LBNL-1005134

USER'S MANUAL OF THE MESHMAKER V1.5 CODE: A MESH GENERATOR FOR DOMAIN DISCRETIZATION IN SIMULATIONS OF THE TOUGH+ AND TOUGH2 FAMILIES OF CODES

George J. Moridis

Energy Geosciences Division
Lawrence Berkeley National Laboratory
Berkeley, CA 94720

February 2016

This work was supported by the Assistant Secretary for Fossil Energy, Office of Natural Gas and Petroleum Technology, through the National Energy Technology Laboratory, under the U.S. Department of Energy, Contract No DE-AC02-05CH11231.

PAGE LEFT INTENTIONALLY BLANK

User's Manual of the MESHMAKER V1.5 Code: A Mesh Generator For Domain Discretization In Simulations of the TOUGH+ and TOUGH2 Families Of Codes

George J. Moridis
Earth Sciences Division, Lawrence Berkeley National Laboratory
University of California, Berkeley, California

Abstract

MESHMAKER V1.5 is a code that describes the system geometry and discretizes the domain in problems of flow and transport through porous and fractured media that are simulated using the TOUGH+ [*Moridis and Pruess*, 2014] or TOUGH2 [*Pruess et al.*, 1999; 2012] families of codes. It is a significantly modified and drastically enhanced version of an earlier simpler facility that was embedded in the TOUGH2 codes [*Pruess et al.*, 1999; 2012], from which it could not be separated. The code (**`MeshMaker.f90`**) is a stand-alone product written in FORTRAN 95/2003, and can be run on any computational platform (workstations, PC, Macintosh). Its architecture follows the tenets of Object-Oriented Programming, has a modular structure and can perform a number of mesh generation and processing operations, including the creation of heterogeneous subdomains and boundaries within the simulated domains. It can generate two-dimensional radially symmetric (*r,z*) meshes, and one, two-, and three-dimensional rectilinear (Cartesian) grids in (*x,y,z*). The code generates the file **`MESH`**, which includes all the elements and connections that describe the discretized simulation domain and conforming to the requirements of the TOUGH+ and TOUGH2 codes. Multiple-porosity processing for simulation of flow in naturally fractured reservoirs can be invoked by means of a keyword `MINC`, which stands for *Multiple INteracting Continua*. The MINC process operates on the data of the primary (porous medium) mesh as provided on disk file **`MESH`**, and generates a secondary mesh containing fracture and matrix elements with identical data formats on file **`MINC`**.

PAGE LEFT INTENTIONALLY BLANK

TABLE OF CONTENTS

PAGE LEFT INTENTIONALLY BLANK

LIST OF FIGURES

PAGE LEFT INTENTIONALLY BLANK

1.0. Introduction

1.1. Background

The TOUGH family of codes [*Pruess et al*., 1999; 2012] and its most recent TOUGH+ [*Moridis et al.*, 2008; *Moridis and Pruess*, 2014] successor generation of codes [*Pruess et al*., 1999; 2012] were developed for the simulation of multi-component, multiphase fluid and heat flow developed at the Lawrence Berkeley National Laboratory (LBNL). These codes are based on the Integral Finite Difference space discretization method [*Narasimhan and Witherspoon*, 1976], a finite-volume method. The corresponding meshes that are needed for such simulations are quite different from the ones created by standard gridding software because they do not use global coordinates, but only local coordinates that describe the relative positions of the centers of the gridblock elements. This precludes the use of common gridding applications and necessitates specialized software.

Pruess et al. [1999; 2012] provided a simple mesh-making facility named MESHMAKER (hereafter referred to as MM1) for use with the TOUGH family of codes.

The MM1 facility is not a stand-alone code, but it is completely integrated within the TOUGH2 code [*Pruess et al*., 1999; 2012]. While useful in creating automatically TOUGH meshes, MM1 has some serious shortcomings. It is applicable to homogeneous domains involving a single medium (rock type), and is limited to the creation of Cartesian or cylindrical meshes of regular geometry. MM1 is limited to 5-character element names and its numbering system is inflexible (proceeding invariably along the X, Y and Z directions, always in that order), thus often preventing the computational and memory savings that could be attained during the matrix solving process by a reduction in the matrix bandwidth (a direct result of a more efficient element numbering). Additionally, the MM1 grids cannot provide the necessary information needed for the description of the more complex boundary specifications that are available options in the TOUGH+ family of codes [*Moridis et al*., 2008; *Moridis and Pruess*, 2014]. For the solution of problems involving heterogeneous systems of complex geometry, other gridding software–e.g., WinGridder [*Pan*, 2008]–have to be used, or the MM1-created grids have to be modified either manually or by using mission-specific software.

