pressure and will be held at that value. If CPNK > 0, then CPNK is the specified relative humidity and the saturation, S_l, is calculated using the vapor pressure lowering formula and the capillary pressure formula: $Pcap(S_l) = \ln(h)\rho_l RT$ where $Pcap$ is the capillary function, h is the humidity, R is the gas constant, T is the temperature, and ρ_l is the liquid density. Once the formula is solved, S_l is held constant.
QCD	real	0.	Specified air flow rate (kg/sec).

The following is an example of **ngas**

```
ngas
   3
   1        800        1       -20

   1        800        1        0.

   1        800        1        0.
```

6.2.28 Control statement nod2 (optional)

Specify the node numbers for which detailed file output is desired and alternate nodes for terminal output.

Group 1 - M, M2

Group 2 - MN (1), MN (2), . . . , MN (M)

Group 3 -X, Y, Z (as needed)

Group 4 - MNI(1), MNI(2), . . . , MNI(M2)

Group 5 - X, Y, Z (as needed)

Input Variable	Format	Description
M	integer	Number of nodes for which information will be printed on the output file (iout). If M ≤ 0, pressure and temperature will be written on the output file for all nodes but no nodal parmeter values will be printed in the history plot files. Group 2 is omitted if M ≤ 0.
M2	integer	Number of nodes for short list (terminal printout). If M2 ≤ 0, Group 4 is omitted.
MN	integer	M node numbers for which information will be printed on the output file (iout). If a MN(I) < 0, then coordinates are used to define that print-out node, and the coordinate sets (X, Y, Z) for each MN(I) < 0 are added after Group 2.
MNI	integer	M2 node numbers for which information will be printed on the terminal (short list). This group exists only if M2 ≠ 0. If MNI(I) < 0, then coordinates are used to define the terminal output nodes, and the coordinate sets (X, Y, Z) for each MNI(I) < 0 are added after Group 4.
X	real	Coordinates of node for which information will be printed. One line for each MN or MNI < 0. The code finds the node closest to the coordinate given. For 2-D problems set Z = 0. No input if no MN or MNI < 0.
Y	real	
Z	real	

The following are examples of **nod2**

```
nod2
  2        1
 50       88
 50
```

```
nod2
  2        1
 50       88
-88
100.    1000.      0.
```

6.2.29 Control statement node (optional)

Specify the node numbers for which detailed output is desired.

Group 1 - M

Group 2 - MN (1), MN (2) ... MN (M)

Group 3 - X, Y, Z (as needed)

...OR...

Group 1 - KEYWORD

Group 2 - JA, JB, JC (JA, JB, JC - defined on page 22)

Input Variable	Format	Description
M	integer	Number of nodes for which information will be printed on the output (iout) and history plot (ishis, istrc) files. If M ≤ 0, pressure and temperature will be written on the output file for all nodes but no nodal parmeter values will be printed in the history plot files. Group 2 is omitted if M ≤ 0.
MN	integer	M node numbers for which information will be printed on the output file (iout). If MN(I) < 0, then coordinates are used to define the print-out node, and the coordinate sets (X, Y, Z) for each MN(I) < 0 are added after Group 2.
X	real	Coordinates of node for which information will be printed. One line for each MN < 0. The code finds the node closest to the coordinate given. For 2-D problems set Z = 0. No input if no MN >0.
Y	real	
Z	real	
KEYWORD	character*5	Key word for invoking node specification by ja, jb, jc format. The necessary word is *block.*

The following are examples of **node**

```
node
  2
 50         88
```

```
node
  2
 50         -88
100.        1000.        0.
```

```
node
block
  1         100         10
 -3         0           0
```

6.2.30 Control statement num (Not implemented)

6.2.31 Control statement perm (required)

Assign permeabilities of the rock. Permeabilities represent average values of a volume associated with a node. Note that using **rlp** to describe relative permeabilities causes these values to be overwritten.

Group 1 - JA, JB, JC, PNXD, PNYD, PNZD (JA, JB, JC - defined on page 22)

Input Variable	Format	Default	Description
PNXD	real	1.e-30	Permeability in the x-direction (m^2).
PNYD	real	1.e-30	Permeability in the y-direction (m^2).
PNZD	real	1.e-30	Permeability in the z-direction (m^2).

The following is an example of the **perm** macro

```
perm
   1      140         1          2.50e-14          2.50e-14          0.00e-00
```

6.2.32 Control Statement ppor (optional)

Group 1 - IPOROS, R1, R2, R3

Group 2 - JA, JB, JC, R4, R5 (JA, JB, JC - defined on page 22)

Input Variable	Format	Default	Description
IPOROS	integer		Porosity/permeability type. IPOROS = 0, constant porosity. IPOROS = 1, simple linear model: $\phi = \phi_0 + (1-\phi_0)(c_r - c_g)(P - P_0)$ IPOROS = 2, Gangi stress model: $\phi = \phi_0\left[1-\left(\frac{P_c}{P_0}\right)^m\right]$ and $P_c = \sigma - P - \alpha E \Delta T$ This option should only be used with single phase liquid water problems.
R1	real		Parameter used in the linear and Gangi models. For the Linear model, pore volume compressibility, c_r (MPa $^{-1}$). For the Gangi model, coefficient of thermal expansion, α (°C $^{-1}$).
R2	real		Parameter used in the linear and Gangi models. For the Linear model, compressibility of the matrix grain, c_g (MPa $^{-1}$). For the Gangi model, Young's modulus, E (MPa).
R3	real		Parameter used in the Gangi model, initial stress, σ (MPa).
R4	real	1	Variable parameter used in Gangi model, exponent m.
R5	real	1.e30	Variable parameter used in Gangi model, pressure P_0 (MPa).

For the linear model P_0 is PEIN from macro **init**, ϕ_0 is PSD from macro **rock,** and k_0 is from macro **perm**. For either porosity model, permeability is given by: $k = k_0\left(\frac{\phi}{\phi_0}\right)^3$

6.2.33 Control statement pres (required if macro init not used)

Group 1 - JA, JB, JC, PHRD, TIND, IEOSD (JA, JB, JC - defined on page 22)

The initial values defined in control statement **pres** supersede all others. Note that the term "saturated" is a thermodynamic definition, and not the groundwater hydrology definition, (volumetric fraction of pore void that is filled with water -- IEOSD = 1). Saturated here indicates that vapor and liquid phases exist simultaneously. The superheated region means that all pore space is filled with gas.

Input Variable	Format	Default	Description
PHRD	real	PEIN	Initial pressure (MPa).
TIND	real		Initial temperature (°C) if IEOSD = 1 or 3, initial saturation if IEOSD = 2

IEOSD	integer	1	Thermodynamic region parameter. If IEOSD < 0 then the code uses ABS (IEOSD) and fixes the values of PHRD and TIND to the values provided above.
			IEOSD = 1, the compressed liquid region
			IEOSD = 2, the saturation region
			IEOSD = 3, the superheated region.

The following is an example of **pres**

```
pres
  -1      0       1       0.1     0.1      2
  -2      0       1       0.1     0.1      2
  -3      0       1       0.1     0.003    2
  -4      0       1       0.1     0.1      2
  -5      0       1       0.1     0.11     2
  -6      0       1       0.1     0.11    -2
   1    800       1       0.1     0.5      2
```

6.2.34 Control statement **ptrk** (optional, cannot be used with **trac**)

Group 1 - NPART, RSEED

Group 2 - DAYCS, DAYCF, DAYHF, DAYHS

Group 3 - TRAK_TYPE, HALF_LIFE, POUT, PRNT_RST

Group 4 - TRANSFLAG(JJ), KD(JJ), TCLX(JJ), TCLY(JJ), TCLZ(JJ), DIFFMAT(JJ), RD_FRAC(JJ), MATRIX_POR(JJ), FSPACING(JJ)

Group 5 - JA, JB, JC, ITRC (JA, JB, JC - defined on page 22)

Group 6- JA, JB, JC, PCNSK, T1SK, T2SK (JA, JB, JC - defined on page 22)

Group 4 is used to define models in which identical sorption and transport parameters are assumed to apply. Group 4 data are read until a blank line is encountered. The model number JJ is incremented by 1 each time a line is read.

The concentration output is written to the .trc, .out, AVS concentration output files, and the .fin file, if specified (non-zero value of PRNT_RST).

Notes on Restarting: As with all restart runs for FEHMN, a .ini file is specified to be read to set the initial conditions upon restarting. However,

Input Variable	Format	Description
NPART	integer	Number of particles in the simulation. Note: the actual number may be slightly less than the number specified by the user because when the code divides the particles among the starting nodes as specified in Group 7, the code must input an integer number of particles at each node.
RSEED	integer	6-digit integer random number seed.
DAYCS	real	Time which the particle tracking solution is enabled (days).
DAYCF	real	Time which the particle tracking solution is disabled (days).
DAYHF	real	Time which the flow solution is disabled (days).
DAYHS	real	Time which the flow solution is enabled (days).
TRAK_TYPE	integer	Flag to denote the fluid phase of the particles: 1 - liquid phase particles 2 - vapor phase particles
HALF_LIFE	real	Half-life for irreversible first order decay reaction (s). Set HALF_LIFE = 0 for no decay.
POUT	integer	Flag to specify the concentration output: 1 - Concentrations computed as number of particles per unit total volume (rock and fluid) 2 - Concentrations computed as number of particles per unit fluid volume (the fluid is liquid for TRAK_TYPE = 1 and gas for TRAK_TYPE = 2). 3 - Concentrations computed as number of particles at a given node point. 4 - Used for radioactive particle mixing model (only liquid tracer). For meaningful results the particles must all be injected simultaneously in a pulse (give a very short duration of injection starting at time 0). The file getconc.f contains data describing the function f(t) vs. time where f(t) is given as $\int_0^t C(t)\exp(-kt)dt$ The meaningful output are the final concentrations after all the particles have left the system. -1, -2, -3, or -4 - Concentrations computed as specified above for abs(pout). The .trc file contains breakthrough output for the first node specified in the node macro. 0 - Concentration output is a running total of the number of particles which have left each node, divided by the fluid or vapor mass at that node depending on trak_type.

Input Variable	Format	Description
PRNT_RST	integer	Flag to specify whether particle information is written to the ".fin" file: 0 - Particle information is not written to ".fin" file. 1 - Particle information is written to the ".fin" file. -1 - Particle positions and ages are written to the ".fin" file. When particle tracking data are written to the .fin file, the arrays are written after all of the heat and mass simulation information. The information written is sufficient to perform a restart of the particle tracking simulation and to post-process the data to compile statistics on the particle tracking run. However, for a large number of particles, this file can become quite large, so particle tracking information should only be written when necessary. Thus, 0 should be used for PRNT_RST unless restarting or post-processing to obtain particle statistics are required. Selecting the -1 option allows a subset of the full set of information needed for a restart (particle positions and ages) to be written. Restart runs that use this file as input will only be approximate, since the particle is assumed to have just entered its current cell. For restart runs, PRNT_RST = 1 is preferred, while PRNT_RST = -1 is appropriate for output of particle statistics for post-processing.
TRANSFLAG	integer	Flag to specify which transport mechanisms apply: 1 - advection only (no dispersion or matrix diffusion) 2 - advection and dispersion (no matrix diffusion) 3 - advection and matrix diffusion (no dispersion) 4 - advection, dispersion, and matrix diffusion
KD	real	Sorption coefficient (linear, reversible, equilibrium sorption). Units are kg-fluid / kg-rock (these units are equivalent to the conventional units of cc/g when the carrier fluid is water at standard conditions). This value applies to the medium as a whole when matrix diffusion is turned off, whereas for simulations invoking matrix diffusion, the value applies to the rock matrix. For the latter case, sorption in the flowing system (fractures) is modeled using the RD_FRAC variable.
TCLX	real	Dispersivity in the x-direction (m). The input value is ignored when dispersion is turned off.
TCLY	real	Dispersivity in the y-direction (m). The input value is ignored when dispersion is turned off.
TCLZ	real	Dispersivity in the z-direction (m). The input value is ignored when dispersion is turned off.
DIFFMAT	real	Molecular diffusion coefficient in the rock matrix (m^2/s). The input value is ignored unless matrix diffusion is invoked.
RD_FRAC	real	Retardation factor within the primary porosity (fractures) for a matrix diffusion particle tracking simulation (use 1 for no sorption on fracture faces). The input value is ignored unless matrix diffusion is invoked.

Input Variable	Format	Description
MATRIX_POR	real	Porosity of the rock matrix. Used to simulate diffusion and sorption in the rock matrix when matrix diffusion is invoked. Note: when matrix diffusion is turned off, particle transport through the medium is computed using the porosity set in the **rock** macro, and the input value of MATRIX_POR is ignored.
FSPACING	real	Mean fracture spacing (m). When matrix diffusion is invoked, the mean fracture aperture (a parameter in the matrix diffusion model) is computed as fracture porosity (from the **rock** macro) divided by FSPACING. When matrix diffusion is turned off, the value of FSPACING is ignored.
ITRC	integer	Model number for parameters defined in group 4. Default is [1].
PCNSK	real	Particle injection parameter assigned for nodes defined by JA, JB, and JC. Two options are available: PCNSK > 0 - particles are injected at each node in proportion to the source mass flow rate at the node. When multiple lines of input are given for Group 6, PCNSK is proportional to the particle injection concentration. This boundary condition is equivalent to injecting a solute of a given concentration into the system. Note: the source flow rates used to assign the number and timing of particle injections are those at the beginning of the particle tracking simulation (time DAYCS). Transient changes in this source flow rate during the particle tracking simulation do not change the input of particles to the system. PCNSK < 0 - particles are introduced at the node(s), regardless of whether there is a fluid source at the node. When multiple lines of input are given for Group 6, abs(PCNSK) is proportional to the number of particles introduced at the node(s). When multiple lines of input are given for Group 6, all PCNSK values must have the same sign (i.e. the two options cannot be invoked in the same simulation). Default is 0 for all unassigned nodes, meaning that no particles are injected at that node.
T1SK	real	Time (days) when particle injection begins. Default is [0].
T2SK	real	Time (days) when particle injection ends. Default is [0].

there are two possibilities for restart calculations with particle tracking: 1) the heat and mass transfer solution is being restarted, but the particle tracking simulation is initiated during the restart run (it was not carried out in the simulation that generated the .ini file); or 2) the heat and mass transfer solution and the particle tracking simulation are both being restarted. If the code does not find the "ptrk" key word at the top of the .ini file, then the original run did not employ particle tracking, and Case 1 is assumed. A common example is a preliminary calculation that establishes a fluid flow steady state, followed by a restart simulation of transport.

If "ptrk" was written into the .ini file in the original run, the particle data in the .ini file is read and used to initialize the particle tracking simulation (Case 2). In this instance, the number of particles (NPART) must be set the same for the restart run as in the original run or the results will be unpredictable.When restarting a particle tracking simulation, certain input data are overwritten by information in the .ini file. These parameters include RSEED, PCNSK, T1SK, and T2SK. Other input parameters can be set to different values in the restart run than they were in the original run, but of course care must be taken to avoid physically unrealistic assumptions, such as an abrupt change in transport properties of Group 4 part way through a simulation.

A final note on restart calculations is in order. A common technique in FEHM restart calculations is to reset the time at the top of the .ini file to 0, so that the starting time of the restart simulation is arbitrarily 0, rather than the ending time of the original simulation. This is useful for the example of the steady state flow calculation, followed by a restart solute transport calculation. Although this technique is acceptable for particle tracking runs that are initiated only upon restart (Case 1), it is invalid when a particle tracking run is being resumed (Case 2). The reason is that all particle times read from the .ini file are based on the starting time of the original simulation during which the particle tracking simulation was intiated.

The following is an example of **ptrk**:

```
ptrk
 100000   122945
  10.      20.      10.      20.
   1        0        2        0
   4        0.       2.       2.       2.      5.e-11     1.      0.1     0.333
   4        3.       2.       2.       2.      1.e-10     1.      0.28    2.

   1        0        0        1
  -2        0        0        2

  -3        0        0        1.       10.     10.0001
```

In this example, 100,000 nondecaying, liquid-borne particles are introduced as a sharp pulse (from time 10 to 10.0001 days) with the injection fluid in zone 3 (an injection well defined in the **zone** macro preceeding **ptrk**). The particle tracking simulation starts as the heat and mass transfer simulation is turned off at day 10, after having established a fluid flow steady state. Two models are defined for assigning transport properties of the particles. All nodes are assigned to model 1, after which model 2 properties are assigned for zone 2. A combined advection, dispersion, and matrix diffusion model is used for all nodes. However, sorption in the matrix occurs only for model 2 (which is zone 2 in this simulation), and the matrix transport properties (porosity, fracture spacing, diffusion coefficient) differ for this model as well.

6.2.35 Control statement **renm** (optional)

Group 1 - JA, JB, JC, IGD (JA,JB,JC - defined on page 22)

Input Variable	Format	Description
IGD	integer	New node number for given node.

6.2.36 Control statement **rflx** (optional)

Radiation heat source term. Not implemented in this version. A negative heat flux indicates heat flow into the reservoir.

Group 1 - EMISS

Group 2- JA, JB, JC, QFLUX, QFLXM (JA, JB, JC - defined on page 22)

Input Variable	Format	Description
EMISS	real	Emissivity.
QFLUX	real	If QFLXM = 0, then QFLUX is the heat flux (MW). If QFLXM ≠ 0, then QFLUX is a temperature and the heat flux is calculated according to the formula: $Q_H = QFLXM(TL - QFLUX)$ (MW). [0]
QFLXM	real	multiplier for heat flux equation given in QFLUX description (MW/°C). If QFLXM = 0, then QFLUX is the heat flux (MW). [0]

6.2.37 Control statement **rlp** (optional)

Relative permeability and capillary pressure model.

Group 1 - IRLP(i), RP1, RP2, RP3, RP4, RP5, RP6, RP7, RP8, RP9, RP10, RP11, RP12, RP13, RP14, RP15 (number of parameters depends on model selected)

Group 2 - JA, JB, JC, I (JA, JB, JC - defined on page 22)

Only those parameters defined for a given model need to be input. Group 1 is ended when a blank line is encountered. The parameter *i* is incremented each time a Group 1 line is read. Group 2 lines will refer to this parameter. Five models are available. For model number 4 (the combined van Genuchten model), the permeability is isotropic and overwrites the input from macro **perm**.

Input Variable	Format	Description
IRLP(i)	integer	Relative permeability model type.
Model 1: IRLP(i) = 1, linear relative permeability, linear capillary pressure (6 parameters required).		
RP1	real	Residual liquid saturation.
RP2	real	Residual vapor saturation.
RP3	real	Maximum liquid saturation.
RP4	real	Maximum vapor saturation.
RP5	real	Capillary pressure at zero saturation (Mpa).
RP6	real	Saturation at which capillary pressure goes to zero.
Model 2: IRLP(i) = 2, Corey relative permeability, linear capillary pressure (4 parameters required).		
RP1	real	Residual liquid saturation.
RP2	real	Residual vapor saturation.
RP3	real	Capillary pressure at zero saturation (Mpa).
RP4	real	Saturation at which capillary pressure goes to zero.
Model 3: IRLP(i) = 3, van Genuchten relative permeability, van Genuchten capillary pressure (6 parameters required).		
RP1	real	Residual liquid saturation.
RP2	real	Maximum liquid saturation.
RP3	real	Inverse of air entry pressure, α_G (1/m) [note some data is given in (1/Pa) convert using pressure = $\rho g \Delta h$].
RP4	real	Power n in van Genuchten formula.
RP5	real	Low saturation fitting parameter, multiple of cutoff capillary pressure assigned as maximum capillary pressure. If RP5 < 0 then a linear fit from this cutoff saturation (RP6 is used). The slope of the cutoff saturation is used to extend the function to saturation = 0. If RP5 = 0, a cubic fit is used. The slope at the cutoff saturation is matched and the conditions $\frac{\partial}{\partial S}Pcap = 0$ and $\frac{\partial^2}{\partial S}Pcap = 0$ are forced at $S = 0$. If RP5 > 0, a multiple of the value of the capillary pressure at the cutoff saturation, $RP5 \bullet Pcap(S_{cutoff})$ is forced at $S = 0$.

Input Variable	Format	Description
RP6	real	Cutoff saturation used in fits described for RP5, must be greater than RP1.
Model 4: IRLP(i) = 4, van Genuchten relative permeability, van Genuchten capillary pressure, effective continuum (15 parameters required).		
RP1	real	Residual liquid saturation, matrix rock material.
RP2	real	Maximum liquid saturation, matrix rock material.
RP3	real	Inverse of air entry pressure, α_G (1/m) [note some data is given in (1/Pa) convert using pressure = $\rho g \Delta h$], matrix rock material.
RP4	real	Power n in van Genuchten formula, matrix rock material.
RP5	real	Low saturation fitting parameter, matrix rock material, multiple of cutoff capillary pressure assigned as maximum capillary pressure. If RP5 < 0 then a linear fit from this cutoff saturation (RP6 is used). The slope of the cutoff saturation is used to extend the function to saturation = 0. If RP5 = 0, a cubic fit is used. The slope at the cutoff saturation is matched and the conditions $\frac{\partial}{\partial S}Pcap = 0$ and $\frac{\partial^2}{\partial S}Pcap = 0$ are forced at $S = 0$. If RP5 > 0, a multiple of the value of the capillary pressure at the cutoff saturation, $RP5 \bullet Pcap(S_{cutoff})$ is forced at $S = 0$.
RP6	real	Cutoff saturation used in fits described for RP5, must be greater than RP1, matrix rock material.
RP7	real	Residual liquid saturation, fracture material.
RP8	real	Maximum liquid saturation, fracture material.
RP9	real	Inverse of air entry pressure, α_G (1/m) [note some data is given in (1/Pa) convert using pressure = $\rho g \Delta h$], fracture material.
RP10	real	Power n in van Genuchten formula, fracture material.