1.2. The MESHMAKER V1.5 Application

The MESHMAKER V1.5 application is not a simple revision of the older MM1 facility, but a new stand-alone software that operates independently of the TOUGH+ or TOUGH2 codes. It was developed for the discretization of domains involved in simulations of all members of the TOUGH family of codes, provides significant capabilities in addition to those of MM1, several shortcomings of which it addresses.

MESHMAKER V1.5 can describe large (in terms of number of elements, theoretically in excess of 10^9 elements) systems with heterogeneous subdomains of arbitrary geometry and complex boundary conditions. It is written in standard FORTRAN 95/2003, and can be run on any computational platform (workstations, PC, Macintosh). It has a modular architecture that follows the tennets of Object-Oriented programming. Unlike MM1, the very large grids it creates have elements with 5- or 8-character long names, with any number of heterogeneous subdomains of complex geometries, and with any number of boundaries, time-invariable or transient.

MESHMAKER V1.5 generates grids that are directly usable by all members of the TOUGH+ and of the conventional TOUGH2 families of codes for no-flow (Neuman-type) boundary systems. The complex boundary subdomains that it can create (usually involving constant conditions, i.e., of the Dirichlet type) are seamlessly used by the TOUGH+ codes, but may require some editing and manipulation for TOUGH2 simulations. The element numbering in the meshes created by MESHMAKER V1.5 begins on the axis with the smallest number of subdivisions and progresses along axes with increasing numbers of subdivisons, thus ensuring minimum bandwidth of the matrices to be solved in TOUGH+/TOUGH2 simulations and significant computational and memory usage efficiencies. It permits up to 1000 elements with the same non-numerical part of their names (see *Moridis and Pruess* [2014]) at the same elevations, thus significantly simplifying initialization and gravity equilibration operations in large 2D and 3D systems.

`MeshMaker.f95`, the FORTRAN90/2003 code for the MESHMAKER V1.5 application, is distributed as part of the TOUGH+ v1.5 [*Moridis and Pruess*, 2014] core code, the User's Manual of which incorporates a complete documentation of the gridding

facility. Because of the potential usefulness of MESHMAKER V1.5 application to users of the standard TOUGH2 family of codes, `MeshMaker.f90` is also distributed by LBNL as an independent separate entity, hence the need for this User's Manual that includes a full description of the code, its input requirements and capabilities, as well as examples of input files. Appropriate illustrative examples with sample input input files (and the corresponding outputs) are also included in the electronic directory that is distributed with the code.

1.3. Compiling and Running the MESHMAKER V1.5 Application

As discussed already, the MESHMAKER V1.5 application comprising a single file, the **`MeshMaker.f95`** code (i.e., an autonomous code unit), written in standard FORTRAN 95/2003. It has been designed for maximum portability, and runs on any computational form (Unix and Linux workstations, PC, Macintosh) for which such compilers are available. Compiling and running the code involves the standard procedures followed in these processes, as dictated by the requirements of the specific compiler employed by the user.

NOTE: In compiling **`MeshMaker.f95`**, *it is important that the free-format source code option be invoked for proper compilation of the FORTRAN 95/2003 code*.

2.0. Geometrical Representation, Domain Discretization, and Grid Generation in MESHMAKER V1.5

The space discretization requirements, specification of elements (data block **ELEM**), connections between elements (data block **CONNE**), and the generation of complete grids (file **MESH**) that are needed for TOUGH+ and TOUGH2 simulations are described in detail elsewhere [*Pruess et al*., 1999; 2012; *Moridis et al.*, 2008; *Moridis and Pruess*, 2014]. In this section we discuss briefly the conventions used in TOUGH+ v1.5 and TOUGH2 codes for entering and processing geometrical data, as well as the **MeshMaker.f90** input requitements.

7.1. TOUGH+ and TOUGH2 Conventions for Geometrical Data

Handling of flow geometry data in TOUGH+ [*Moridis and Pruess*, 2014] is backward compatible with the TOUGH2 [*Pruess et al*., 1999; 2012] input formats and data handling. As in other *integral finite difference* codes [*Edwards*, 1972; *Narasimhan and Witherspoon*, 1976], flow geometry is defined by means of a list of volume elements (*grid blocks*), and a list of flow connections between them. This formulation can handle regular and irregular flow geometries in one, two, and three dimensions. Single- and multiple-porosity systems (porous and fractured media) can be specified, and higher order methods, such as seven- and nine-point differencing, can be implemented by means of appropriate specification of geometric data [*Pruess and Bodvarsson*, 1983].

In TOUGH+ v1.5 [*Moridis and Pruess*, 2014], as in TOUGH2 [*Pruess et al*., 1999; 2012], volume elements are identified by names that consist of a string of either five or eight characters, '`12345`' or '`12345678`'. These are arbitrary, except that the last two characters (#4 and #5 in 5-character names; #7 and #8 in 8-character names) must be numbers; an example of a valid 5-character element name is '`ELE10`', and an example of a valid 8-character name is '`AB00CC23`'. Flow connections are specified as ordered pairs of elements, such as such as '`ELE10ELE11`' (5-character names) or '`AB00CC23AB00CC24`' (8-character names). A variety of options and facilities are available for entering and processing geometric data. As in TOUGH2 [*Pruess et al*., 1999; 2012], element volumes and domain identification can be provided by means of a data block **`ELEME`** in the *standard input file*, while a data block **`CONNE`** can be used to supply connection data, including interface area, nodal distances from the interface, and orientation of the nodal

line relative to the vertical. These data are internally written to a disk file **MESH**, which in turn initializes the geometry data arrays used during the flow simulation. It is also possible to omit the **ELEME** and **CONNE** data blocks from the *standard input file*, and provide geometry data directly on a disk file **MESH**.

The MESHMAKER V1.5 application involves the **MeshMaker.f95** code, and can perform the mesh generation and processing operations discussed earlier. It can generate two-dimensional radially symmetric (*r,z*) meshes, and one-, two-, and three-dimensional rectilinear (Cartesian) grids in (*x,y,z*). Multiple-porosity processing for simulation of flow in naturally fractured reservoirs can be invoked by means of a keyword **MINC**, which stands for *Multiple INteracting Continua* (see *Pruess et al*., 2012; *Moridis and Pruess*, 2014). The MINC process operates on the data of the primary (porous medium) mesh as provided on disk file **MESH**, and generates a secondary mesh containing fracture and matrix elements with identical data formats on file **MINC**. The file **MESH** used in this process can be either directly supplied by the user, or it can have been generated from an earlier application of the **MeshMaker.f95** application.

2.2. The MeshMaker.f95 Code

In this section we discuss the use of the **MeshMaker.f95** code, and the required parameter inputs for mesh generation and processing. This section provides detailed instructions for preparing the input files, illustrative examples of which are shown in **Figures 1** to **4**.

2.2.1. Inputs Related to Problem Definition and Dimensioning

These inputs occupy two records (both mandatory), and provide (a) a short description of the problem through an informative title, and (b) data to allow proper dimensioning of the work arrays and formatting of the **MESH** file to be generated. These two initial records are discussed in detail below:

Record MESHMAKER.1

The first record of the input file in any **MeshMaker.f95** application is the character variable **TITLE**, which includes a header of up to 80 characters and is read using a free format. This record is necessary for any **MeshMaker** simulation to begin.

Record MESHMAKER.2

The following variables are read in MESHMAKER.2 using a free format:

```
MaxNum_Elem, Longest, ElemNameLength,
FormatType, LengthUnits, media_by_number
```

These parameters are defined as follows:

`Max_NumElem`
: Integer denoting the maximum number of elements in the grid under construction.

`Longest`
: Integer indicating the maximum expected number of subdivisions along any of the coordinates in the grid under construction.

`ElemNameLength`
: Integer variable defining the number of characters in the element names. It may be either 5 or 8. If unequal to 8, `ElemNameLength` is internally resest to the default (`= 5`).

`FormatType`
: Character variable indicating the format of the data in the MESH file to be created. It may have one of two values:

  `= 'Old'`: This option creates a **MESH** file that conforms to the data format described in Sections 7.2 and 7.3, and is consistent with the TOUGH2 formats [*Pruess et al*., 1999; 2012].