Input Variable	Format	Description
RP11	real	Low saturation fitting parameter, fracture material, multiple of cutoff capillary pressure assigned as maximum capillary pressure. If RP11 < 0 then a linear fit from this cutoff saturation (RP6 is used). The slope of the cutoff saturation is used to extend the function to saturation = 0. If RP11 = 0, a cubic fit is used. The slope at the cutoff saturation is matched and the conditions $\frac{\partial}{\partial S}Pcap = 0$ and $\frac{\partial^2}{\partial S}Pcap = 0$ are forced at $S = 0$. If RP11 > 0, a multiple of the value of the capillary pressure at the cutoff saturation, $RP11 \bullet Pcap(S_{cutoff})$ is forced at $S = 0$.
RP12	real	Cutoff saturation used in fits described for RP11, must be greater than RP7, fracture material.
RP13	real	Matrix rock saturated permeability (m^2).
RP14	real	Fracture permeability (m^2).
RP15	real	Fracture porosity.
Model 5: IRLP(i) = 5, a file containing the relative permeability and capillary models is required. The name of this file is given on the line following IRLP(i) = 5. The models listed above are available. (See example.)		
I	integer	Number referring to the sequence of models read in Group 1. The default is [1].

The following are examples of **rlp**

```
rlp
   2         0.3        0.1        2.0        1.

   1     140            1          1
```

```
rlp
   5
  rlp.dat
```

File rlp.dat:

```
  2      0.3     0.1     2.      1.
  2      0.3     0.1     2.      1.
  .
  .
  .
  .
  2      0.3     0.1     2.      1.
```

number of grid blocks

6.2.38 Control statement rock (required)

Assign rock density, specific heat and porosity.

Group 1 - JA, JB, JC, DENRD, CPRD, PSD (JA,JB,JC - defined on page 22)

Input Variable	Format	Description
DENRD	real	Rock density (kg/m^3).
CPRD	real	Rock specific heat ($\frac{MJ}{kg \cdot K}$). If CPRD > 1 the code will assume the units are ($\frac{J}{kg \cdot K}$) and multiply by 10^{-6}.
PSD	real	Porosity.

The following is an example of **rock**

```
rock
   1        140          1         2563.        1010.          0.3500
```

6.2.39 Control Statement rxn (optional)

Chemical reactions between species are invoked with this control statement. It is used in conjunction with control statement **trac**.

Group 1 - KEY_GROUP

Group 2 - NGROUPS

Group 3 - GROUP(ISPECIES), ISPECIES = 1, NSPECI (repeated NGROUPS times, once for each group)

Group 4 - NRXNS, RXN_INTERVAL

For each reaction, there are two possibilities that require different input: equilibrium reaction (KEY_RXN = "equilibrium") and kinetic reaction (KEY_RXN = "kinetic"). These two choices may be mixed in a given simulation to simulate a combination of equilibrium and kinetic reactions. Group 5 parameters are repeated NRXN times, once for each reaction. Within the an equilibrium reaction option, there are also two options based on the choice of the EQUIL_MODEL parameter. For EQUIL_MODEL = 1: the following input is used:

Group 5 - KEY_RXN, EQUIL_MODEL, EQUIL_CONST25, ENTHALPY25, GAMMA_CHECK, RATE_FACTOR, ROUND_TOL

For EQUIL_MODEL = 2, the input for this group is:

Group 5 - KEY_RXN, EQUIL_MODEL, AWWA(1), AWWA(2), AWWA(3), AWWA(4), AWWA(5), GAMMA_CHECK, RATE_FACTOR, ROUND_TOL

Finally, for a kinetic reaction the following input is used:

Group 5- KEY_RXN, AR_FOR, EA_FOR, AR_REV, EA_REV

Group 6 - STOIC(ISPECIES), ISPECIES = 1, NSPECI (repeated NRXN times, once for each reaction)

Group 7 - RATE_POWER(ISPECIES), ISPECIES= 1, NSPECI (repeated NRXN times, once for each reaction)

Group 8 - FL_MULT(ISPECIES), ISPECIES= 1, NSPECI (repeated NRXN times, once for each reaction)

Group 9 - SB_MULT(ISPECIES), ISPECIES= 1, NSPECI (repeated NRXN times, once for each reaction)

Group 10 -H_MULT(ISPECIES), ISPECIES= 1, NSPECI (repeated NRXN times, once for each reaction)

Input for Groups 8 and 9 may be omitted by entering a blank line after the entries for Group 7. In this case, all reactions are assumed to occur with both fluid- and sorbed-phase solutes. If this is the case, or if there are no sorbing solutes in the simulation, then Groups 8 and 9 need not be entered. To specify the nature of the reactions involving sorbed-phase solutes (i. e., whether to compute the rate based on fluid concentration, sorbed-phase concentration, or both), see the variable descriptions below. Note that if any values in Group 8 or 9 are to be set other than to their default values, then all values for these two groups are required.

If there are no Henry's Law species present in the simulation then Group 10 should be omitted. However, this group is necessary when a Henry's Law species is present, even if it does not participate in any reactions.

Input Variable	Format	Description
KEY_GROUP	character	Key word to specify that species are to be placed into groups that are solved simultaneously. If "gr" is found as the first two characters of the line immediately after the **rxn** control statement, then the code reads in the species group information (Groups 2 and 3) of input. Otherwise, the code assumes that each species is to be solved separately with an outer iterative loop to ensure overall convergence.
NGROUPS	integer	Number of groups of species. The species assigned to a group (using the GROUP input below) are solved simultaneously. If there is only one group, the solution is complete when convergence is achieved. If there is more than one group, an outer iterative loop over all groups is employed to ensure overall convergence.
GROUP	integer	NSPECIES values are entered for each line of input, and NGROUPS lines of input are required, one for each group. If a value is non-zero, then that species is present in the group. A value of zero denotes that the species is not present in the group. No more than four species can be assigned to any group, but there is no restriction on the number of groups that a species can be assigned to. Grouping of species that take part in rapid kinetics or equilibrium reactions is required for convergence. However, memory requirements increase as the square of the maximum number of species in a group.
NRXNS	integer	Number of chemical reactions
RXN_INTERVAL	real	This parameter allows the user to choose between two different iteration schemes as follows (see Model and Methods Summary for a description of these solution procedures): RXN_INTERVAL = 0 - Solution scheme 1 always used. RXN_INTERVAL > 0 - Solution scheme 2 will be used RXN_INTERVAL times between each use of scheme 1.
KEY_RXN	character	Denotes the type of reaction. The first letter of the keyword is all that is required, but it must appear as the first character of the line. "equilibrium" - equilibrium reaction "kinetic" - kinetic reaction
EQUIL_MODEL	integer	Flag denoting which model to be used for defining the temperature dependence of the equilibrium constant 1 - van't Hoff model 2 - Multi-parameter fit to experimental data for carbonate system.
EQUIL_CONST25	real	The term A_{rxn} in equilibrium constant temperature dependence model.

Input Variable	Format	Description
ENTHALPY25	real	The term ΔH_H in equilibrium constant temperature dependence model.
GAMMA_CHECK	real	Equilibrium tolerance parameter γ_{tol}. The reaction will be required to be at equilibrium to within γ_{tol} at every node. For example, a value of 10^{-2} (recommended value) means that the reaction is at least 99% to equilibrium everywhere.
RATE_FACTOR	real	Parameter for scaling the rate constants used in the code for simulating equilibrium behavior. Recommended value: 10^{-3}
ROUND_TOL	real	Cut-off parameter for forward reaction rate below which the check for equilibrium behavior is not made. Recommended value: 10^{-10}
AWWA(1)	real	The term $A_{rxn,1}$ in equilibrium constant temperature dependence model.
AWWA(2)	real	The term $A_{rxn,2}$ in equilibrium constant temperature dependence model.
AWWA(3)	real	The term $A_{rxn,3}$ in equilibrium constant temperature dependence model.
AWWA(4)	real	The term $A_{rxn,4}$ in equilibrium constant temperature dependence model.
AWWA(5)	real	The term $A_{rxn,5}$ in equilibrium constant temperature dependence model.
AR_FOR	real	Pre-exponential factor of the forward reaction [Equation (85) in FEHMN.MMS.doc]. In keeping with the conventional method of defining rate constants, this parameter has units of [(concentration units)^p x s]$^{-1}$, where p is the sum of the exponents on all concentrations in the forward reaction minus 1. Thus, the units of the reaction rate are (concentration units) / s.
EA_FOR	real	Activation energy of the forward reaction (J/mol).
AR_REV	real	Pre-exponential factor of the reverse reaction.
EA_REV	real	Activation energy of the reverse reaction.
STOIC	real	For each solute, the stoiciometric coefficient [the a's in Equation (83) in FEHMN.MMS.doc] for the particular reaction. If positive, the solute is a reactant; if negative, the solute is a product; and if 0, the solute is not present in the reaction.

Input Variable	Format	Description
RATE_POWER	real	For each solute, the exponent in the rate law [the b's in Equation (84) in FEHMN.MMS.doc] for the particular reaction. If the corresponding value of STOIC is positive (the solute is a reactant), then b = RATE_POWER; if the corresponding value of STOIC is negative (the solute is a product), then b = - RATE_POWER; and if the corresponding value of stoic is 0, this value of RATE_POWER is ignored. The reason for this convention is that when the law of microscopic reversibility applies (the exponents in the rate law are the stoiciometric coefficients), the values of stoic and RATE_POWER are identical. Note also that a value of 0 for RATE_POWER when the corresponding value of STOIC is non-zero implies a zero-th order reaction, and does not mean that the reactant is absent from the reaction.
FL_MULT	real	For each solute, a parameter signifying whether this particular reaction occurs for this solute in the fluid phase. Set the parameter to a nonzero value if the solute in the fluid phase is involved in the reaction, otherwise set the parameter to 0. If this solute is not present in the reaction (the corresponding value of STOIC is 0), then the value of FL_MULT is irrelevant.
SB_MULT	real	For each solute, a parameter signifying whether this particular reaction occurs for this solute in the sorbed phase. Set the parameter to a nonzero value if the solute in the sorbed phase is involved in the reaction, otherwise set the parameter to 0. If this solute is not present in the reaction (the corresponding value of STOIC is 0), then the value of SB_MULT is irrelevant.
H_MULT	real	For each Henry's Law species, set this parameter to 1 if the liquid-borne fraction of the solute is reacting, or to -1 if the vapor-borne portion of the species is reacting. For each non-Henry's law species this parameter is not used but is read in to keep a similar input format to SB_MULT and FL_MULT. Therefore, H_MULT *must* be set for *each* species in *every* reaction if a Henry's law species is present in the simulation. Both the liquid- and vapor-borne portions of a Henry's Law species cannot react in one reaction. However, *two* reactions, one with the liquid portion reacting and the other with the vapor portion reacting, allows for both phases of the Henry's Law species to react.

The following is an example of rxn

```
rxn
group
  2
  1            0            0
  0            1            1
  2            0
kinetic        3.1688e-11   0.           3.1688e-10  0.
equilibrium    1            0.2          0.          1.e-2       1.e-3       1.e-10
  1           -1            0
  0            1           -1
  1           -1            0
  0            1           -1
  1            1            1
  1            1            1
  0            0            0
  0            0            0
```

This example, along withthe example data input for the **trac** macro, defines the reactive transport problem that is called Run 1 in Section 9.5. The reaction system contains three species (A, B, and C in the description below refer to species 1, 2, and 3, respectively), and the following two chemical reactions:

$$A \Leftrightarrow B$$

$$B \Leftrightarrow C$$

The reactions are reversible, with stoiciometric coefficients of 1, and the powers in the kinetic or equilibrium expressions are all 1 as well. Equilibrium sorption is specified for solute 1 (A) using the isotherm formulations in the **trac** macro. The FL_MULT and SB_MULT input are set so that the chemical reactions here pertain only to solute present in the fluid phase. Group 10 is omitted because for this example there are no Henry's law spacies.

The first reaction is kinetically controlled, and forward and reverse rate constants (with no temperature dependence) are given. The second reaction is an equilibrium reaction, and an equilibrium constant (with no temperature dependence) is given. The code will iteratively solve the system in two groups, first with A only, then with B and C coupled. Coupling of the solutes in the equilibrium reaction is necessary for convergence, but solutes that are uncoupled or weakly coupled through "slow" kinetic reactions can be solved separately to minimize computer storage requirements. If the number of finite element nodes is small enough, all three species could be coupled, and better performance would be expected if a large number of outer iterations are required. To do this in the example above, NGROUP would be set to 1, and a single line of three 1's for GROUP would denote that all three solutes are to be solved simultaneously.

6.2.40 **Control Statement sol** (required)

Group 1 - NTT, INTG

Input Variable	Format	Description
NTT	integer	Parameter that defines the type of solution required NTT > 0 coupled solution NTT ≤ 0 heat transfer only solution
INTG	integer	Parameter that defines element integration type INTG ≤ 0 Lobatto (node point) quadrature is used, recommended for heat and mass problems without stress. INTG > 0 Gauss quadrature is used, recommended for problems requiring a stress solution. This is the default for 3D problems.

The following is an example of **sol**

```
sol
    1          -1
```

6.2.41 **Control statement solv** (Not implemented)

6.2.42 **Control Statement stea** (optional)

No input is associated with this macro statement. The code will attempt to find a steady state solution if present. Not tested.

This statement enables a 1-D solution in the y-direction (2-D) or z-direction (3-D) when gravity is present to generate an initial steady state solution.

6.2.43 **Control Statement stop** (required)

No input is associated with this control statement. It signals the end of input, and as such it always appears as the last line of an input deck.

6.2.44 **Control Statement strs** (Not implemented in this version of FEHMN)

6.2.45 **Control Statement text** (optional)

Group 1- WDD1

Input Variable	Format	Description
WDD1	character*80	Line of text. A maximum of 80 characters per line are entered. Text is input until a blank line is inserted to signal the end of the control statement. This text is written to the output file (iout).

The following is an example of **text**

```
text
This is a 2-d model of the PACE problem
It will be used to study thermal effects
user # = -20 to get waste packages
```

6.2.46 Control statement thic (optional)

Input for variable thickness for two-dimensional problems.

Group 1 - JA, JB, JC, THIC (JA, JB, JC - defined on page 22).

Input Variable	Format	Description
THIC	real	Thickness of the model domain in the third dimension (m). Default is [1].

The following is an example of **thic**

```
thic
    1          0          0          10.
   -2          0          0           5.
```

In this example, the thickness for all nodes is set to 10 m, after which the nodes defined by zone 2 are set to 5 m. Thus, the thickness is 10 m everywhere except zone 2, where thickness is 5 m.

6.2.47 Control statement time (required)

Time step and time of simulation data.

Group 1 - DAY, TIMS, NSTEP, IPRTOUT, YEAR, MONTH, INITTIME

Group 2 - DIT1, DIT2, DIT3, ITC (as needed)

DAY should be larger than DAYMIN defined in Control Statement **ctrl.** The code proceeds to the next control statement when a blank line is encountered for Group 2. This can be used to generate output at specific times (with multiple group 2s). A contour plot will be drawn at each DIT1 regardless of the input in control statement **cont**. The restart file will be written (or rewritten if one already exists) at each DIT1.

Input Variable	Format	Description
DAY	real	Initial time step size (days).
TIMS	real	Final simulation time (days).
NSTEP	integer	Maximum number of time steps allowed.
IPRTOUT	integer	Print-out interval for nodal information (pressure, enthalpy etc.), as set up under control statement node. (i.e., number of time steps).
YEAR	integer	Year that simulation starts.
MONTH	integer	Month that simulation starts.
INITTIME	real	Initial time of simulation (days). For compatibility with older versions, if this parameter is absent the initial time of simulation will be 0 if no restart file is used, or the time in the restart file if one is used.
DIT1	real	Time (days) for time step change.

Input Variable	Format	Description
DIT2	real	New time step size (days). If DIT2 < 0 then ABS (DIT2) is the new time step multiplier.
DIT3	real	Implicitness factor for new time step. DIT3 ≤ 1.0 backward Euler. DIT3 > 1.0 for second-order implicit scheme.
ITC	integer	New print-out interval.

The following is an example of **time**

```
time
  30.0      3650.0      20          5       1989        10
   1.0        -1.2       1.0       10
```

6.2.48 Control statement trac with x, y, z dispersion (optional)

Group 1 - USER_MACRO, ANO, AWC, EPC, UPWGTA

Group 2 - DAYCS, DAYCF, DAYHF, DAYHS

Group 3 - IACCMX, DAYCM, DAYCMM, DAYCMX

Group 4 - NSPECIES

There are two options for group five. If the same diffusion coefficient and dispersivities are to be used for all species of the same type (liquid and/or vapor) then a keyword must be entered. This will make the calculations more efficient and thus should be used if applicable. In the absence of a keyword the following input is used:

Group 5 - ICNS

Group 6 - IADSF, A1ADSF, A2ADSF, BETADF, DIFFM, TCX, TCY, TCZ

Group 7 - JA, JB, JC, ITRCD (JA, JB, JC - defined on page 22)

Group 8 - HENRY_MODEL, HAWWA(1), HAWWA(2), HAWWA(3), HAWWA(4), HAWWA(5) (only for a Henry's Law species)

Group 9 - JA, JB, JC, ANQO (JA, JB, JC - defined on page 22)

Group 10 -JA, JB, JC, CNSK, T1SK, T2SK (JA, JB, JC - defined on page 22)

Groups 5, 6, 7, 8, 9, and 10 are entered as a unit for each solute. However, for a solid species, only groups 5, 9, and 10 are entered (groups 6, 7, and 8 are not applicable for a solid species).

Groups 6 and 7 are used to define transport models for which sorption and dispersion parameters are identical. For a liquid or vapor species, only one set of Group 6 parameters should be entered per region. However, for a Henry's Law species, two sets of parameters per region must be entered. For this case, the liquid sorption parameters should be entered on the first line and the vapor sorption parameters on a second line or as a continuation of the first line. Groups 6 & 7 are not applicable to a solid species and should not be entered. Group 6 is read in until a blank line is encountered. The model number is incremented by 1 each time a line is read. Group 7 then assigns a transport model number to every node.

For the same diffusion coefficient and dispersivities:

Group 5 - KEYWORD

If only liquid species are present KEYWORD is *dspl*. If only vapor species are present KEYWORD is *dspv*, and if both are present KEYWORD is *dspb*. If the KEYWORD is *dspl* or *dspv* the following input is used:

Group 6 - DIFFM, TCX, TCY, TCZ

Group 7 - JA, JB, JC, ITRCDSP (JA, JB, JC - defined on page 22)

Group 8 - IADSF, A1ADSF, A2ADSF, BETADF

Group 9 - JA, JB, JC, ITRCD (JA, JB, JC - defined on page 22)

Group 10 - HENRY_MODEL, HAWWA(1), HAWWA(2), HAWWA(3), HAWWA(4), HAWWA(5) (only for a Henry's Law species)

Group 11 -JA, JB, JC, ANQO (JA, JB, JC - defined on page 22)

Group 12 -JA, JB, JC, CNSK, T1SK, T2SK (JA, JB, JC - defined on page 22)

If KEYWORD is *dspb* the following input is used.

Group 6 - DIFFML, TCXL, TCYL, TCZL, DIFFMV, TCXV, TCYV, TCZV

Group 7 - JA, JB, JC, ITRCDSP (JA, JB, JC - defined on page 22)

Group 8 - IADSF, A1ADSF, A2ADSF, BETADF

Group 9 - JA, JB, JC, ITRCD (JA, JB, JC - defined on page 22)

Group 10 - HENRY_MODEL, HAWWA(1), HAWWA(2), HAWWA(3), HAWWA(4), HAWWA(5) (only for a Henry's Law species)

Group 11 -JA, JB, JC, ANQO (JA, JB, JC - defined on page 22)

Group 12 -JA, JB, JC, CNSK, T1SK, T2SK (JA, JB, JC - defined on page 22)

Injection nodes must be specified in control statement **flow**.

Input Variable	Format	Description
USER_MACRO	character*5	Key word for invoking a solute transport user subroutine. If the word *userc* is placed in this position, then the code invokes a solute transport user subroutine at each time step. Omit this key word if there is no solute user subroutine for the simulation.
ANO	real	Initial solute concentration, set at all nodes for all species unless overwritten by a restart file input or values in group 9 below (moles/kg fluid).
AWC	real	Implicitness factor for solute solution. AWC > 1.0 gives 2nd order solution AWC ≤ 1.0 gives 1st order solution

Input Variable	Format	Description
EPC	real	Equation tolerance for solute solution
UPWGTA	real	Upstream weighting term for the solute solution. UPWGTA < 0.5 UPWGTA is set to 0.5 UPWGTA > 1.0 UPWGTA is set to 1.0
DAYCS	real	Time which the solute solution is enabled (days).
DAYCF	real	Time which the solute solution is disabled (days).
DAYHF	real	Time which the flow solution is disabled (days).
DAYHS	real	Time which the flow solution is enabled (days).
IACCMX	integer	Maximum number of iterations allowed in solute solution if time step multiplier is enabled
DAYCM	real	Time step multiplier for solute solution
DAYCMM	real	Initial time step for solute solution (days)
DAYCMX	real	Maximum time step for solute solution (days)
NSPECIES	integer	Number of solutes simulated.
KEYWORD	character*4	See comments in group specifications above. *dspl* indicates that only liquid species exist and that all of them are to be given the same diffusion coefficient and dispersivities. *dspv* similarly indicates that only vapor species exist and that all of them are to be given the same diffusion coefficient and dispersivities. *dspb* indicates that both liquid and vapor species exist, that all liquid species are to have the same above properties, and that all vapor species are to have the same above properties.
ICNS	integer	Phase designation for the ith solute -2 - Henry's Law species (input and output concentration values are gas concentrations). -1 - Vapor species. 0 - Solid species 1 - Liquid species 2 - Henry's Law species (input and output concentration values are liquid concentrations)
IADSF	integer	Adsorption model type for the ith species, ith region 0 - conservative solute 1 - linear sorption isotherm 2 - Freundlich sorption isotherm 3 - Modified Freundlich sorption isotherm 4 - Langmuir sorption isotherm
A1ADSF	real	α_1 parameter in adsorption model
A2ADSF	real	α_2 parameter in adsorption model
BETADF	real	β parameter in adsorption model

Input Variable	Format	Description
DIFFM	real	Molecular diffusion coefficient (m^2/s)
DIFFML	real	Same as above, for liquid
DIFFMV	real	Same as above, for vapor
TCX	real	Dispersivity in x-direction (m)
TCY	real	Dispersivity in y-direction (m)
TCZ	real	Dispersivity in z-direction (m)
TCXL, TCYL, TCZL	real	Same as above, for liquid
TCXV, TCYV, TCZV	real	Same as above, for vapor
ITRCD	integer	Region number for sorption and dispersion parameters given in group 6. Default is [1].
HENRY_MODEL	integer	Flag denoting which model to be used for defining the temperature dependence of the Henry's law constant 1 - van't Hoff model 2 - Multi-parameter fit to experimental data (used for carbonate system).
HAWWA(1)	real	Term in Henry's Law temperature dependence model: For model 1 - parameter value is A_H For model 2 - parameter value is $A_{H,1}$
HAWWA(2)	real	Term in Henry's Law temperature dependence model: For model 1 - parameter value is ΔH_H For model 2 - parameter value is $A_{H,2}$
HAWWA(3)	real	Term in Henry's Law temperature dependence model: For model 1 - not used For model 2 - parameter value is $A_{H,3}$
HAWWA(4)	real	Term in Henry's Law temperature dependence model: For model 1 - not used For model 2 - parameter value is $A_{H,4}$
HAWWA(5)	real	Term in Henry's Law temperature dependence model: For model 1 - not used For model 2 - parameter value is $A_{H,5}$
ANQO	real	Initial concentration of tracer, which will supercede the value given by ano in group 1. Note that if initial values are read from a restart file, these values will be overwritten. Units are moles per kg vapor or liquid for a liquid, vapor, or Henry's law species, and moles per kg of solid for a solid species. Default is [0].

Input Variable	Format	Description
CNSK	real	Injection concentration at inlet node (moles per kg liquid or vapor). If fluid is exiting at a node, then the in-place concentration is used. If cnsk < 0, then the concentration at that particular node will be held at a concentration of abs(cnsk) (default is 0 for all unassigned nodes).
T1SK	real	Time (days) when tracer injection begins. Default is [0].
T2SK	real	Time (days) when tracer injection ends. Default is [0].

The following is an example of **trac**

```
trac
 0          1          1.e-6      .5
 3.6525e6   5.47875e6  3.6525e6   5.47875e6
 50         1.2        3.6525e3   3.6525e4
 3
 1
 1          0.1        0.         1.        1.e-10      50          1.e-30  1.e-30

 1          0          0          1

 1          0          0          0.

 1          202        201        1.        3.6525e6    3.689025e6

 1
 0          0          0          1.        1.e-10      50          1.e-30  1.e-30

 1          0          0          1

 1          0          0          0.

 1
 0          0          0          1.        1.e-10      .033333     1.e-30  1.e-30

 1          0          0          1

 1          0          0          0.
```

In this example, three liquid solutes are simulated. The solute transport solution is turned on as the heat and mass solution is turned off at day 3.6525e6. Solute 1 sorbs with an equilibrium sorption K_d of 0.1; solutes 2 and 3 exhibit no sorption. All three solutes have the same transport parameters, although this is not a requirement of the code, even when the solutes are coupled through chemical reactions. All solutes start at a concentration of 0 within the model (in this case a one-dimensional column). Solute 1 is injected at a concentration of 1 for a short time interval; there is no source for solutes 2 or 3.This example is meant to be used in combination with the example given for the **rxn** macro, which defines a system of chemical reactions among the three solutes. Therefore, solutes 2 and 3 are generated only through chemical reactions (neither of these solutes appear in the system initially, and there is no injection source term for either solute).

The reactive transport problem specified here and in the example **rxn** macro is discussed further in Section 9.5.

```
trac
 0          1           1.e-6       .5
 3.6525e6   5.47875e6   3.6525e6    5.47875e6
 50         1.2         3.6525e3    3.6525e4
 3
dspl
 1.e-10     .033333     1.e-30      1.e-30

 -1         0           0           1

 1
 1          0.1         0.          1.

 1          0           0           1

 1          0           0           0.

 1          202         201         1.          3.6525e6    3.689025e6

 1
 0          0           0           1.

 1          0           0           1

 1          0           0           0.

 1
 0          0           0           1.

 1          0           0           1

 1          0           0           0.
```

6.2.49 Control statement trac with longitudinal / transverse dispersion (optional)

The required input for this option is identical to that above for x, y, z dispersion with the following exceptions.

- Between groups four and five, a new group must be added consisting only of the keyword ldsp.
- Dispersivities are no longer input in the order x, y, and then z, but instead in the order longitudinal and then transverse.

Following are the above two examples of trac, modified for use with longitudinal and transverse dispersion.

```
trac
 0            1            1.e-6        .5
 3.6525e6     5.47875e6    3.6525e6     5.47875e6
 50           1.2          3.6525e3     3.6525e4
 3
ldsp
 1
 1            0.1          0.           1.           1.e-10       50           5.0

 1            0            0            1

 1            0            0            0.

 1            202          201          1.           3.6525e6     3.689025e6

 1
 0            0            0            1.           1.e-10       50           5.0

 1            0            0            1

 1            0            0            0.

 1
 0            0            0            1.           1.e-10       .0333        .00333

 1            0            0            1

 1            0            0            0.
```

```
trac
  0          1          1.e-6      .5
  3.6525e6   5.47875e6  3.6525e6   5.47875e6
  50         1.2        3.6525e3   3.6525e4
  3
ldsp
dspl
  1.e-10     .0333      .00333

  -1         0          0          1

  1
  1          0.1        0.         1.

  1          0          0          1

  1          0          0          0.

  1          202        201        1.        3.6525e6    3.689025e6

  1
  0          0          0          1.

  1          0          0          1

  1          0          0          0.

  1
  0          0          0          1.

  1          0          0          1

  1          0          0          0.
```

6.2.50 Control statement user (optional)

Group 1 - KK

Input Variable	Format	Description
KK	integer	Integer number passed to subroutine **user** for user defined input parameters. This user subroutine call differs from one invoked in the control file in that whereas that subroutine is called at every time step, this one is called only at the beginning of the simulation to set parameters that do not change later in the simulation.

6.2.51 Control Statement vcon (optional)

Group 1 - IVCON(i), VC1F(i), VC2F(i), VC3F(i)

Group 2 - JA, JB, JC, IVCND (JA, JB, JC - defined on page 22)

The parameter (i) is incremented each time Group 1 is read. Group 2 lines will refer to this parameter. Group 1 is ended with a blank line.

Input Variable	Format	Description
IVCON(i)	integer	Model type for ith conductivity model. IVCON(i) = 1, linear variation of thermal conductivity with temperature. IVCON(i) = 2, square root variation of thermal conductivity with liquid saturation.
VC1F(i)	real	Reference temperature (°C) for IVCON(i) = 1. Conductivity ($\frac{W}{m \cdot K}$) at liquid saturation = 1 for IVCON(i) = 2.
VC2F(i)	real	Reference conductivity ($\frac{W}{m \cdot K}$) for IVCON(i) = 1. Conductivity ($\frac{W}{m \cdot K}$) at liquid saturation = 0 for IVCON(i) = 2.
VC3F(i)	real	Change in conductivity with respect to temperature for IVCON(i) = 1. Not used for IVCON(i) = 2.
IVCND	integer	Number referring to the sequence of models read in Group 1. The default is [1].

6.2.52 Control Statement velo (optional)

The input for this macro is identical to macro **flxo**, except that velocities instead of fluxes are calculated (see page 38).

6.2.53 Control statement wlbr (optional)

Not supported in this implementation of FEHM.

6.2.54 Control statement zone (optional)

Geometric definition of grid for input parameter assignment, the default is input by nodes.

Group 1- IZONE

Group 2- X1, X2, X3, X4 (for 2-D) or X1, X2, X3, X4, X5, X6, X7, X8 (for 3-D)

Group 3- Y1, Y2, Y3, Y4 (for 2-D) or Y1, Y2, Y3, Y4, Y5, Y6, Y7, Y8 (for 3-D)

Group 4- Z1, Z2, Z3, Z4, Z5, Z6, Z7, Z8 (for 3-D problems only)

The following alternate form of input may be used (starting with Group 2):

Group 2 - MACRO

Group 3 - XG, YG (for 2D) or XG, YG, ZG (for 3-D) [used with 'list' option]

or

Group 3 - NIN, NCORD(1), . . . , NCORD(NIN) [used with 'nnum' option]

The geometric zone description is implemented by defining geometric regions. The coordinates given in Group 2, 3, and 4 refer to node positions shown in Fig. 2. All properties defined by node (JA, JB, JC) in any control statements may be defined by **zone**. In the previous macro descriptions if JA < 0, then the zone IZONE = ABS (JA) is referenced.

It is a good policy to refer to the input check file to insure that node assignments have been made as expected. When X, Y, Z coordinates are used to define zones, boundaries of those zones may be slightly different than specified. This is due to the inclusion of volume from elements adjoining included nodes.

When macro statements **dpdp** and **dual** are used, additional zone definitions are automatically generated. These are identified by zones 101-200 for the first set of matrix nodes and 201-300 for the second set of matrix nodes. For example, Zone 101 corresponds to the matrix material occupying the same space occupied by the fracture identified by Zone 1. Furthermore, Zone 201 refers to the second matrix layer in the **dual** control statement.

The macro **zone** must precede the usage of a ZONE reference. **zone** is ended with a blank line. **zone** can be called more than once and regions redefined. When this is done, all previous zone definitions are eliminated. A node may be included in only a single zone at a time.

Input Variable	Format	Description
IZONE	integer	Zone identification number for geometric input.
X1-X8	real	X coordinates defining zone IZONE.
Y1-Y8	real	Y coordinates defining zone IZONE.
Z1-Z8	real	Z coordinates defining zone IZONE.
MACRO	character*4	String denoting alternate input format MACRO = "list", read a list of X, Y, Z - coordinates, one set per line until a blank line is encountered. The nodes corresponding to these coordinates make up the zone. MACRO = "nnum", read the number of specified nodes, followed by the node numbers. These comprise the zone.
XG	real	X coordinate of node to be included in IZONE.
YG	real	Y coordinate of node to be included in IZONE.
ZG	real	Z coordinate of node to be included in IZONE.
NIN	integer	Number of nodes in IZONE.
NCORD(i)	integer	NIN node numbers of the nodes to be included in IZONE.

The following is an example of **zone**

```
zone
  1
  0.00       1000.00     1000.00      0.00
1075.00      1074.00     1079.00     1080.00
  2
  0.000      1000.000    1000.00      0.00
 870.000      869.000    1074.00     1075.00
  3
  0.000      1000.000    1000.00      0.000
 860.000      859.000     869.000     870.000
  4
  0.000      1000.000    1000.00      0.000
 780.000      779.000     859.000     860.000
  5
  0.000      10000.00    1000.00      0.000
 730.000      730.000     779.000     780.000
  6
  0.000      1000.000    1000.00      0.000
 700.000      700.000     730.000     730.000
```

7.0 OUTPUT

Output is found in the code generated files (output file, write file, history plot file, solute plot file, contour plot file, contour plot file for dual or dpdp, stiffness matrix data, input check file, and AVS output files) described in Section 5.0.

Macro commands (input options) dealing with output control are **cont** (page 26), **ctrl** (page 29), **nod2** (page 44), **node** (page 45), and **time** (page 64): **cont** is used to specify output format and time intervals for contour data output (*fehmn.con*, *fehmn.dp*), **ctrl** is used to specify if element coefficients calculated in the code should be saved (*fehmn.stor*), **node** and **nod2** are used to provide nodal or coordinate positions for which general information and time history data will be output (*fehmn.out*, *fehmn.his*, *fehmn.trc*, and terminal output), and **time** provides input on the time printout interval for nodal information (*fehmn.out* and terminal output).

The code itself provides no graphical capabilities. A software environment, called the Browser provides pre-processing, processing, and post-processing capabilities and a help-index for FEHMN. A complete description of how to use the Browser is given in Section 8.5. The Browser Pre-Processor assists the user in setting up an input file by displaying the appropriate pages from this User's Manual, doing syntax checks, analyzing the input file, and providing direct access to auxiliary programs for structured and unstructured grid generation.

The Browser Processor allows the user to run FEHMN, restart it, and provides assistance in setting up a control file.

The Browser Post-Processor provides FEHMN time history plots, contour plots, and access to AVS (an AVS license is required to use).

Time history plots of the energy source, source strength, temperature, pressure, capillary pressure, and saturation are made from the *filen.his* FEHMN output files. Data from the *filen.trc* files is used to make time history tracer plots of the 10 species concentrations. Capabilities for zooming, scaling, using a log scale, and printing the data are provided.

Contour plots using 2-d quad grids and 3-d hex grids for material properties, temperature, saturation, pressure, velocities, and solute concentrations can be made from the AVS FEHMN output files. The plots can be rotated, zoomed, scaled, translated, and printed. Axis values and the color bar can be customized.

AVS provides tools for visualization and analysis of volumetric scalar and vector data. AVS FEHMN output files are available for the following node data: material properties, liquid and vapor phase values, velocity and pressure values, temperature, saturation, concentration dual, and dpdp. The AVS output files from FEHMN are written in either an ASCII or binary format that can be imported into AVS UCD graphics routines for viewing.

Two additional graphical post-processing routines are available for use with output files *filen.his*, *filen.trc*, and *filen.con* as discussed in Section IIIC of Zyvoloski et al. (1991).

Additional information on the data found in the output files is given below.

7.1 Output file (*filen.out*)

Information contained in the general output file is mostly self explanatory. The file starts with the code version, date, and time followed by the user input problem title. A summary of the I/O files used, macro control statements read, and array storage follow. Variable information for user specified nodes at user selected time intervals is written. The file ends with a summary of simulation time, number of time steps in the problem, the number of iterations taken, and total cpu time.

7.2 Write file (*filen.fin*)

The write file contains the final values of pressure, temperature, saturation, and simulation time for the run. The final version of the file is generally written at the end of the simulation. This information is also written if the minimum user supplied time step has been realized and execution is terminated. If the write file has not been specified at startup the code will use *fehmn.fin*. The primary use of the write file is as a restart file. The write file contains the following:

Code version number, date, time

Problem title

Simulation time (days)

Gas flag (ngas, h20, air)

Tracer flag (trac, ptrk, ntra)

Stress flag (strs, nstr)

Dpdp flag (dpdp, ndpd)

Dual flag (dual, ndua)

If ngas flag is set, followed by

Final temperature (°C) at each node

Final saturation (dimensionless) at each node

Final pressure (MPa) at each node

Final capillary pressure (MPa) at each node

Or if neither the air or ngas (h20) flag are set, followed by

Final temperature (°C) at each node

Final saturation (dimensionless) at each node

Final pressure (MPa) at each node

Or if air flag is set, followed by

Final saturation (dimensionless) at each node

Final pressure (MPa) at each node

If trac flag is set followed by

Number of species

Species concentration (dimensionless) for each node for each species

Or if ptrk flag is set followed by

Number of particles, final random number seed

Final node position for each particle. If the value is negative, the particle left the model domain at a fluid sink at that node.

Fractional time remaining at current node for each particle.

Multiplier to the plug flow residence time for each particle at the current node position, accounting for dispersion, sorption, and matrix diffusion effects.

Age for each particle, i.e. the time since the particle entered the system. However, if the particle has left the system, this value is the time that the particle left the system.

If the random number seed in the file is negative, the arrays for the fractional time remaining and the multiplier to the plug flow time have been omitted using the PRNT_RST = -1 option (see PRNT_RST description in the PTRK macro). A restart simulation using this input file will only approximate the behavior of particles since each particle will be assumed to have just entered the node. It is preferable to restart a particle tracking simulation using a file that contains the full restart information.

If strs (not implemented in this version)

If dpdp or dual flag was set

The above information included dual porosity/dual permeability nodes.

7.3 History plot file (*filen.his*)

The history plot file contains the following:

Code version number, date, time

Problem title

Tracer flag ('trac' or blank)

Stress flag ('strs' or blank)

Number of nodes for which data is output

Node number and X, Y, and Z coordinate (m) of each node for which data is output

'headings'

'node flow enthalpy(Mj/kg) flow(kg/s) temperature (deg C) total pressure (Mpa)'

'capillary pressure(Mpa) saturation(kg/kg)'

And for each time step

Time (days) followed by

Node number, Energy source (MJ/s), Source strength (kg/s), Temperature (°C), Pressure (MPa), Capillary pressure (MPa), Saturation (dimensionless) for each specified output node.

7.4 Solute plot file (*filen.trc*)

Solute data is output for the same nodes used for the history plot file. The nodal information (see Section 7.3) is not duplicated in this file so if it is needed the history plot file must also be used. The solute plot file contains:

Code version number, date, time

Problem title

Number of different species for tracer solution

and for each time step

Time (days), species number followed by

Species concentration (dimensionless) for each specified output node.

When particle tracking is used, the concentration can be output in several different forms (number of particles, number per fluid mass, or number per total volume). The choice of which form to use is input in the **ptrk** macro.

7.5 Contour plot file (*filen.con*)

The contour plot file contains:

Code version number, date, time

Problem title

Tracer ('trac') solution or blank

Stress ('strs') solution or blank

Number of nodes for which data is output

X, Y, and Z coordinate (m) of each node for which data is output

Number of nodes per element, total number of elements

Nodal connectivity information for each node of each element

X, Y, Z permeability (m^2) for each node

X, Y, Z thermal conductivity ($\frac{W}{m \cdot K}$) for each node

Porosity, Rock specific heat ($\frac{MJ}{kg \cdot K}$), Capillary pressure (MPa) for each node

Number of degrees of freedom per node for the current problem, Direction of gravity in problem, Value of gravity

If tracer solution is present

Number of species

and for each specified time

Time (days), injection phase (0 liquid, < 0 vapor) followed by

If injection phase is liquid

Liquid transmissibility / density, Liquid density (), Pressure - Capillary Pressure (MPa), Temperature (°C)

and if tracer solution is present

Species concentration of liquid phase

Or if injection phase is vapor

Vapor transmissibility / density, Vapor density (), Pressure (MPa), Temperature (°C)

and if tracer solution is present

Species concentration of vapor phase.

7.6 Contour plot file for dual or dpdp (*filen.dp*)

The contour plot file for dual or dpdp contains the same information as the regular contour plot file only the parameter values are for the dual porosity / dual permeability nodes.

7.7 Stiffness matrix data (*filen.stor*)

The stiffness matrix data file is used to store the finite element coefficients for each node. It eliminates the need for the code to recompute the coefficients for subsequent runs. It contains the following:

> Code version number, date, time
>
> Problem title
>
> Number of storage locations needed to store geometric input types, Number of nodes, Size of connectivity array
>
> Volume associated with each node
>
> Nodal connectivity information for each connection
>
> Position of geometric coefficient for each connection
>
> Diagonal position in connectivity array for each node
>
> Finite element geometric coefficient for each for each storage location

If stress solution is enabled

> Finite element geometric coefficient for each for each storage location for the stress module.

7.8 Input check file (*filen.chk*)

This file contains a summary of input information that may be of use in debugging or memory management of the code. The positions of maximum and minimum values of input parameters and derived quantities are given. Also provided is an analysis of array storage requirements.

7.9 Error output file (*fehmn.err*)

This file contains the code version number, date, and time followed by any error or warning messages issued by the code during a run.

7.10 AVS log output file (*filen.10001_avs_log*)

The AVS log output file contains:

> Code version number, date
>
> AVS log identifier
>
> Problem title

and for each specified time

> AVS output file prefix, Call number, and Time (days)

7.11 AVS header output files (*filen.number_type_head*)

The data types are given in Section 7.13.1.