= 'New': This option creates a **MESH** file in which the element and connection data are listed using the NAMELIST format facility available in FORTRAN 95/2003. This is intended for future versions of TOUGH+

NOTE: *Currently, the option* FormatType = 'Old' *must be used in all* **MeshMaker.f95** *applications*.

LengthUnits
: Character variable specifying the units of length of the grid to be created. The possible options are: 'm', 'mm', 'km', 'ft' and 'in', indicating meters, millimeteres, kilometers, feet and inches, respectively. Note that all dimensions are converted internally into SI units (meters), which is the standard unit of length of the resulting mesh.

media_by_number
: Logical variable specifying how the various media (rocks) are to be named. The possible options are: .T. or .TRUE., in which case the media are named by the character variable H_RegionName (see Section 7.4.2), and .F. of .FALSE., indicating media identification by the number in the sequence of their listing. The media_by_number = .T. option is to be used only in the case of fractured media that are to be described by the Multiple Intercative Continua (MINC) concept, and is needed for the creation of an MESH interim files. This will be further processed to create the MINC file to be used in the simulation. When the media_by_number = .TRUE. option is invoked, unfractured media are identified by a negative number, and fractured media (identified by an 'F' or 'f' as the first letter of the value of the H_RegionName character variable) are identified by a positive number.

2.2.2. Inputs Related to Domain Heterogeneity

These optional inputs provide information that allows the description of heterogeneity within the domain. The assignment of heterogeneity is based on the definition of *regions* using geometrical information that describe the location and extent of these subdomains. Thus, each of the individually defined regions is assigned the properties of a particular medium that will have to be included in the **ROCKS** data block in the input file of the subsequent TOUGH+ v1.5 or TOUGH2 simulation.

The records in this data block are discussed in detail below:

Record MESHMAKER.3 (**Optional**)

This record includes only the keyword (character variable) 'Regions', which is read using a free format. If the domain is homogeneous, then there is no need to provide any of the inputs discussed in Section 7.4.2.

Record MESHMAKER.3.1 (**Optional** – when the MESHMAKER.3 record is included)

This record is read using a free format, and includes the single integer variable Num_HetRegions (> 0) that describes the number of heterogeneous regions (subdomains) that are to be described by the ensuing data records. The minimum value that Num_HetRegions can accept is 1, corresponding to a homogeneous system. An error will occur and the execution will stop if Num_HetRegions <1.

Because of dynamic dimensioning, there is no limit in the number of heterogeneous regions defined by Num_HetRegions. However, practical considerations may limit the size of Num_HetRegions. Thus, although it is possible to define element-by-element heterogeneity using this approach, this would be a very tedious process. Generally speaking, the most useful application of this facility is in the description of extensive geologic units with distinctly different properties.

Record MESHMAKER.3.2 (**Optional** – when the MESHMAKER.3 record is included)

This record is read using a free format, and includes the single character variable dominant_medium that provides the name (5 character long) of the dominant (reference) porous medium in the heterogeneous domain. Note that the selection of a medium as dominant_medium is be arbitrary, as there are no restrictions on the extent of its spatial distribution for it to be designated as such.

If Num_HetRegions = 1, no more data need to be read. This case is equivalent to a homogeneous system, and the entire heterogeneity-related data block (records MESHMAKER.3 to MESHMAKER.3.2 may be omitted.

Record MESHMAKER.3.3.0 (**Optional** – when the MESHMAKER.3 record is included and Num_HetRegions > 1)

This record is read using a free format, and includes the single character variable H_RegionName that provides the name (5 character long) of the porous medium in the region (subdomain) that is about to be defined.

Record MESHMAKER.3.3.1 (**Optional** – when the MESHMAKER.3 record is included and Num_HetRegions > 1)

Here the following character variables are read using a free format:

H_RegionCoordinates, H_RegionUnits

These parameters are defined as follows:

H_RegionCoordinates
: A character variable indicating the coordinate system used in the geometric definition of the region that is about to be described in the heterogeneous domain. It my have one of two values:

  = 'Cartesian' : This option indicates that the region is defined geometrically in terms of Cartesian coordinates.