7.11.1 ASCII header

The AVS ASCII (formatted) header files contain:

20 lines of text with information about the FEHM AVS output files. The text is followed by a one line AVS UCD file header containing:

number of nodes

number of cells

number of data components for the nodes

number of data components for the cells (currently 0)

number of data components for the model (currently 0)

7.11.2 Binary header

The AVS binary (unformatted) header files consist of 21 bytes with the following values:

number '7'indicating binary file (1 byte unsigned char)

number of nodes (4 byte int)

number of cells (4 byte int)

number of node data (4 byte int)

number of cell data (4 byte int) (currently 0)

number of model data (4 byte int) (currently 0)

7.12 AVS geometry output file (*filen.10001_geo*)

7.12.1 ASCII geometry output file

The ASCII (formatted) geometry file contains the following:

Node id and X, Y, Z coordinates for each node

Cell id, Material id, Cell type, and the list of Cell vertices

7.12.2 Binary geometry output file

The binary (unformatted) geometry file contains the following:

Number of nlist nodes (4 byte int)

Cell id, Material id, Number of nodes, Cell type (16 * num_cells)(ints)

Cell vertice list (4 * num_nlist_nodes)(ints)

X coordinates for nodes (num_nodes * 4)(floats)

Y coordinates for nodes (num_nodes * 4)(floats)

Z coordinates for nodes (num_nodes * 4)(floats)

7.13 AVS data output files (*filen.number_type_node*)

7.13.1 ASCII node data output files

All the ASCII (formatted) node data files contain the following:

Number of data components and size of each component

A label/unit string for each data component

and for each node

the associated node data (described by data type below).

7.13.1.1 Material properties *(_mat and _mat_dual)*

These data will consist of 11 fields. The order of the fields are:

Permeability in X, Y, and Z direction (m^2),

Thermal conductivity in X, Y, and Z direction ($\frac{W}{m \cdot K}$),

Porosity,

Rock specific heat ($\frac{MJ}{kg \cdot K}$),

Capillary pressure (MPa),

Relative permeability model, and

Capillary pressure model.

The dual or dpdp values for each of these fields will be written to a file with "mat_dual_node" appended to the file name.

7.13.1.2 Scalar parameters *(_sca and sca_dual)*

These data files will contain scalar data including

Saturation,

Temperature (°C) ,

Liquid pressure(MPa), and

Vapor pressure(MPa).

If dual values are calculated, they can be written to the sca_dual output file.

7.13.1.3 Vector parameters *(_vec and _vec_dual)*

These data files contain the vector values for

Liquid and Vapor velocities (m/s).

If idualp is defined, dual porosity values for the vapor phase will be written to the _vec_dual file. If idpdp is defined, double porosity/double permeability values for the liquid phase will be written.

7.13.1.4 Solute concentrations *(_con and _con_dual)*

Up to 20 fields per node can be written for solute concentrations. The number written is determined by the number of species. The dual counterparts to each will be written to the _con_dual file.

7.13.2 Binary node data output file

All the binary (unformatted) AVS data files contain the following:

node data labels (1024 byte string)

node data units (1024 byte string)

number of node components (4 byte int)

node component list (num_node_data * 4)(ints)

minimums for node data (num_node_data * 4)(floats)

maximums for node data (num_node_data * 4)(floats)

data blocks with values for each node (num_nodes * num_node_data * 4)(floats)

The data types are described above in Section 7.13.1.

8.0 SYSTEM INTERFACE

8.1 System-Dependent Features

In addition to standard intrinsic math routines only two system routines are required by the FEHMN code. The code uses a system call to get the date (subroutine dated) and a system routine to get the CPU clock time (subroutine tyming).

8.2 Compiler Requirements

FEHMN is written in Fortran 77 and C. FEHMN has been successfully compiled and run on SUN, HP, IBM RISC, SGI, and Cray computers.

8.3 Hardware Requirements

No special hardware features or environments are required by the software.

8.4 Control Sequences or Command Files

None.

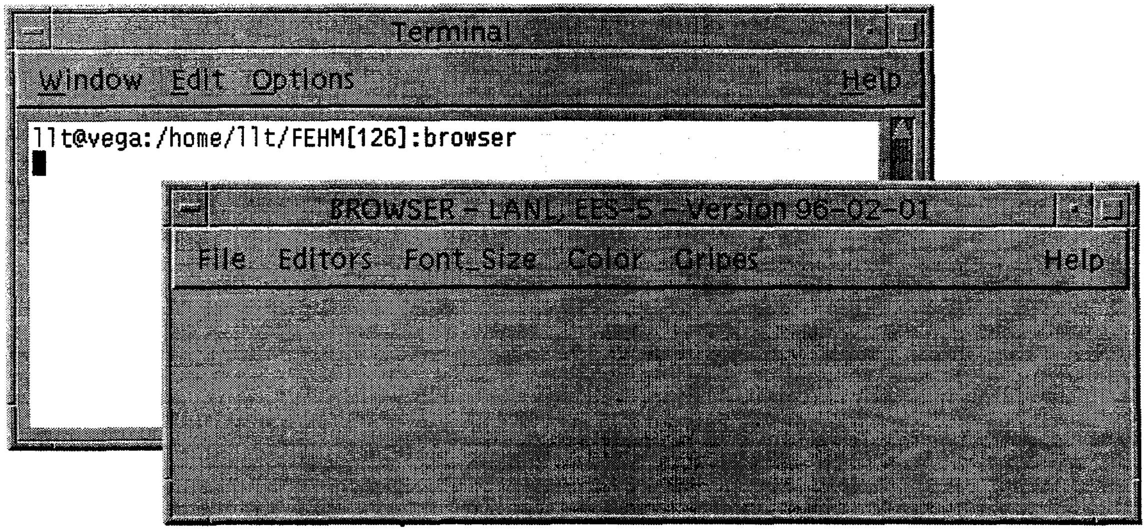
8.5 Software Environment

FEHMN can be run under a software environment called the Browser. The Browser is a column-oriented graphical user interface that provides pre-processing, processing, and post-processing capabilities and a help index for FEHMN.

8.5.1 Browser

The Browser Pre-processor assists the user in setting up an input file to be run with the Processor and provides direct access to auxiliary programs for grid generation. The Processor allows the user to run FEHMN and restart it. The Post-processor allows the user to view ascii output files, make time history plots, make contour plots, and display AVS output files generated by FEHMN (user must have an AVS license). There is also a help index that includes documentation for FEHMN and auxiliary programs. To bring up the main Control Window for the Browser under X-windows type:

browser



```
llt@vega:/home/llt/FEHM[126]:browser
```

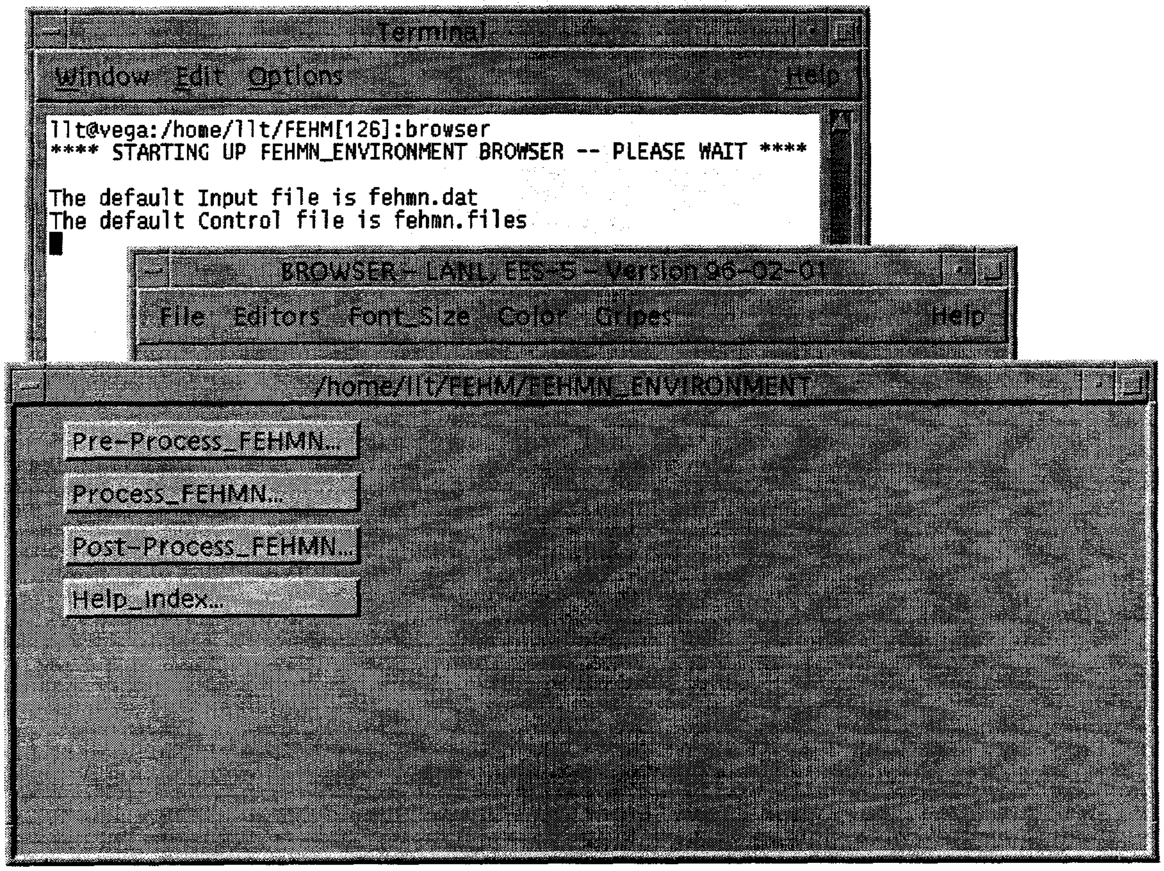
Terminal
Window Edit Options
Help
BROWSER – LANL EES-5 – Version 96-02-01
File Editors Font_Size Color Gripes
Help

- Control Window

The *Editors* button along the top allows the user to switch editors. *vi* is the default. The editors *emacs* and *fred* are menu options for the Suns. Jot is a menu option for the SGIs. The *Font_Size* button allows the user to change the font size. *A* medium size font is the default. The *Color* button allows the user to use the Browser on a black & white terminal. The *Gripes* button allows the user to report bugs and view a list of current bugs.

The *Help* button has a *Getting_Started* menu option that gives more details on using the above buttons. The *Help* button also allows the user to turn the man pages on or off. The default is to have the man pages on.

To bring up the Browser window, in the Control Window under *File*, select *Open*, highlight the file *FEHMN_ENVIRONMENT,* and click on the *OK* button. The following Browser Window will appear.



```
llt@vega:/home/llt/FEHM[126]:browser
**** STARTING UP FEHMN_ENVIRONMENT BROWSER -- PLEASE WAIT ****

The default Input file is fehmn.dat
The default Control file is fehmn.files
```

Terminal
Window Edit Options Help
BROWSER - LANL EES-5 - Version 96-02-01
File Editors Font_Size Color Gripes Help
/home/llt/FEHM/FEHMN_ENVIRONMENT
Pre-Process_FEHMN...
Process_FEHMN...
Post-Process_FEHMN...
Help_Index...

- Browser Window

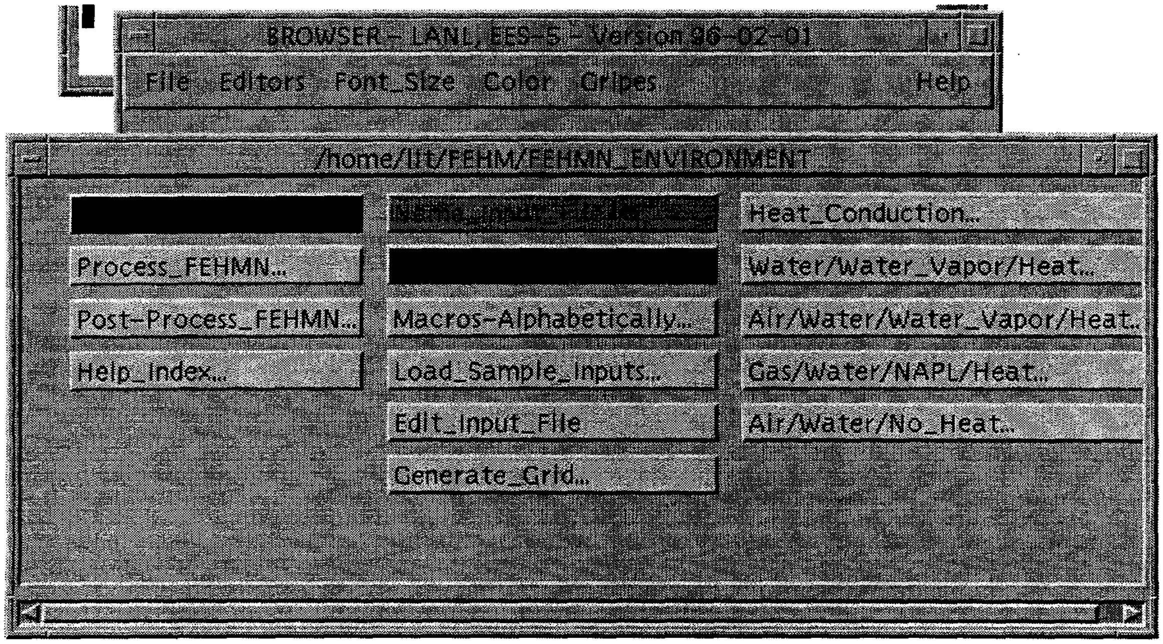
Use the Browser by clicking ONCE with the LEFT mouse button on the desired button. If button name is suffixed with "..." another column of buttons will appear when selected. Otherwise a function will be executed. Red buttons are the last buttons selected. Green buttons were pushed previously. Grey buttons have not been accessed.

The scroll bar can be used to reach buttons outside the displayed window or the window can be resized by dragging the corner of the window.

When finished, quit the Browser from the Control Window, under *File*, by selecting *Exit*.

8.5.1.1 Pre-Process FEHMN

The *Pre-process_FEHMN* button provides guidance, syntax checking, analysis, and help in creating an input file. Several auxiliary grid generating programs are provided. Click on *Pre-Process_FEHMN* to get the column with these options.



- Create Input File

  The default input file name is *fehmn.dat*. To change this, the user should click on the button *Name_Input_File*. This file name must be assigned before working on the file. There are four ways the user can work on this file. A user can input by process model type, from a list of macros, by loading a sample input file, and by editing the input file directly. A combination of these can be used.

  The *Input_by_Process_Model* button allows the user to select the type of problem from five process models and guides the user through the required and optional macros for that process type. In the example above, the *Heat_Conduction* process model was selected. The user clicks on the *Required_*_Inputs* button to get a list of the macros required for a specific process type. When a button is clicked on for a specific macro, the appropriate pages from the FEHMN User's Manual are displayed and an editing window is opened for the user to input the macro. Macros input are appended to the input file. A macro may be written to this file more than once. Displaying the FEHMN User's Manual pages can be turned off under *Help* on the main Control Window. The editor can be switched by clicking on the *Editors* button in the main Control Window. As input is entered, a syntax check is made for the correct number and type of arguments. Corrections are made, if possible and displayed. The *Analyze_Input* button can be used to verify that all the required macros are present and to list the appropriate optional macros that have not been included. There is an option, the *Use_Sample_File_** button, to use an example input file for each process type. The example input file is loaded into this file. The *Edit_Input_File* button gives the user direct access to this input file at any time.

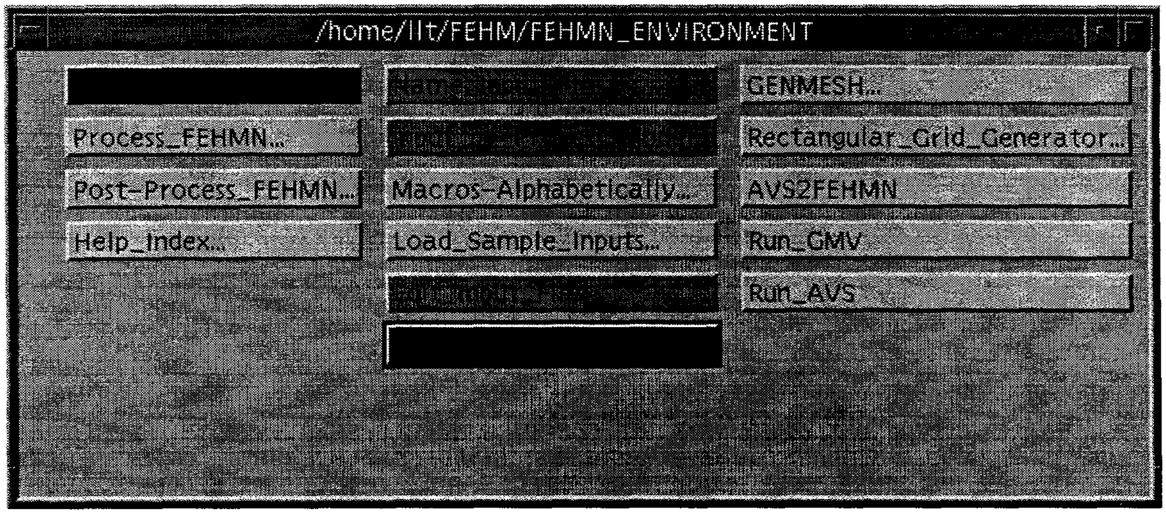
The *Macros-Alphabetically* button provides an alphabetical listing of all the macros. This mode will automatically bring up the FEHMN User's Manual pages and an editing window. Macros entered are appended to the input file, and a syntax check is made on the input. Direct access to the input file is also available through the *Edit_Input_File* button. There is no input guidance or analysis of required and optional macros.

The *Load_Sample_Inputs* button replaces or creates a new input file. Direct access to the input file is available, by going through the *Edit_Input_File* button. Access to the User's Manual is available through the *Help_Index*, in the first column.

The *Edit_Input_File* button allows the user to directly edit the input file. The editor can be changed with the *Editors* button in the Control Window. The User's Manual is available by going through the *Help_Index*, in the first column. There is no syntax checking, input guidance, or analysis on the input.

- Generate Grid

Direct access is given to several auxiliary grid generating programs. A brief description is given below. Additional help is available under the *Help_Index* button.



GENMESH is an automatic mesh generation code that directly generates the finite element mesh input required by FEHMN. Help pages and an editing window are used to create the input file. Examples input files for 2-d and 3-d grids are provided. A User's Manual is available in the Browser.

Rectangular_Grid_Generator program creates a structured grid in the format for an FEHMN input file. If the file *input.grid* is constructed, it can be run with an input file. Otherwise an interactive mode is available that prompts for the necessary inputs.

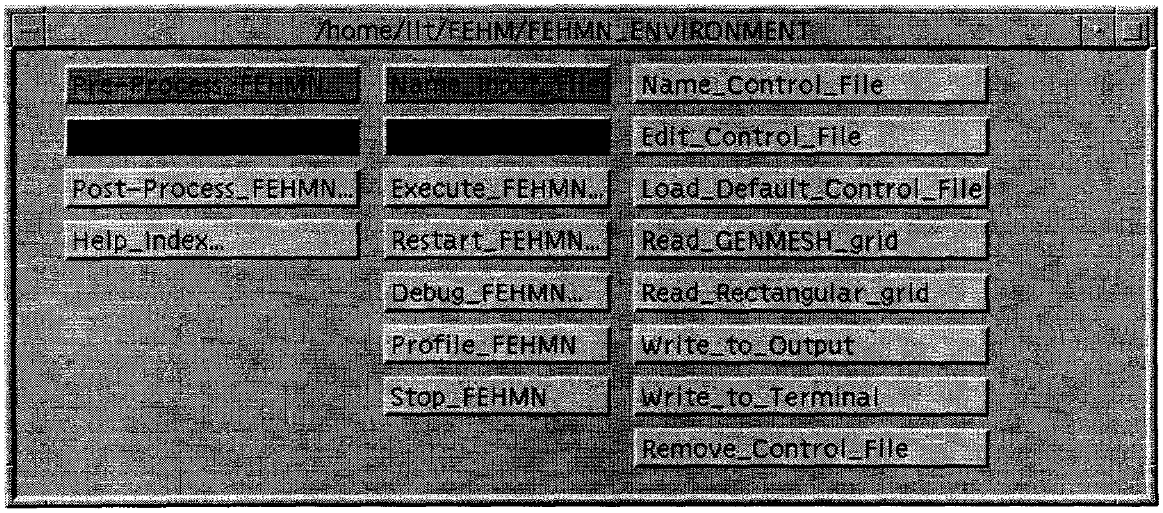
AVS2FEHMN takes an AVS file and converts it to the format for an FEHMN input file.

Run_GMV is a General Mesh Viewer, visualization tool that provides 3-D interactive graphics for data from any 3-D mesh.

Run_AVS executes the commercial program AVS. (An AVS license is required.)

8.5.1.2 Process FEHMN

The *Process_FEHMN* button provides assistance in creating a control file and in executing, restarting and stopping FEHMN. The input file name will be the name used under *Pre-Process_ FEHMN*, or it can be changed with the *Name_Input_File* button under *Process_FEHMN*.

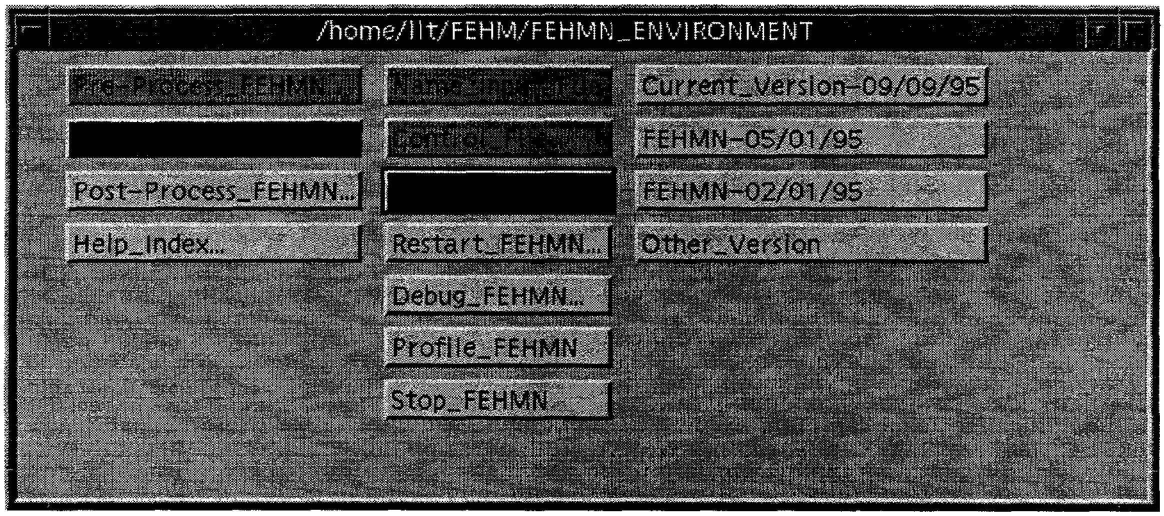


- Control Files

The *Control_File* button allows the user to create their own control file or have the Browser create the appropriate control file. The default control file name is *fehmn.files*. Another name can be entered with the *Name_Control_File* button. The *Edit_Control_File* button allows the user to directly edit the control file. The *Load_Default_Control_File* button will create a control file based on the name of the current input file. The *Read_GENMESH_grid* and *Read_Rectangular_grid* buttons modify (or creates if none exists) the control file to read the appropriate grid input file. The *Write_to_Output* button modifies (or creates if none exists) the control file to write the output to a file. The *Write_to_Terminal* button modifies (or creates if none exists) the control file to write the output to the terminal. If you do not want FEHMN to automatically run with a control file, fehmn.files can be deleted with the *Remove_Control_File* button.

- Executing, Restarting, Stopping FEHMN

Options for the most current version of FEHMN and older, frozen versions are available under the *Execute_FEHMN* button. Simply click on the version needed. If the control file *fehmn.files* is in your local space, the program will automatically begin. If another control file name is used, an interactive mode will start in an xterm window and the control file name can be entered at the first prompt. If no control file is used, the user will be prompted to input file names.



Options for the most current version of FEHMN and older, frozen versions are available to continue a run under the *Restart_FEHMN* button. When clicked on, a help page will pop up that describes how the macro **time** needs to be adjusted for a restart run. The control file will be changed for a restart run, or if there is no control file, one will be made. The **.fin* file will be copied to **.ini*.

Options for the most current version of FEHMN and older, frozen versions are also available for a debug mode and a profile mode, under the *Debug_FEHMN and Profile_FEHMN,* buttons, respectfully.

FEHMN can be stopped at any time by using the *Stop_FEHMN* button. This button will give you the process id and directions on how to kill the process in the xterm window provided.

8.5.1.3 Browser FEHMN Post-Processor

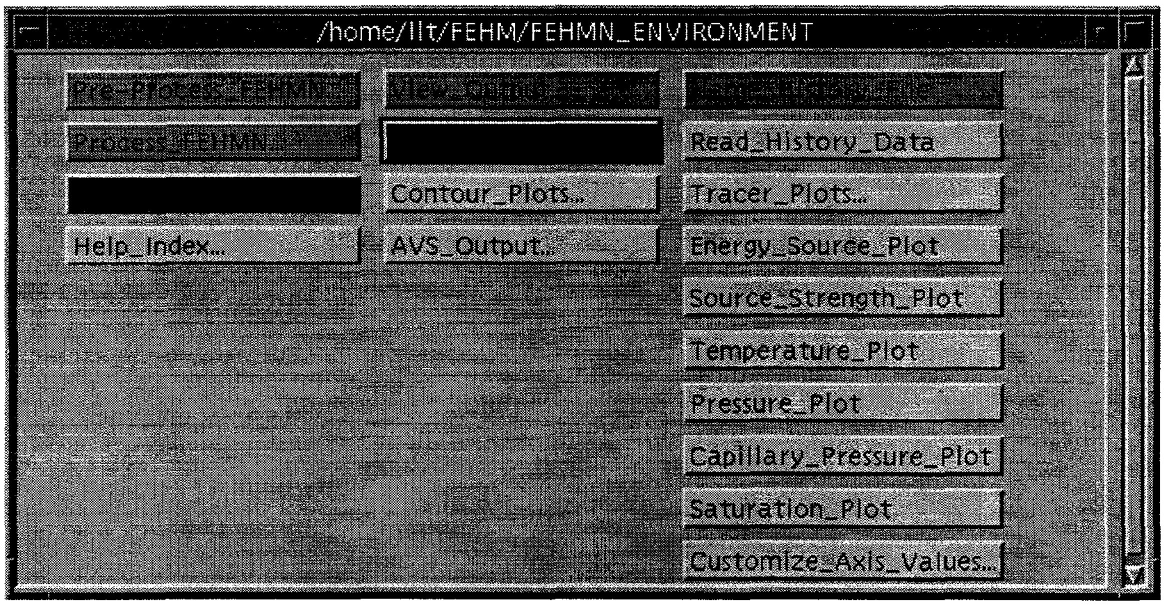
The Browser Post-processor allows the user to view FEHMN output files, make time history plots, make contour plots, and display the output with AVS (an AVS license is required).

- Viewing FEHMN Output Files

  The *View_Output* button will allow the user to view the FEHMN output files. The name used for the * (wildcard character) will be the prefix of the last input file name used, or the name entered in the *Name_Output_File* button.

- Time History Plots

  The *Time_History_Plots* button allows the user to make time history plots of the Energy Source, Source Strength, Temperature, Pressure, Capillary Pressure, and Saturation from the FEHMN **.his* and time history Tracer plots of the 10 species concentrations from FEHMN **.trc* output files. **.his* is specified on the seventh line of the control file and **.trc* on the eighth line (see Section 6.2.1 for more information on how to setup your control file to create these output files). A line for each node specified in the input is plotted, as time versus the value. The plots can be zoomed in, the window can be resized, and a log scale can be used for the x- and/or y-axis.



The title on the first line of the input deck is used for the title of the plot. The second title line displays the version of FEHMN and the date run. The x- and y-axis are automatically scaled. A legend is automatically displayed showing which line corresponds to which node.

The default time history file is *fehmn.his*. To change this, the user should click on the button *Name_History_File*. The data will automatically be converted to xrt plotting format when a new history file is named. To read in new data with the same file name (or if using the default), select the *Read_History_Data* button. This will convert the new data to the xrt plotting format.

Once the data is converted, plots can automatically be brought up by selecting the button for the plot desired. The *Tracer_Plots* button will bring up another column listing the 10 species. Only the species in the **.trc* file will be available for plotting. A message will be given if a species is selected that was not in the **.trc* file.

The values for the axis can be specified. Select the *Customize_Axis_Values* button, the *New_XY_Values* button, then the axis you want to specify. You can get the minimum and maximum values or return to having it automatically specified with the *Default_XY_Values* button. After setting the axis values, select the *Save_Axis_Values* button to save them.

Following are the controls available to manipulate the time history plots:

Zooming: Press control and hold down the left mouse button. Move the mouse to draw a box around the area to zoom into.

Scaling: Press control and hold down the middle mouse button. Move the mouse down to increase the graph's size. Move the mouse up to decrease the graph's size.

Reset: To reset the window, type r.

Resizing Window: The window size can be changed by clicking on a corner of the window and dragging the mouse to the desired size.

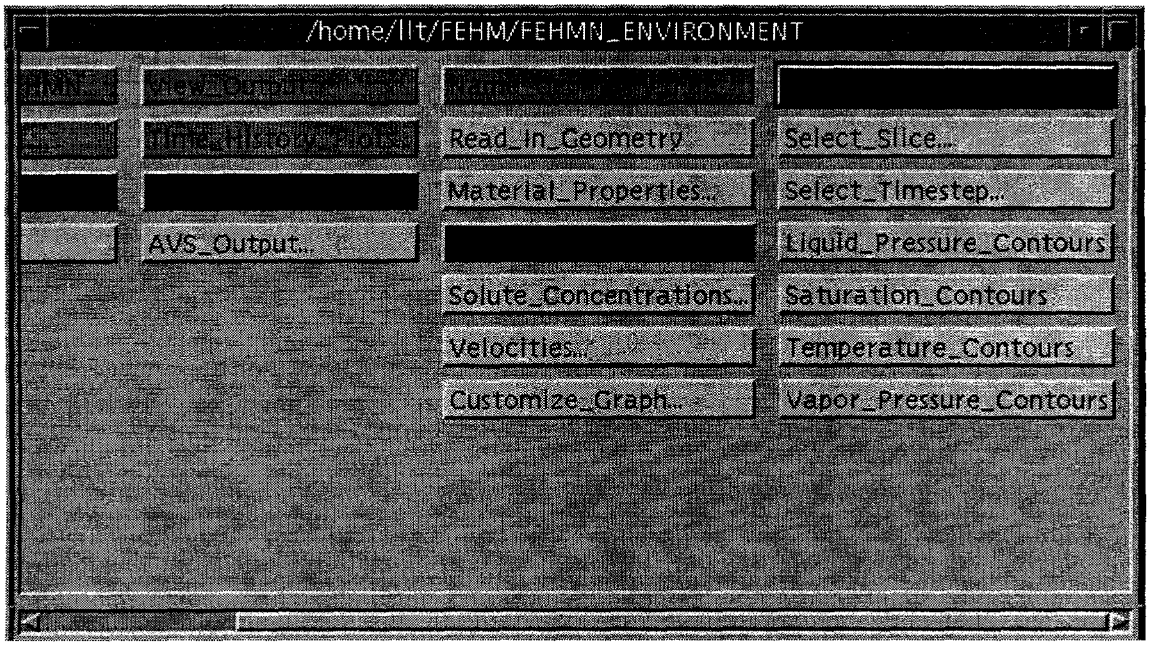
Log Scale: The x and/or y axis can be changed to a log scale by clicking on the toggle button on the top row.

Printing: Click on the *Print* button, on the top row. The default is set for the printer named *graphics*. This can be changed by editing the name. A postscript file or a XWD file can also be printed. Other options can be changed under the *Properties* button, along the bottom.

Exiting: Click on the *Exit* button, on the top row.

- Contour Plots

The *Contour_Plots* button allows the user to make x-, y-, or z-slices of material properties, temperature, saturation, pressure, velocities, and solute concentrations plots for 2-d quad and 3-d hex grids. The plots can be rotated freely; in the x-, y-, or z-direction; or from any end point. The plots can be zoomed, scaled, translated, and the window can be resized.



There are options for drawing the mesh, drawing shaded surfaces, drawing contour lines, and/or drawing zones.

The title on the first line of the input deck is used for the title of the plot. The version of the FEHMN executable and date run is the second line of the title. A title at the bottom of the plot includes, the type of plot, the slice taken, and at which time step if applicable. The x-, y-, and z-axis and the color bar are automatically scaled, unless specific values are specified. A legend is automatically displayed showing the numeric value of the colors.

The contour plots use data from FEHMN AVS output files, created by specifying the **cont** macro with **avs** option in the FEHMN input file (see Section 6.2.7). Slices of Material Property contour plots can be made by specifying **material** after **avs**. These include Permeability in the x-, y-, or z-direction; Thermal Conductivity in the x-, y-, or z- direction; Porosity; Rock Specific Heat; Capillary Pressure; and Models for Relative Permeability and Capillary Pressure.

Time dependent slices of Field Variables plots can be made for Liquid Pressure, Vapor Pressure, Saturation, and Temperature. A combination of **pressure** and **liquid** or **vapor**, **saturation**, **temperature** after **avs** in the **cont** macro as needed.

Time dependent slices of Solute Concentrations can be made by specifying **concentration** after **avs** in the **cont** macro. Buttons for the 10 species are provided.

Time dependent slice of Velocities can be made by specifying **velocity** and **liquid** or **vapor** after **avs** in the **cont** macro.

The speed (the square root of the x-, y-, and z-velocities) for liquid and vapor can be displayed.

The default AVS geometry file is *fehmn.10001_geo*. To change this, the user should click on the button *Name_Geometry_File*. The geometry will automatically be converted to xrt plotting format when a new geometry file is named. To read in new data with the same geometry file name (or if using the default), select *Read_In_Geometry*. This will convert the new data to the xrt plotting format. The same geometry can be used with multiple contour files.

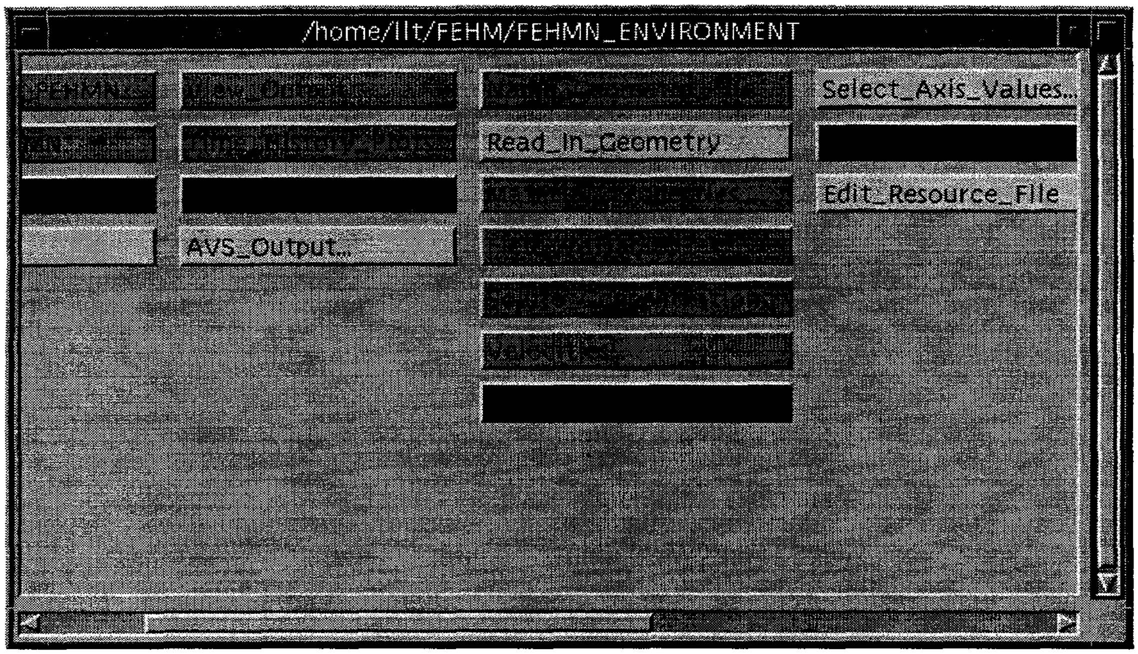
The default AVS contour files are *fehmn.1000**. To change this, the user should click on the button *Name_Contour_File*. The contour data will be read from the file selected.

Once the geometry is converted and a contour file named, plots can automatically be brought up. For Material Property plots with 3-d grids, the user must select the type of slice and the value for the slice. If not selected, the default will be a xy slice with a z value of 1.0. The geometry values (from the file **.10001_geo*) can be displayed by clicking on the button *Display_Geometry_File*. To display the plot, the user should click on the plot type. For 3-d grids, the slice type and value will remain the same for additional plot types, until changed.

For Field Variable, Solute Concentrations, and Velocity plots with 3-d grids, the user must select the type of slice and the value for the slice. If not selected, the default will be a xy slice with a z value of 1.0. For 2-d and 3-d grids, the user must also select the time step (the default is the second time step). The applicable time steps (from the file **.10001_avs_log*) can be viewed by clicking on the button *Display_Time_Steps*. The user can enter the time step desired with the *Input_Time_Step* button or the value of the day desired with the *Input_Number_Days* button. To display the plot, the user should click on the plot type. The slice type, value, and time step will remain the same for additional plot types, until changed.

The values for the axis can be specified. Select the *Customize_Graph* button, then the *Select_Axis_Values* button, and the axis you want to specify. You can get the minimum and maximum values or return to having it automatically specified.

The number of colors and the colors can be specified. Select the *Select_Color_Bar* button. The colors for the color bar can be selected from a range of colors with the *Select_Colors* buttons and the number of color divisions can be selected with the *Change_Number_Color_ Divisions* button.



Buttons are also available to *Use_Standard_Color_Bar*, *Use_Random_Colors*, *Use_Gray_Scale*, *Use_No_Color* (black & white), to select any color available on the system with the *Display_Available_Colors* button, or to return to having automatically generated color bars with the *Use_Auto_Color_Bar* button. The files containing the list of colors and the number of color divisions can also be directly edited with the *Edit_Color_List* button and the *Edit_Color_Divisions*. Colors are appended to the color list when the *Select_Color* buttons are selected. To pick up new colors, click on the *Edit_Color_List* button and delete the list contents of this file first.

With the *Edit_Resource_File* button, the graph and axis titles can be modified. The rotation of the graph can also be modified.

Following are the controls available to manipulate the contour plots:

Zooming: Press control and hold down the left mouse button. Move the mouse to draw a box around the area to zoom into.

Scaling: Press control and hold down the middle mouse button. Move the mouse down to increase the graph's size. Move the mouse up to decrease the graph's size.

Translation:Press shift and hold down the middle mouse button. Move mouse to shift the graph.

Reset: To reset the window (for zooming, scaling, and translation only), type r.

Rotating: Hold down the middle mouse button and either:

- move mouse counter-clockwise to rotate view clockwise or
- press x, y, z, or e to select an axis and then move mouse perpendicular to axis.

Resizing Window: The window size can be changed by clicking on a corner of the window and dragging the mouse to the desired size.

Printing: Click on the *Print* button, on the top row. The graph can be printed to any printer or a file. It can create postscript, XWD, or CGM files. Other options can be changed under the *Properties* button, along the bottom.

Exiting: Click on the *Exit* button, on the top row.

Various displays can be made using combinations of the four Toggles at the top of the display:

DrawMesh:When highlighted displays the X-Y grid projected onto the 3-D surface in a 3-D view with a Z-axis. The graph honors rotation and perspective control.

DrawShaded: When highlighted displays the data as a flat shaded surface in a 3-D view with a Z-axis. The graph honors rotation and perspective control.

DrawContours: When highlighted examines the distribution of the data and draws contour lines demarcating each of the distribution levels.

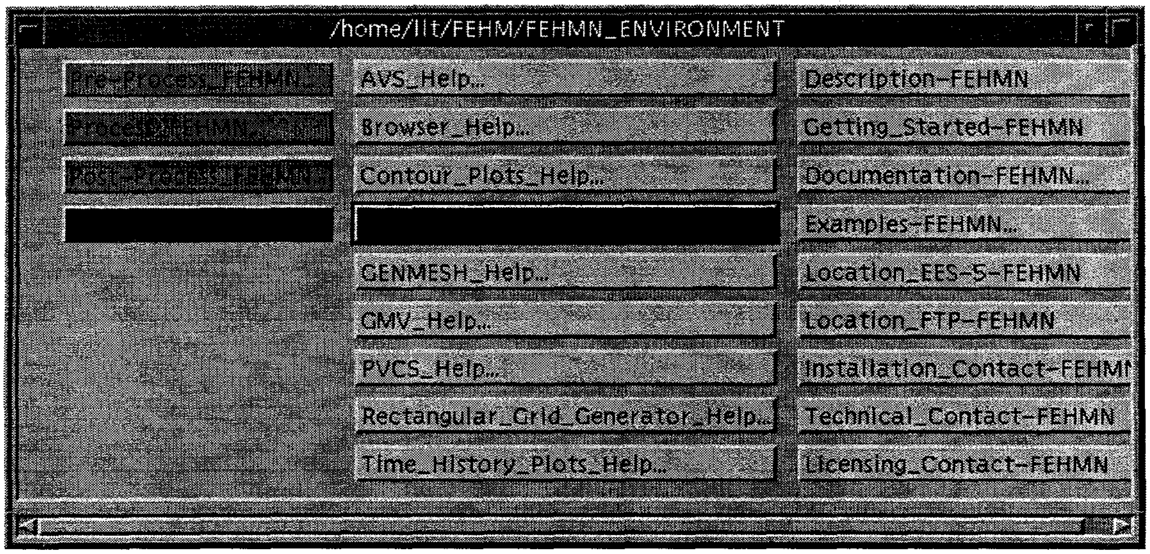
DrawZones:When highlighted examines the distribution of the data and fills each level with a solid color.

- Display AVS Output

The *Run_AVS* button will execute the commercial program, AVS (an AVS license is required). The name of the FEHMN AVS output file is entered within AVS.