  = 'Cylindrical' : This option indicates that the region is defined geometrically in terms of cylindrical coordinates.

H_RegionUnits
: A character variable describing the units of length used in the description of the geometry of the region. The following values are acceptable options:

  'mm', 'km', 'km', 'in', or 'ft',

  indicating millimeters, meters, kilometers, inches and feet, respectively. Note that **MeshMaker.f95** converts all length units into meters (the length unit used in the TOUGH+ simulations) prior to producing the **MESH** file.

Record MESHMAKER.3.3.2 (**Optional** – when the MESHMAKER.3 record is included and Num_HetRegions > 1)

If H_RegionCoordinates = 'Cartesian', the following real variables are read in MESHMAKER.3.3.2 using a free format:

Xmin, Xmax, Ymin, Ymax, Zmin, Zmax

These parameters are defined as follows:

Xmin, Xmax
: Real variables indicating the range of the region along the *x*-axis of the Cartesian coordinate system.

Ymin, Ymax
: Real variables indicating the range of the region along the *y*-axis of the Cartesian coordinate system.

`Zmin, Zmax`
: Real variables indicating the range of the region along the *z*-axis of the Cartesian coordinate system.

If `H_RegionCoordinates = 'Cylindrical'`, the following real variables are read in `MESHMAKER.3.3.2` using a free format:

`Rmin, Rmax, Zmin, Zmax`

These parameters are defined as follows:

`Rmin, Rmax`
: Real variables indicating the range of the region along the *r*-axis of the cylindrical coordinate system.

`Zmin, Zmax`
: Real variables indicating the range of the region along the *z*-axis of the cylindrical coordinate system.

Repeat records `MESHMAKER.3.3.0`, `MESHMAKER.3.3.1`, `MESHMAKER.3.3.3` for a total of `Num_HetRegions – 1` regions. Note that the region `dominant_medium` does not need geometric definition.

2.2.3. Inputs Related to Description of Boundaries

These optional inputs provide information that describes the outer boundaries of the domain under discretization. The assignment of grid subdomains as boundaries is based on the geometry-based definition of the outer spatial limits of the domain under discretization. Thus, the boundaries are treated as special types of regions (see Section 2.2.2) with the properties of a particular medium that will have to be included in the **ROCKS** data block in the input file of the subsequent TOUGH+ simulation. It is possible for a region and one or more boundaries to be described by the same medium (i.e., a medium that has the same name) in the **ROCKS** data block.

The records in this data block are discussed in detail below:

Record MESHMAKER.4 (**Optional**)

This record includes only the keyword (character variable) 'Boundaries', which is read using a free format. If the domain is confined by no-flow (Newman-type) boundaries, then there is no need to provide any of the inputs discussed in Section 7.4.3.

Record MESHMAKER.4.1 (**Optional** – when the MESHMAKER.4 record is included)

This record is read using a free format, and includes the single integer variable Num_Boundaries (> 0) that describes the number of boundaries that are to be described.

Record MESHMAKER.4.2.0 (**Optional** – when the MESHMAKER.4 record is included and Num_Boundaries > 1)

The following character variables are read in MESHMAKER.4.2 using a free format:

BoundID, BoundRegionName

These parameters are defined as follows:

BoundID

A character variable indicating the type of boundary described at the prescribed location. It my have one of two values:

= 'I': This option indicates an *inactive* boundary, the conditions and properties of which are time-invariant.

= 'V': This option indicates a time-variable boundary.

The result of this designation is reflected in the **ELEME** block, in which elem_activity (see Section 7.2) is set to BoundID in all the cells corresponding to the boundary defined here. Any other value of the BoundID variable causes the program to print an error message and stop execution.

BoundRegionName

A character variable H_RegionNname that provides the name (5 character long) of the porous medium in the boundary that is about to be defined.

Record MESHMAKER.4.2.1 (**Optional** – when the MESHMAKER.4 record is included and Num_Boundaries > 1)

Here the following character variables are read using a free format:

BoundRegionCoordinates, BoundRegionUnits

These parameters are defined as follows:

BoundRegionCoordinates

A character variable indicating the coordinate system used in the geometric definition of the boundary that is about to be described in the heterogeneous domain. It my have one of two values:

= 'Cartesian': This option indicates that the region is defined geometrically in terms of Cartesian coordinates.