8.5.1.4 Browser Help Index

The *Help_Index* button allows the user to view the FEHMN documentation and documentation for other codes used within the Browser.



8.6 Installation Instructions

On the EES-5 computer network, no installation is required to run FEHMN. The following Table lists the location and executable name for each platform.

Table VI. FEHMN Executable Locations

Platform	Path	Executable Name
SunOS	/pvcs.config/fehmn/fehmn_09_09/objects	xfehmn
Solaris	/pvcs.config/fehmn/fehmn_09_09/objects_sol	xfehmn
IBM	/pvcs.config/fehmn/fehmn_09_09/objects_ibm	xfehmn
HP	/pvcs.config/fehmn/fehmn_09_09/objects_hp	xfehmn
SGI	/pvcs.config/fehmn/fehmn_09_09/objects_sgi	xfehmn

FEHMN object code is provided for users modifying a few routines to link against, for installation. FEHMN source code is provided for users needing the entire source and for remote users.

Remote users can obtain FEHMN executable, objects, and/or source through a passworded ftp account after a software license agreement form has been completed.

8.6.1 Installation of FEHMN Using Objects

User's that need to modify a few routines, but do not need the entire source should only have the source they are changing in their local space. On the EES-5 Sun network, the following makefile should be used:

```
/pvcs.config/fehmn/ref/makefile
      -OR-
/pvcs.config/fehmn/ref/makefile-g (for debug)
```

FEHMN is installed by typing:

```
make
      -OR-
make -f makefile-g (for debug)
```

On the EES-5 IBM (magma), the following makefile should be used:

```
/pvcs.config/fehmn/ref/makefile_ibm
      -OR-
/pvcs.config/fehmn/ref/makefile_ibm-g
```

FEHMN is installed by typing:

```
make -f makefile_ibm
      -OR-
make -f makefile_ibm-g (for debug)
```

All makefiles create an executable called:

```
xfehmn
```

8.6.2 Complete Installation of FEHMN

User's that need the entire source and remote users can automatically setup the FEHMN directory structure. On the EES-5 Sun network, get the following tar files:

```
/pvcs.config/fehmn/fehmn_current/src_files.tar
/pvcs.config/fehmn/fehmn_current/objects_files.tar
```

and then type:

```
tar xvf src_files.tar
tar xvf objects_files.tar
```

The FEHMN directory structure should look like the following:

```
                   ┌── /airsj ──────── *.f
                   ├── /avs ────────── *.f, *.c
                   ├── /book_t ─────── *.f, *.c
                   ├── /cont_t ─────── *.f
                   ├── /dpdpj ──────── *.f
                   ├── /dualj ──────── *.f
                   ├── /equsj ──────── *.f
                   ├── /main_t ─────── *.f, *.c
                   ├── /mult_t ─────── *.f
         ┌── /src ─┼── /optsj ──────── *.f
         │         ├── /ptrk ───────── *.f
         │         ├── /read_t ─────── *.f
         │         ├── /solve_new ──── *.f
         │         ├── /solvj ──────── *.f
         │         ├── /strsj ──────── *.f
         │         ├── /ther_low ───── *.f
         │         ├── /therj ──────── *.f
~fehmn ──┤         ├── /time_sun ───── *.f
         │         ├── /tradj ──────── *.f
         │         ├── /user_con ───── *.f
         │         └── /welbj ──────── *.f
         │
         │                ┌── *.h           [include files]
         │                ├── *.dat         [example input files]
         └── /objects ────┼── makefile_*    [builds]
                          ├── *.o           [object files]
                          └── xfehmn        [executable]
```

Makefiles are included in these tar files and will be placed in your objects directory. There are makefiles for various platforms with options for debug, pvcs configuration builder commands, and purify. The format is:

makefile_*machine type*{_pvcs or nopvcs_}[_purify_][-g]

Machine_type can be cray, hp, ibm_long, sgi, or sun. The options inside {} are only available with the sun makefiles and are required. The items inside [] are optional. The purify option is to link in purify and is only available with the sun makefiles. The -g option is to link in debug and is available on all machines. Following are some examples:

```
makefile_ibm_long
makefile_hp-g
makefile_sun_nopvcs
makefile_sun_pvcs-g
makefile_sun_pvcs_purify-g
```

To compile and link FEHMN, select the appropriate makefile and type the following:

```
cd ~fehmn/objects
make -f makefile_sun_nopvcs  -OR-  make -f makefile_hp
```

All makefiles create an executable called:

```
xfehmn
```

8.6.3 FEHMN Licensing at Remote Sites

A software license agreement must be completed before receiving an FEHMN executable or source. To obtain access, get either the Commercial_agreement.ps or Government_agreement.ps form from anonymous ftp:

```
ftp ees5-ftp.lanl.gov
Name (): anonymous
Password:       [enter your email address]
cd pub/fehm
ftp> binary
ftp> get Commerical_agreement.ps
    or
ftp> get Government_agreement.ps
```

The form can be printed with any postscript printer. After reading, please sign the form and return to:

Lynn Trease
Mail Stop F665
P.O. Box 1663
Los Alamos National Laboratory
Los Alamos, NM 87545

FAX: 505-665-3687

9.0 EXAMPLES AND SAMPLE PROBLEMS

The following describes execution of the FEHMN code. Section 9.1 discusses the construction of an input file. Section 9.2 illustrates the entire procedure for executing the FEHMN code using terminal input. Example 1 describes the setup and results from a simple 2-D heat conduction simulation. The remaining sections provide more complex example problems and deal only with problem setup and expected results.

9.1 Constructing an Input File

FEHMN is a very general simulation code. Thus it is preferable to discuss the construction of an input file from a problem oriented point of view. In what follows the needs of the physical problem (initial conditions, boundary conditions, etc.) will be addressed in terms of the macro statements.

Initial conditions. These are needed for every problem, even if it is a steady state simulation. If the simulation is comprised of fully saturated water flow or heat conduction only, then the appropriate control statement would be **init** (page 40). The use of **init** also allows the specification of initial temperature and pressure (gravity) gradients. If two phase flow is prescribed (thermal or isothermal) then entering the initial conditions through the control statement **pres** (page 47) is more convenient. Initial values for noncondensible gas are handled in the **ngas** (page 43) control statement. It should be remembered that if a restart file is present, those values will have precedence over values input in control statement **init** but not over values input in control statement **pres**. Solute initial conditions are prescribed through the control statment **trac** (page 65).

Boundary conditions. Fluid and heat flow boundary conditions can be prescribed through control statements **pres**, **flow** (page 36), **hflx** (page 39), and **rflx** (page 53). Boundary conditions are entered with **pres** by specifying a negative phase state designation (the code will actually use the absolute value of the phase state designation). In this case the code will keep the variable values constant at whatever value was prescribed in **pres**. Flowing pressures are input with the **flow** control statement. Solute boundary conditions are prescribed through the control statment **trac**.

Material and Energy Balance Equations. The choice of the coupled system equations is made in control statements **sol** (page 63), **ngas**, and **air** (page 25).

Rock or Media Properties. These are found in the **rock** (page 57) and **perm** (page 46) control statements.

Fluid Properties. These are found in control statement **eos** (page 34), which is optional. If **eos** is not invoked, then the properties of water and air included in the code are used. Relative permeabilities, depending on both the fluid and media type, are found in control statement **rlp** (page 53).

Mesh Geometry and Nodal Coordinates. This geometry information is found in control statements **coor** (page 28) and **elem** (page 32). This information is usually created with a mesh generation program.

Simulation Time. The time stepping information including printout intervals and time step sizing is found in control statement **time** (page 64).

Numerics. Convergence criteria, upwinding parameters, fill-in for the preconditioned conjugate gradient solver and geometry type (2-D, 3-D, radial) are entered with control statement **ctrl** (page 29).

Advanced Iteration Control. Reduced degree of freedom methods are invoked with the **iter** (page 41) control statement. One important quantity entered with this statement is the maximum time for the job to run on the computer.

Sources and Sinks. These are input with the control statement **flow**. Care must be taken as the parameters have different meanings for different physical models.

The following table lists the input macros which should be used to formulate various types of problems.

Table VII. Required and Optional Macros by Problem Type

Problem Type	Required Macros	Optional Macros
Heat Conduction	title	comments
	cond	alti
	coor	cont
	ctrl	finv
	elem	flo2
	flow or hflx	flxo
	init or pres	iter
	node or nod2	renm
	rock	rflx
	sol	stea
	time	text
	stop	user
		vcon
		zone

Table VII. Required and Optional Macros by Problem Type (Continued)

Problem Type	Required Macros	Optional Macros
Water/Water Vapor/Heat Equivalent Continuum Dual Porosity * Dual Permeability **	title	comments
	coor	alti
	ctrl	cap
	elem	cond
	init or pres	cont
	node or nod2	eos
	perm	finv
	rlp	flow
	rock	flo2
	sol	hfxo
	time	hflx
	stop	iter
		ppor
	dual (* only)	pres
	dpdp (** only)	renm
		rflx
		rxn
		stea
		text
		trac
		vcon
		velo
		zone

Table VII. Required and Optional Macros by Problem Type (Continued)

Problem Type	Required Macros	Optional Macros
Air/Water/Water Vapor/Heat Equivalent Continuum Dual Porosity* Dual Permeability**	title	comments
	coor	adif
	ctrl	alti
	elem	cap
	flow or hflx	cond
	init or pres	cont
	ngas	eos
	node or nod2	finv
	perm	flow
	rlp	flo2
	rock	flxo
	sol	hflx
	time	iter
	stop	ppor
		renm
	dual (*only)	rflx
	dpdp (**only)	rxn
		stea
		text
		trac
		vcon
		velo
		zone
Gas/Water/NAPL/Heat		

Table VII. Required and Optional Macros by Problem Type (Continued)

Problem Type	Required Macros	Optional Macros
Air/Water/No Heat Equivalent Continuum Dual Porosity* Dual Permeability**	title	comments
	airwater	alti
	cond	cap
	coor	cond
	ctrl	cont
	elem	eos
	flow or hflx	finv
	init or pres	flow
	node or nod2	flo2
	perm	flxo
	rock	hflx
	sol	iter
	time	ppor
	stop	pres
		renm
	dual (*only)	rflx
	dpdp (**only)	rxn
		stea
		text
		trac
		vcon
		velo
		zone

9.2 Code Execution

To run FEHMN, the program executable file name is entered at the system prompt:

```
<PROMPT> xfehmn
```

The I/O file information is provided to the code from an input control file or the terminal. The default control file name is *fehmn.files*. If a control file with the default name is present in the directory from which the code is being executed, no terminal input is required. If the default control file is not present, input prompts are written to the screen. A short description of the I/O files used by FEHMN precedes the initial prompt. The following assumes the default control file was not found.

After the command **xfehmn** is given, the code queries the user regarding the input files, as follows:

```
Enter name for iocntl -- default file name: not using

[(name/na or not using), RETURN = DEFAULT]
```

This query asks for a control file name. For now, we assume a carriage return <cr> is entered and a control file is not being used. The following query will appear

```
Enter name for inpt -- default file name: fehmn.dat

[(name/na or not using), RETURN = DEFAULT]
```

This query asks for an input file name. If a <cr> is given, the default *fehmn.dat* is used for the input file. We shall assume that the input file name entered is

```
example/heat2d
```

Note that a subdirectory containing the file is also given. If the file did not exist, the code would repeat the prompt for an input file. Next the code would query to determine if the prefix of the input file name (the portion of the name preceding the final "." or first space) should be used for code generated file names.

```
Do you want all file names of the form example/heat2d.* ? [(y/n), RETURN = y]
*** Note: If "y" incoor and inzone will equal inpt ***
```

A <cr> will produce files with identical prefixes, including the subdirectory. If the response is negative, the code will query for the names of all required files. Assume we enter "n".

```
Enter name for incoor -- default file name: example/heat2d

[(name/na or not using), RETURN = DEFAULT]
```

(See Fig. 3 for the remaining file name queries.)

Next a query for terminal output appears.

```
tty output -- show all reference nodes, selected reference nodes, or none:
[(all/some/none), RETURN = none]
```

An "all" reply prints out the primary node information to the terminal at every time step. A "some" reply prints a selected subset of the node information. A reply of "none" suppresses all tty output with the exception of error messages printed if code execution is terminated abnormally or when maximum number of iterations are exceeded. Assume we enter "some".

The next query concerns the subroutine USER. This subroutine is used for special purposes and is not available to the general user.

```
user subroutine number (provided to subroutine USER before every time step):
[RETURN = none]
```

Assume a <cr> is entered.

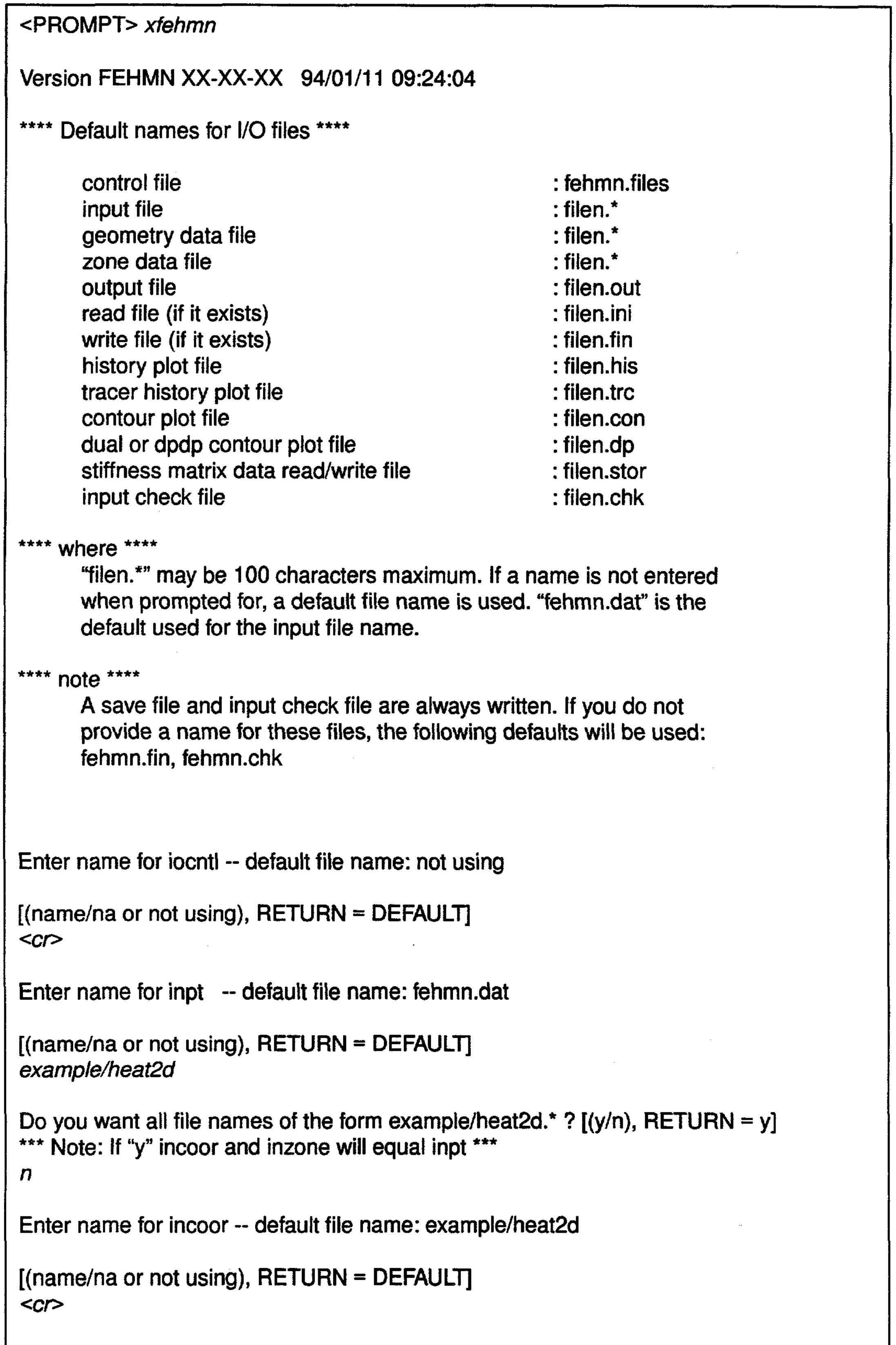
The code will then print a summary of the I/O files to be used.

The final query regards the acceptance of the file set just created. A “yes” reply denotes that the user has accepted the file set and the code proceeds with calculations. A “no” reply starts the query sequence again so I/O file names may be reentered or modified. A “stop” reply stops the current computer job.

```
If data is OK enter yes to continue, no to restart terminal input,
or stop to end program: [(yes/no/stop), RETURN = yes]
```

Screen output for this example execution using terminal input is shown in Fig. 3. User responses are shown in *italics*.

Returning to the first query regarding the *iocntl* file, if a control file name is entered no further terminal input is required. Figure 4 shows the control file that would produce the same results as the terminal responses illustrated in Fig. 3. Files that are not needed for output can be represented with a blank line. If names are not provided for the write file and/or the data check file, the code will use the following defaults: *fehmn.fin* and *fehmn.chk*. Following the file names is the flag that controls terminal output. The last line of the file is the user subroutine number. Omitting these values results in no terminal output and no user subroutine call.



```
<PROMPT> xfehmn

Version FEHMN XX-XX-XX   94/01/11 09:24:04

**** Default names for I/O files ****

      control file                                   : fehmn.files
      input file                                     : filen.*
      geometry data file                             : filen.*
      zone data file                                 : filen.*
      output file                                    : filen.out
      read file (if it exists)                       : filen.ini
      write file (if it exists)                      : filen.fin
      history plot file                              : filen.his
      tracer history plot file                       : filen.trc
      contour plot file                              : filen.con
      dual or dpdp contour plot file                 : filen.dp
      stiffness matrix data read/write file          : filen.stor
      input check file                               : filen.chk

**** where ****
      "filen.*" may be 100 characters maximum. If a name is not entered
      when prompted for, a default file name is used. "fehmn.dat" is the
      default used for the input file name.

**** note ****
      A save file and input check file are always written. If you do not
      provide a name for these files, the following defaults will be used:
      fehmn.fin, fehmn.chk



Enter name for iocntl -- default file name: not using

[(name/na or not using), RETURN = DEFAULT]
<cr>

Enter name for inpt   -- default file name: fehmn.dat

[(name/na or not using), RETURN = DEFAULT]
example/heat2d

Do you want all file names of the form example/heat2d.* ? [(y/n), RETURN = y]
*** Note: If "y" incoor and inzone will equal inpt ***
n

Enter name for incoor -- default file name: example/heat2d

[(name/na or not using), RETURN = DEFAULT]
<cr>
```

Figure 3.   Terminal query for FEHMN example run.

```
Enter name for inzone -- default file name: example/heat2d

[(name/na or not using), RETURN = DEFAULT]
<cr>

Enter name for iout  -- default file name: example/heat2d.out

[(name/na or not using), RETURN = DEFAULT]
<cr>

Enter name for iread  -- default file name: example/heat2d.ini

[(name/na or not using), RETURN = DEFAULT]
na

Enter name for isave  -- default file name: example/heat2d.fin

[(name/na or not using), RETURN = DEFAULT]
<cr>

Enter name for ishis  -- default file name: example/heat2d.his

[(name/na or not using), RETURN = DEFAULT]
<cr>

Enter name for istrc  -- default file name: example/heat2d.trc

[(name/na or not using), RETURN = DEFAULT]
na

Enter name for iscon  -- default file name: example/heat2d.con

[(name/na or not using), RETURN = DEFAULT]
na

Enter name for iscon1 -- default file name: example/heat2d.dp

[(name/na or not using), RETURN = DEFAULT]
na

Enter name for isstor -- default file name: example/heat2d.stor
[(name/na or not using), RETURN = DEFAULT]
na

Enter name for ischk  -- default file name: example/heat2d.chk

[(name/na or not using), RETURN = DEFAULT]
<cr>
```

Figure 3. Terminal query for FEHMN example run. (Continued)

```
tty output -- show all reference nodes, selected reference nodes, or none:
[(all/some/none), RETURN = none]
some

user subroutine number (provided to subroutine USER before every time step):
[RETURN = none]
<cr>

First reference output node will be written to tty

File purpose - Variable - Unit number - File name

        control        - iocntl   -  0  - not using

        input          - inpt     - 11  - example/heat2d

        geometry       - incoor   - 11  - example/heat2d

        zone           - inzone   - 11  - example/heat2d

        output         - iout     - 14  - example/heat2d.out

        initial state  - iread    -  0  - not using

        final state    - isave    - 16  - example/heat2d.fin

        time history   - ishis    - 17  - example/heat2d.his

        time his.(tr)  - istrc    - 18  - not using

        contour plot   - iscon    - 19  - not using

        con plot (dp)  - iscon1   - 20  - not using

        fe coef stor   - isstor   - 21  - not using

        input check    - ischk    - 22  - example/heat2d.chk

        Value provided to subroutine user: not using


If data is OK enter yes to continue, no to restart terminal input,
or stop to end program: [(yes/no/stop), RETURN = yes]
<cr>
```

Figure 3. Terminal query for FEHMN example run. (Continued)

```
heat2d
heat2d
heat2d
heat2d.out

heat2d.fin
heat2d.his




heat2d.chk
some
0
```

Figure 4. Input control file for heat conduction example.

9.3 Heat Conduction in a Square

This simple 2-D problem is used to illustrate input file construction and basic output. Heat conduction in a 1 meter square with an initial temperature, T_0 = 200 °C, is modeled after a surface temperature, T_s = 100 °C, is imposed at time, t = 0 (Fig. 5).The input parameters used for the heat conduction problem are defined in Table VIII.The finite element mesh for this problem is shown in Fig. 6. Only a quarter of the square needs to be modeled because of problem symmetry.

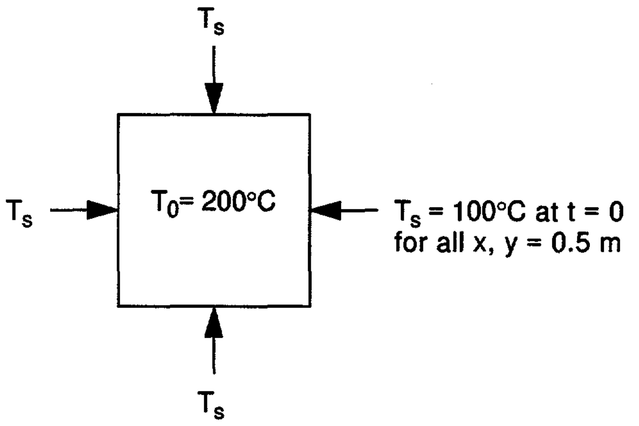


Figure 5. Schematic diagram of 2-D heat conduction problem.

Table VIII. Input Parameters for the 2-D Heat Conduction Problem

Parameter	Symbol	Value
Rock thermal conductivity	κ_r	$2.7 \frac{W}{m \cdot K}$
Rock density	ρ_r	2700 kg/m^3
Rock specific heat	C_r	$1000 \frac{J}{kg \cdot K}$
Width	a	0.5 m
Length	b	0.5 m
Initial temperature	T_0	200 °C
Surface temperature for all x, y = 0.5 m	T_s	100 °C
Rock thermal diffusivity	$\kappa = \frac{\kappa_r}{\rho_r C_r}$	

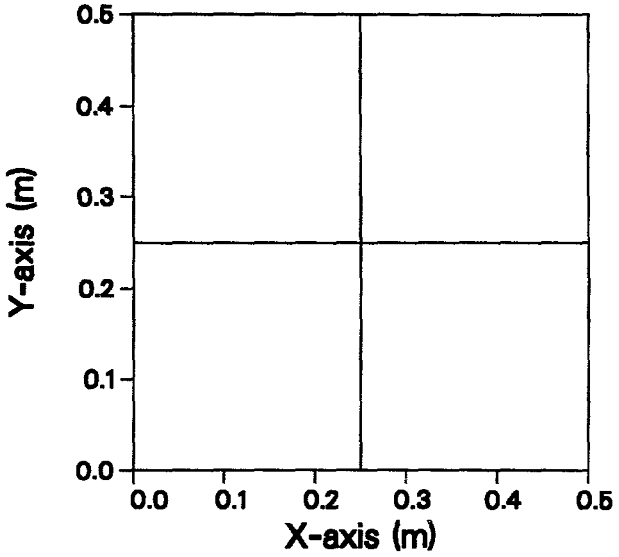


Figure 6. Finite element mesh used for 2-D heat conduction problem.

The input file (see Fig. 7) uses the required macro control statements **node** (output nodes), **sol** (solution specification - heat transfer only), **init** (initial value data), **rock** (rock properties), **cond** (thermal conductivities), **perm** (permeabilities), **time** (simulation timing data), **ctrl** (program control parameters), **coor** (node coordinates), **elem** (element node data), and **stop**. For this problem macro control statement **flow** is also used to set the temperature boundary conditions. A portion of the output file is reproduced in Fig. 8.

The analytical solution for 2-D heat conduction (Carslaw and Jaeger, 1959) is given by

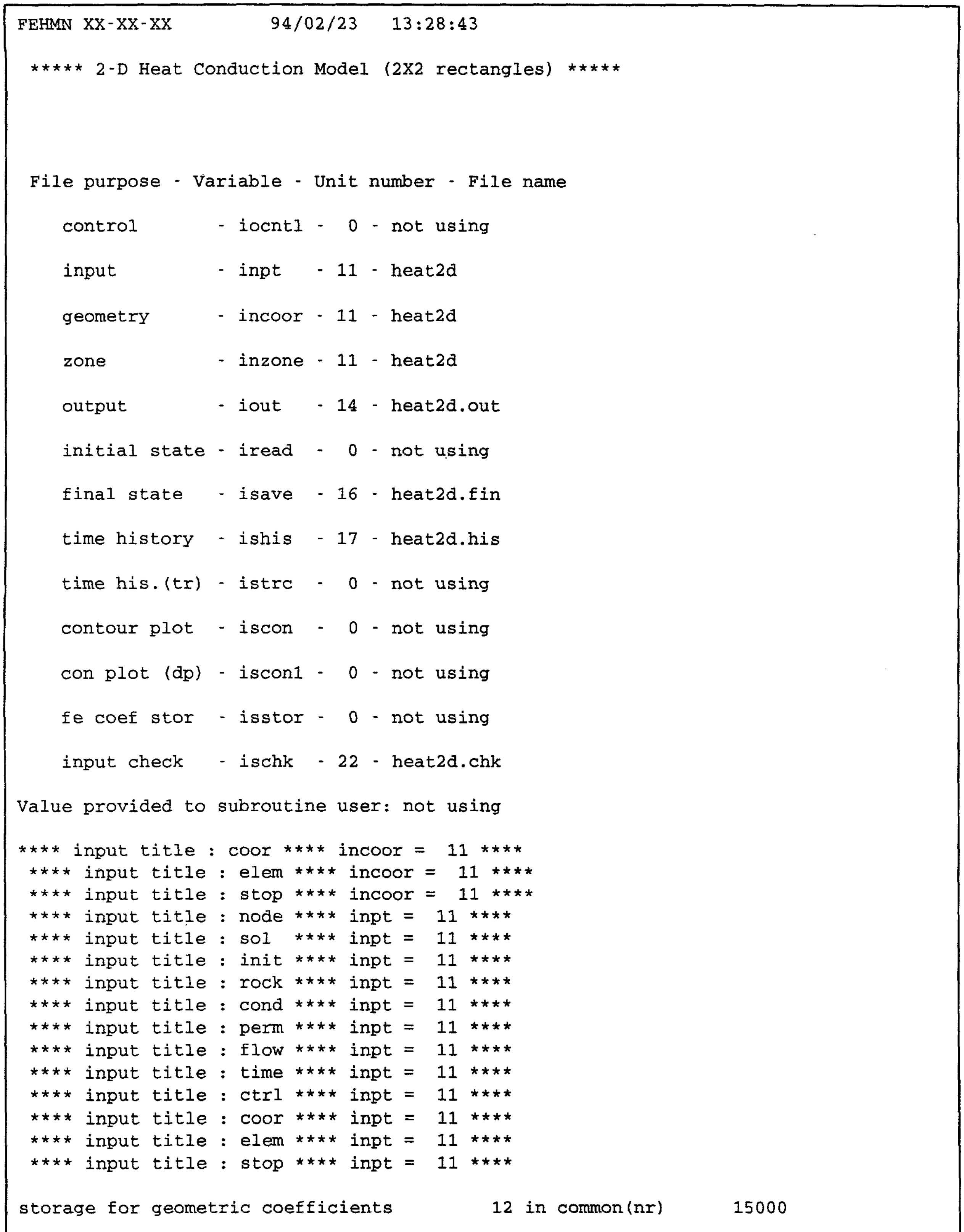
$$T = T_s + \frac{16(T_0 - T_s)}{\pi^2} \sum_{m=0}^{\infty} \sum_{n=o}^{\infty} \frac{(-1)^{m+n}}{(2m+1)(2n+1)} \cos\frac{(2m+1)\pi x}{2a} \cos\frac{(2n+1)\pi y}{2b} e^{-\alpha_{m,n} t}$$

where $\alpha_{m,n} = \frac{\kappa\pi^2}{4}\left[\frac{(2m+1)^2}{a^2} + \frac{(2n+1)^2}{b^2}\right]$ and the region is taken to be $-a < x < a, -b < y < b$.

Figure 9 shows a plot of the simulation results compared to the analytical solution for the selected output nodes.

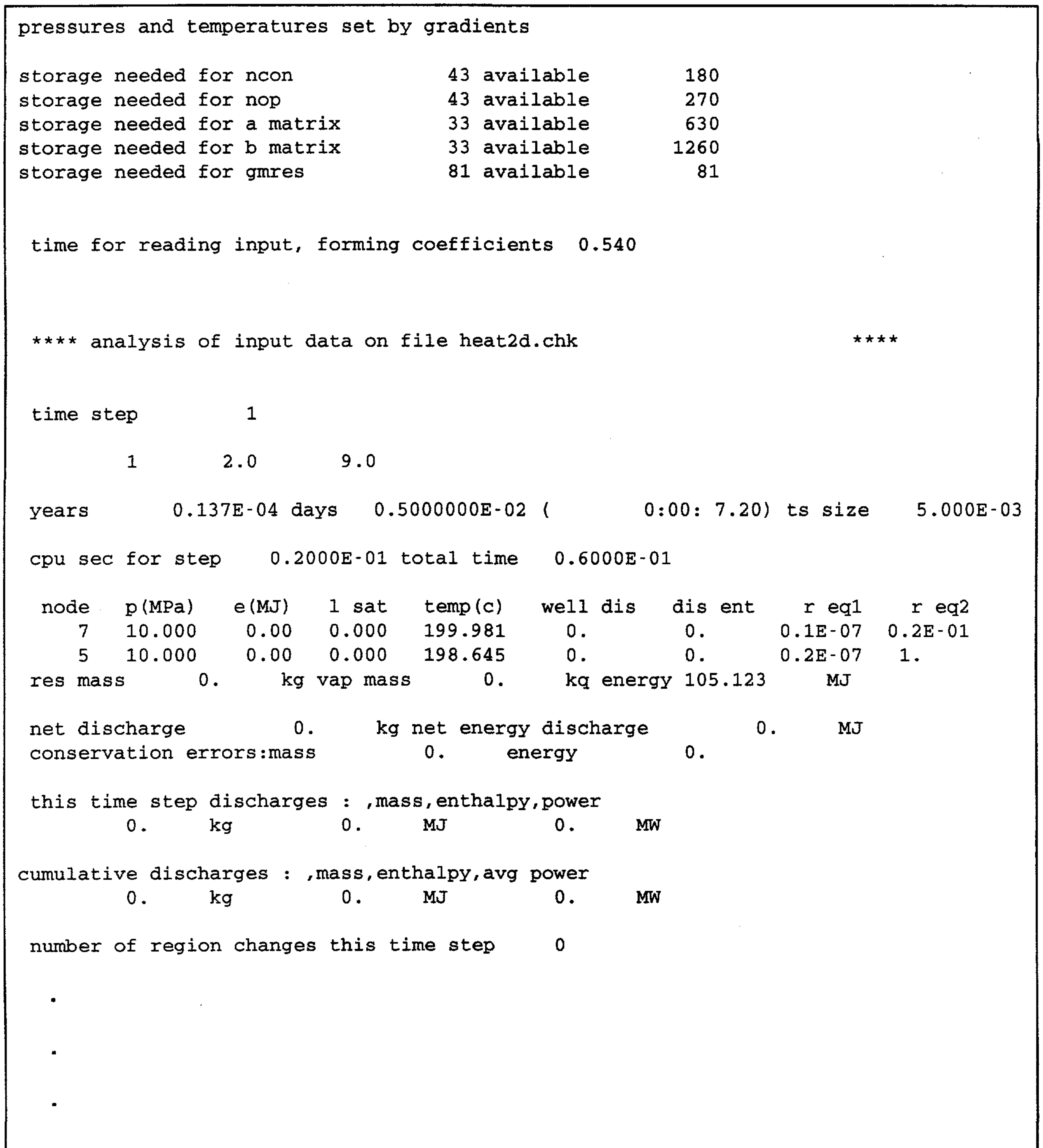
```
***** 2-D Heat Conduction Model (2X2 rectangles) *****
node
   2
   7   5
sol
  -1  -1
init
  10.    0.    200.      0.       0.    200.      0.       0.
rock
   1   9   1   2700.   1000.     0.

cond
   1   9   1     2.7e-00    2.7e-00    2.7e-00

perm
   1   9   1     1.e-30     1.e-30     1.e-30

flow
   1   3   1      10.00     -100.00    1.e03
   3   9   3      10.00     -100.00    1.e03

time
  0.005     4.00  1000      10    1994      02

ctrl
  40   1.e-04    08
   1   9   1   1

  1.0    0.0     1.0
  10     1.0     0.00005     0.005
   1   0
coor  Feb 23, 1994 11:39:40
   9
   1      0.      0.50    0.
   2      0.25    0.50    0.
   3      0.50    0.50    0.
   4      0.      0.25    0.
   5      0.25    0.25    0.
   6      0.50    0.25    0.
   7      0.      0.      0.
   8      0.25    0.      0.
   9      0.50    0.      0.

elem
   4   4
   1   4   5   2   1
   2   5   6   3   2
   3   7   8   5   4
   4   8   9   6   5

stop
```

Figure 7.   FEHMN input file for heat conduction example.



```
FEHMN XX-XX-XX         94/02/23   13:28:43

 ***** 2-D Heat Conduction Model (2X2 rectangles) *****



 File purpose - Variable - Unit number - File name

    control        - iocntl -  0 - not using

    input          - inpt   - 11 - heat2d

    geometry       - incoor - 11 - heat2d

    zone           - inzone - 11 - heat2d

    output         - iout   - 14 - heat2d.out

    initial state - iread   -  0 - not using

    final state    - isave  - 16 - heat2d.fin

    time history   - ishis  - 17 - heat2d.his

    time his.(tr) - istrc   -  0 - not using

    contour plot   - iscon  -  0 - not using

    con plot (dp) - iscon1 -  0 - not using

    fe coef stor   - isstor -  0 - not using

    input check    - ischk  - 22 - heat2d.chk

Value provided to subroutine user: not using

**** input title : coor **** incoor =  11 ****
 **** input title : elem **** incoor =  11 ****
 **** input title : stop **** incoor =  11 ****
 **** input title : node **** inpt =  11 ****
 **** input title : sol  **** inpt =  11 ****
 **** input title : init **** inpt =  11 ****
 **** input title : rock **** inpt =  11 ****
 **** input title : cond **** inpt =  11 ****
 **** input title : perm **** inpt =  11 ****
 **** input title : flow **** inpt =  11 ****
 **** input title : time **** inpt =  11 ****
 **** input title : ctrl **** inpt =  11 ****
 **** input title : coor **** inpt =  11 ****
 **** input title : elem **** inpt =  11 ****
 **** input title : stop **** inpt =  11 ****

storage for geometric coefficients         12 in common(nr)       15000
```

Figure 8. FEHMN output from the 2-D heat conduction example.



```
pressures and temperatures set by gradients

storage needed for ncon             43 available        180
storage needed for nop              43 available        270
storage needed for a matrix         33 available        630
storage needed for b matrix         33 available       1260
storage needed for gmres            81 available         81


 time for reading input, forming coefficients  0.540



 **** analysis of input data on file heat2d.chk                       ****


 time step         1

         1       2.0       9.0

 years       0.137E-04 days   0.5000000E-02 (        0:00: 7.20) ts size    5.000E-03

 cpu sec for step    0.2000E-01 total time   0.6000E-01

  node   p(MPa)    e(MJ)   l sat   temp(c)    well dis    dis ent     r eq1    r eq2
     7   10.000    0.00   0.000   199.981      0.          0.       0.1E-07  0.2E-01
     5   10.000    0.00   0.000   198.645      0.          0.       0.2E-07   1.
 res mass       0.      kg vap mass       0.      kq energy 105.123      MJ

 net discharge          0.      kg net energy discharge          0.      MJ
 conservation errors:mass          0.      energy          0.

 this time step discharges : ,mass,enthalpy,power
        0.     kg          0.     MJ          0.     MW

cumulative discharges : ,mass,enthalpy,avg power
        0.     kg          0.     MJ          0.     MW

 number of region changes this time step     0

   .

   .

   .
```

Figure 8.   FEHMN output from the 2-D heat conduction example. (Continued)

```
time step       801

        1       1.0       9.0

years       0.110E-01 days    4.000050     (        4:00: 0.07) ts size    5.000E-05

cpu sec for step    0.2000E-01 total time    30.85

 node   p(MPa)   e(MJ)   l sat   temp(c)   well dis   dis ent    r eq1    r eq2
    7   10.000    0.00   0.000   100.230     0.         0.       0.1E-12  0.2E-04
    5   10.000    0.00   0.000   100.115     0.         0.       0.6E-13  0.9E-05
res mass      0.      kg vap mass      0.      kq energy 67.5565     MJ

net discharge         0.      kg net energy discharge         0.      MJ
conservation errors:mass          0.      energy          0.

this time step discharges : ,mass,enthalpy,power
        0.      kg         0.      MJ         0.      MW

cumulative discharges : ,mass,enthalpy,avg power
        0.      kg         0.      MJ         0.      MW

number of region changes this time step     0


simulation ended: days    4.00     timesteps   801


total newton-raphson iterations =        801


total code time(timesteps) =     30.900000



****--------------------------------------------------------****
**** This program for                                       ****
****   Finite Element Heat and Mass Transfer in porous media ****
****--------------------------------------------------------****
****                      Version   : FEHMN XX-XX-XX        ****
****                      End Date : 94/02/23               ****
****                          Time : 13:35:48               ****
****--------------------------------------------------------****
```

Figure 8. FEHMN output from the 2-D heat conduction example. (Continued)

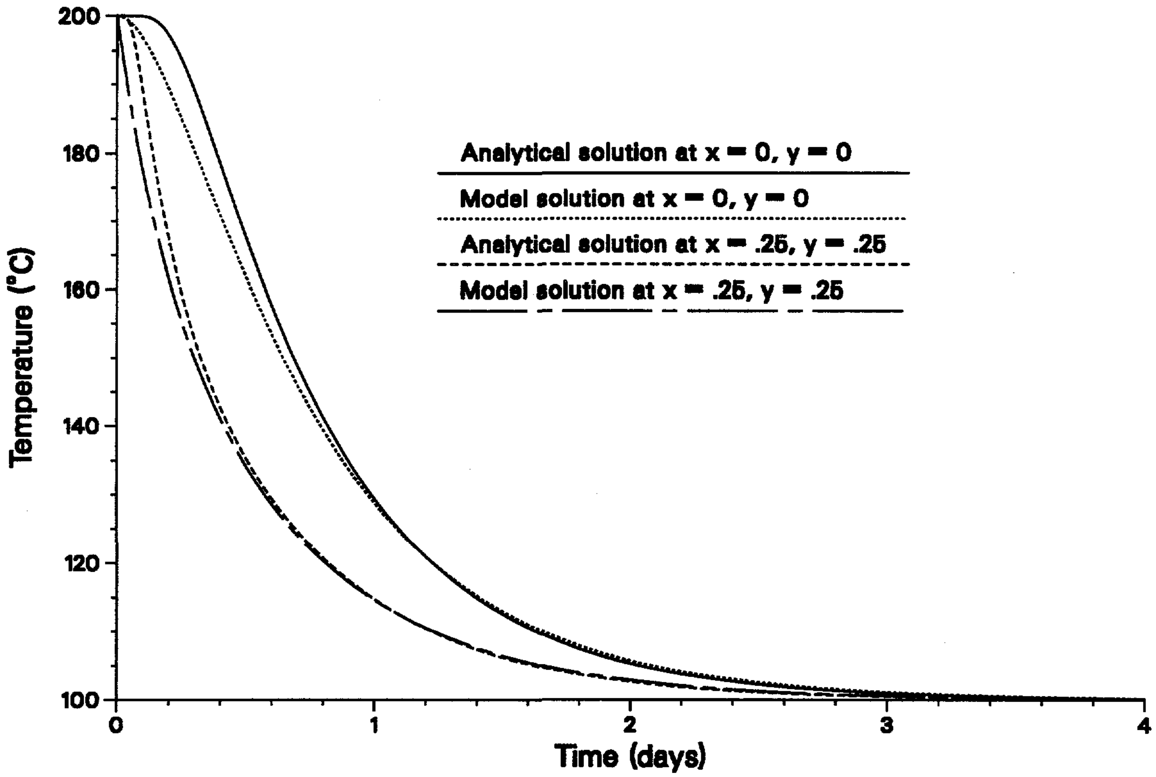


Figure 9. Comparison of analytical and model solution for 2-D heat conduction.

9.4 DOE Code Comparison Project, Problem 5, Case A

This problem involves multiphase flow in a 2-D horizontal reservoir. The problem is characterized by a moving two-phase region, i.e., the fluid produced at the production well is replaced by cold water recharge over one of the outer boundaries. The problem parameters are given in Table IX and the geometry and boundary conditions are shown in Fig. 10. Of particular note are the variable initial temperature field, provided to the code through a read file (see Section 5.6 on page 13), and the prescribed pressure and temperature onthe right boundary.A partial listing of the input file is provided in Fig. 11. In addition to the required macros, macro **flow** is used to specify the pressure and temperature boundary condition and the production flow rate. Macro **rlp** is used to set the residual liquid and gas saturations.

Table IX. Input Parameters for the DOE Code Comparison Project, Problem 5, Case A

Parameter	Symbol	Value
Reservoir permeability	k	$2.5 \times 10^{-14}\ \mathrm{m}^2$
Reservoir porosity	f	0.35
Rock thermal conductivity	κ_r	$1\ \frac{\mathrm{W}}{\mathrm{m} \cdot \mathrm{K}}$
Rock density	ρ_r	$2563\ \mathrm{kg/m}^3$
Rock specific heat	C_r	$1010\ \frac{\mathrm{J}}{\mathrm{kg} \cdot \mathrm{K}}$
Reservoir length	x	300 m
Reservoir thickness	y	200 m
Liquid residual saturation	s_{lr}	0.3
Gas residual saturation	s_{gr}	0.1
Reservoir discharge	q_m	$0.05\ \frac{\mathrm{kg}}{\mathrm{m} \cdot \mathrm{s}}$
Initial Pressure	P_o	3.6 MPa
Production well coordinates:	$x = 62.5$ m,	$y = 62.5$ m
Observation well coordinates:	$x = 162.5$ m,	$y = 137.5$ m

Initial temperature distribution (T in °C, r in m):

$$T(x, y, 0) = \left\{ \begin{array}{c|c} 240 & 0 \le r \\ 240 - 160\left(\frac{r-100}{200}\right)^2 + 80\left(\frac{r-100}{200}\right)^4 & 100 < r < 300 \\ 160 & r \ge 300 \end{array} \right.$$

where $r = \sqrt{x^2 + y^2}$

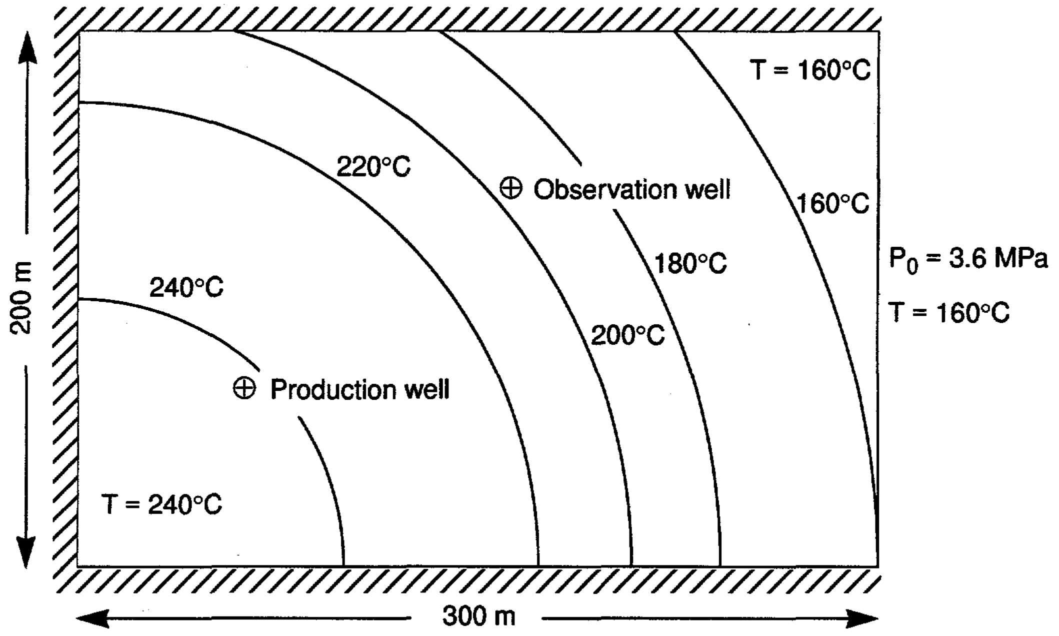


Figure 10. Schematic diagram of the geometry and boundary conditions for the DOE code comparison project problem.

There is no analytical solution for this problem, but six researchers produced results for the DOE code comparison project (Molloy, 1980). The reader is referred to this reference for a more detailed discussion of this problem and the code comparison. Results from this problem are compared to those for the other codes, obtained from Molloy (1980), as a check on FEHMN. The results for the outlet temperature, shown in Fig. 12, are in excellent agreement with the other codes. The results for the outlet pressure, Fig. 13, and pressure at an observation well 125 m distant, Fig. 14, are also in good agreement with the other codes. A contour plot of temperature at the end of the simulation was also generated for this problem and is shown in Fig. 15.

```
*** DOE Code Comparison Project, Problem 5, Case A ***
node
   2
   50  88
sol
   1   1
init
   3.6    0.    240.      0.       0.    240.      0.      0.
rlp
   2      0.3     0.1     0.0      0.0

   1     140      1       1

rock
   1     140      1    2563.       1010.         0.35

cond
   1     140      1       1.00e-00   1.00e-00   1.00e-00

perm
   1     140      1       2.5e-14    2.5e-14    0.e-00

flow
  88      88      1       0.050      -25.00      0.
  14     140     14       3.600    -160.00       1.

time
   30.0  3650.   10000   1000   1994      03

ctrl
   40   1.e-07    08
   1     140      1       1

   1.0    0.0     1.0
   40     1.2     0.1          60.
   1      0
coor
   140
   .
   .
   .
elem
   4    117
   .
   .
   .
stop
```

Figure 11. FEHMN input file for DOE problem.

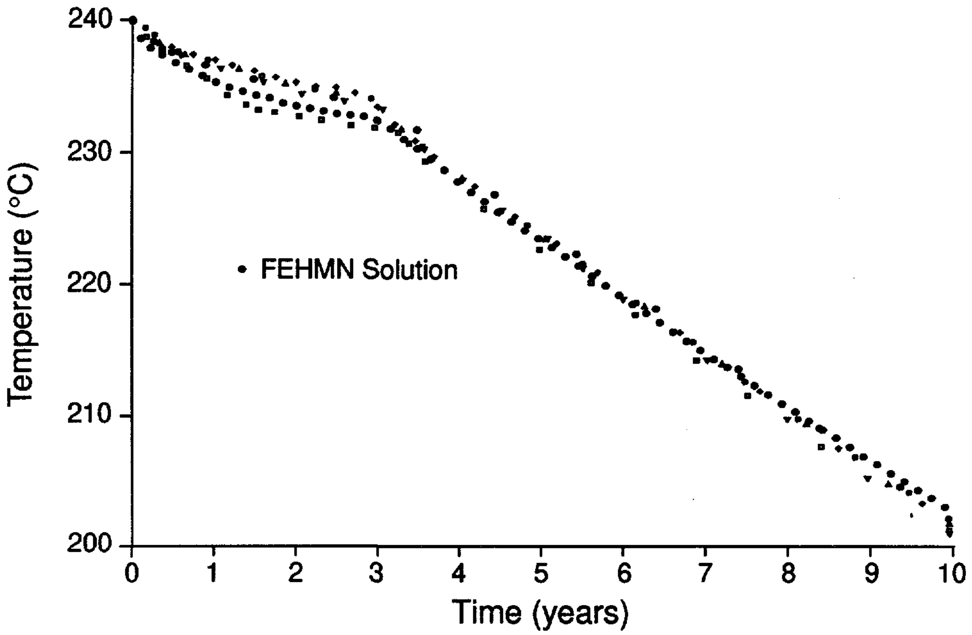


Figure 12. FEHMN production well temperatures compared with results from other codes.

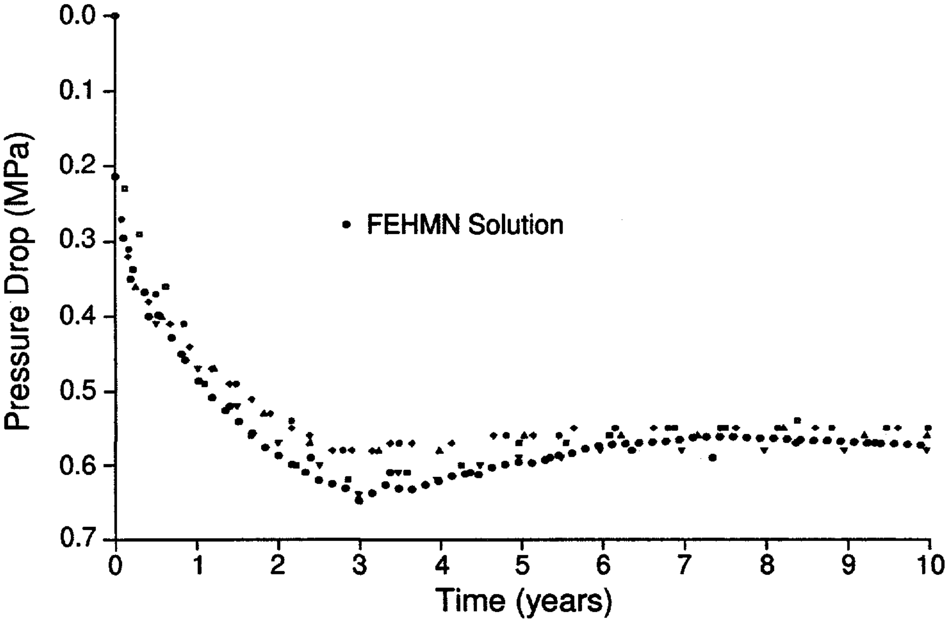


Figure 13. FEHMN production well pressures compared with results from other codes.

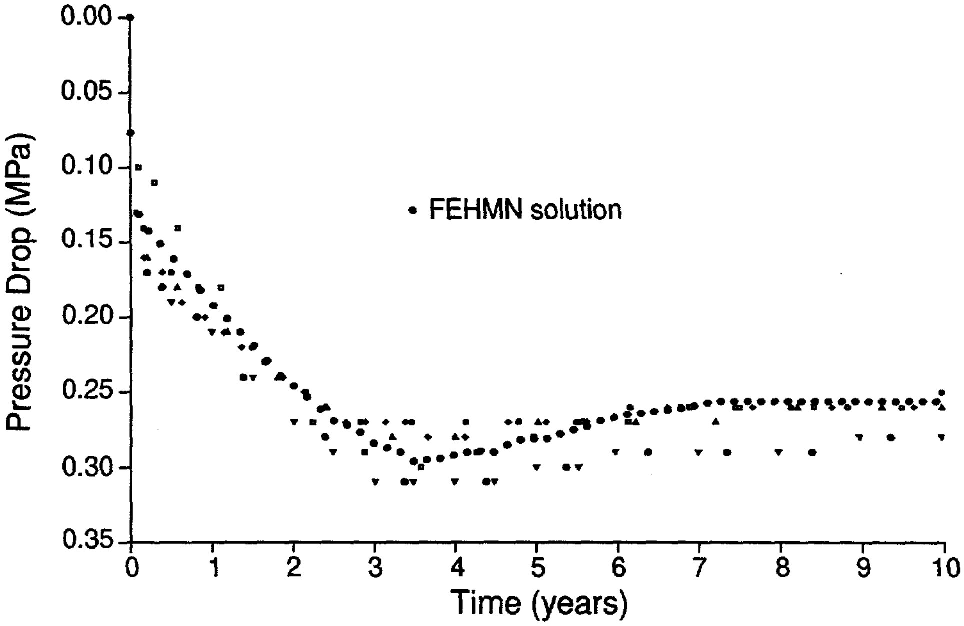


Figure 14. FEHMN observation well pressures compared with results from other codes.

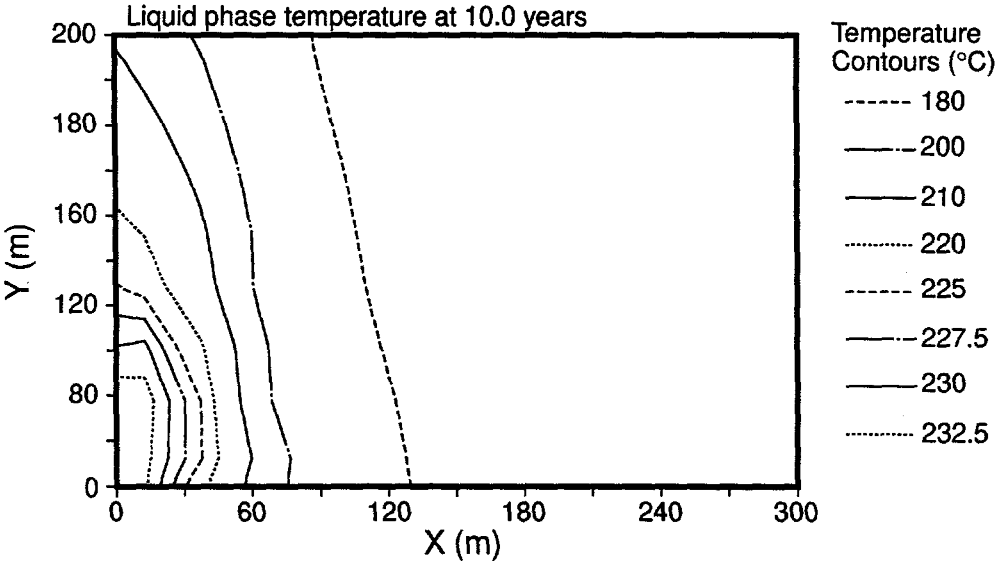


Figure 15. Contour plot of temperature for the DOE problem.

9.5 Reactive Transport Example

This one-dimensional example demonstrates the use of the reactive transport module of FEHMN. The input for the **trac** and **rxn** macros are those discussed in the example input for these macros in Section 6.2.39 and Section 6.2.48. The flow system is a one-dimensional flow path of 402 nodes (201 x 2) with rock properties and flow rates such that the mean fluid residence time in the path is 10,000 yr. Species *A* sorbs with a K_d of 0.1, which, for the rock properties chosen, is equivalent to a retardation factor of 5. The solute transport problem is run for 5,000 yr, or half of the mean residence time of the fluid. Therefore, in the absence of other reactions, species *A* would be expected to travel 0.5 / 5 = 0.1 of the length of the column.

When chemical reactions are included, the situation becomes more complex. Fig. 17 shows the concentration profiles at the end of the simulation for this example (called Run 1). Even though the first reaction is specified as kinetically controlled, the rate constants are large enough that for the reaction to virtually reach equilibrium over the time period of the simulation ($k_r\tau = 50$, where τ is the time of the simulation). Thus, the concentrations of *A* and *B* essentially reach equilibrium; the equilibrium constant is the ratio of the rate constants, or 0.1 (there is 10 times as much *A* as *B* in solution). Of course, *A* is also present on the rock surface wherever concentrations are non-zero. Solute *C* travels with *A* and *B* and is in equilibrium with *B* in solution; its concentration is 0.2 times that of *B* everywhere because of the equilibrium constant chosen. The entire suite of solutes has moved roughly 10% of the way down the column, as discussed above. With chemical reactions, the rate of movement of the solutes can be faster than that of a non-reacting species that only sorbs, because solutes *B* and *C* do not sorb. However, for this example solute *A* constitutes about 90% of the aqueous portion of

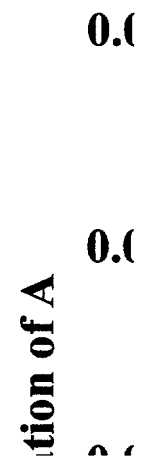


Figure 16. Concentration Profiles for Run 1.

the contaminant. Kinetics and equilibrium parameters that favor the formation of *B* and *C* would in turn increase the rate of movement of the contaminants.

A common application is radionuclide migration, for which we must include radioactive decay of each solute. It may also be of interest to track the movement of the daughter product. In Run 2, we assume that all species of Run 1 are species of the same radionuclide, each of which undergoes decay to a long-lived, mobile radionuclide that does not sorb. The input file is changed to include a fourth solute in the **trac** macro that is identical in input to the second and third solutes. The changes to the **rxn** macro are more complex, and thus are shown in Fig. 17.

```
rxn
group
2
1            1            1          0
0            0            0          1
5            0
kinetic      3.1688e-11   0.         3.1688e-10  0.
equilibrium  1            0.2        0.          1.e-2      1.e-3      1.e-10
kinetic      3.1688e-12   0.         0.          0.
kinetic      3.1688e-12   0.         0.          0.
kinetic      3.1688e-12   0.         0.          0.
1           -1            0          0
0            1           -1          0
1            0            0         -1
0            1            0         -1
0            0            1         -1
1           -1            0          0
0            1           -1          0
1            0            0         -1
0            1            0         -1
0            0            1         -1
1            1            1          1
1            1            1          1
1            1            1          1
1            1            1          1
1            1            1          1
0            0            0          0
0            0            0          0
1            1            1          1
0            0            0          0
0            0            0          0
```

Figure 17. FEHMN macro **rxn** in the input file for reactive transport example.

Radioactive decay is handled by adding three reactions, one each for *A*, *B*, and *C* reacting to form D. The kinetics of each of these reactions are identical, reflecting an irreversible, first order decay to form the daughter product. Solute *A* sorbs to the rock surface: therefore, we must specify that the decay reaction takes place for both the aqueous and solid portions of the solute. This is done by setting FL_MULT and SB_MULT to 1 for the third reaction.

Figure 18 shows the results of this example. A significant amount of daughter product D has been produced, and it is not forced to travel at the reduced velocity of sorbing solute *A* because it is connected to the other solutes only through the radioactive decay source term. The original solutes behave as they did in Run 1, except that their concentrations are somewhat reduced due to radioactive decay.

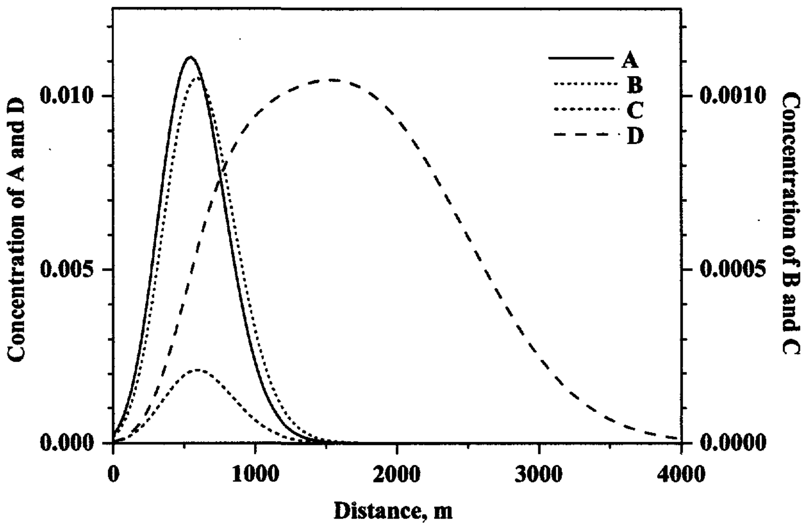


Figure 18. Concentration profiles for Run 2.

Considerations of Numerical Efficiency. The decision of how to group the solutes represents a trade-off among the robustness required for a given reactive transport system, memory requirements, and computational speed. We use these example runs to illustrate some of the considerations. In the discussions below, when we speak of the coupling of solutes, we refer to the method of grouping the solutes into systems of equations that are solved simultaneously. Regardless of the grouping of and order which the solute concentrations are solved, the code requires that the full system of interacting solutes reach convergence at every time step. The overall solution is therefore "fully coupled", regardless of the details of the solution procedure.

In Run 1, we solved first for solute *A* alone, after which solutes *B* and *C* were coupled. Solutes that are coupled only through a kinetic reaction need not be solved simultaneously as long as the kinetics are not too rapid. When systems are solved as more than one group, the time required to complete one outer iteration is shorter, but more outer iterations will be required (only one outer iteration is required if all solutes are coupled into a single group). A rule of thumb is that as long as only a few outer iterations are required, solving the problem in several groups will be competitive with a more fully coupled solution. For example, in Run 1, two or three outer iterations were typically required. Coupling all three solutes reduces the number of outer iterations to 1, but the computational time was virtually identical in this example problem. Note that solutes coupled through equilibrium reactions must always be solved simultaneously, since internal to the code, these reactions are specified with very rapid kinetics to approximate equilibrium behavior. An added benefit of solving the system as several groups is that the memory requirements are lower. This factor was not a consideration for this example, which solved a system of only 402 nodes.

In Run 2, notice that the fourth solute was solved alone after a group that coupled the first three. There is no benefit to the convergence of the system of equations from simultaneously solving solute D with the other three because it is formed only from irreversible reactions (radioactive decay) involving solutes A, B, and C. This means that there is no “feed-back” from the concentration of D onto the other concentrations. The decay reactions provide the source term for solute D, but the concentration of D does not impact the solution of the other solutes; the coupling is one-way. The only potential benefit is that when all solutes are coupled, the system is automatically solved in a single outer iteration, whereas the code cannot assume overall convergence and must perform a second outer iterations when D is decoupled from the other three solutes. For this problem, the benefit of fully coupling the solution is almost exactly counterbalanced by the additional work of solving a four degree-of-freedom solution (versus three followed by one), so that the two solutions take comparable time to finish.

10.0 USER SUPPORT

Licensing and installation support can be received from:

Lynn Trease
llt@vega.lanl.gov
505-667-0140

Technical support can be received from:

George Zyvoloski
gaz@vega.lanl.gov
505-667-1581

Bruce Robinson
robinson@vega.lanl.gov
505-667-1910