= 'Cylindrical': This option indicates that the region is defined geometrically in terms of cylindrical coordinates.

BoundRegionUnits

A character variable describing the units of length used in the description of the geometry of the boundary. The following values are acceptable options:

'mm', 'km', 'km', 'in', or 'ft',

indicating millimeters, meters, kilometers, inches and feet, respectively. Note that **MeshMaker.f95** converts all length units into meters (the length unit used in the TOUGH+ simulations) prior to producing the **MESH** file.

Record MESHMAKER.4.2.2 (**Optional** – when the MESHMAKER.4 record is included and Num_Boundaries > 1)

If BoundRegionCoordinates = 'Cartesian', the following real variables are read in MESHMAKER.3.3.2 using a free format:

Xmin, Xmax, Ymin, Ymax, Zmin, Zmax

These parameters are defined as follows:

Xmin, Xmax

Real variables indicating the range of the boundary region along the *x*-axis of the Cartesian coordinate system.

`Ymin, Ymax`
Real variables indicating the range of the boundary region along the *y*-axis of the Cartesian coordinate system.

`Zmin, Zmax`
Real variables indicating the range of the boundary region along the *z*-axis of the Cartesian coordinate system.

If `BoundRegionCoordinates = 'Cylindrical'`, the following real variables are read in `MESHMAKER.3.3.2` using a free format:

```
Rmin, Rmax, Zmin, Zmax
```

These parameters are defined as follows:

`Rmin, Rmax`
Real variables indicating the range of the region boundary along the *r*-axis of the cylindrical coordinate system.

`Zmin, Zmax`
Real variables indicating the range of the boundary region along the *z*-axis of the cylindrical coordinate system.

Repeat records `MESHMAKER.4.2.0, MESHMAKER.4.2.1, MESHMAKER.4.2.3` for a total of `Num_Boundaries` boudaries.

NOTE: *This capability creates files that are directly usable by TOUGH+ codes [Moridis and Pruess, 2014], which can differentiate between active and inactive (with constant or time-variable conditions) elements and do not require positioning of inactive elements at the end of the list of the active elements. Standard TOUGH2 [Pruess et al., 1999; 2012] does not have this capability. Thus, if* **`MeshMaker.f95`** *is used to prepare the mesh for a standard TOUGH2 simulation, the user needs to modify the resulting MESH file (either manually or by using an appropriate script) by (a) collecting all elements having the "I" or "V" activity identifier (variable* `BoundID`*), (b) moving them below the last active element in the* **`ELEME`** *data block, and (c) setting the volume of the first inactive element to a zero or negative value (see Moridis and Pruess [2014]).*

2.2.4. Inputs for Grid Construction

There are the grid construction options available in **`MeshMaker.f95.`** These options are activated by appropriate keywords. Thus, the keywords '`RZ2D`' or '`RZ2DL`' invoke generation of a one or two-dimensional radially symmetric (*r,z*) mesh; the keyword '`XYZ`'

initiates generation of a one, two, or three dimensional Cartesian (*x,y,z*) mesh. The third is invoked by the keyword 'MINC', and will be discussed in the Section 2.2.5.

The meshes generated under keyword 'RZ2D' or 'XYZ' are internally written to file **MESH**. We shall now separately describe the preparation of input data for the grid construction options.

7.4.4.1. Generation of radially symmetric grids (keyword 'RZ2D' or 'RZ2DL'). The keywords 'RZ2D' or 'RZ2DL' invoke generation of a radially symmetric mesh. Values for the radii to which the grid blocks extend can be provided by the user or can be generated internally (see below). Nodal points will be placed halfway between neighboring radial interfaces. When 'RZ2D' is specified, the mesh will be generated by columns; i.e., in the **ELEME** block we will first have the grid blocks at smallest radius for all layers, then the next largest radius for all layers, and so on.

With keyword 'RZ2DL' the mesh will be generated by layers; i.e., in the **ELEME** block we will first have all grid blocks for the first (top) layer from smallest to largest radius, then all grid blocks for the second layer, and so on. Apart from the different ordering of elements, the two meshes for 'RZ2D' and 'RZ2DL' are identical. The reason for providing the two alternatives is as a convenience to users in implementing boundary conditions by way of inactive elements (see Section 6.4 in *Pruess et al*. [1999]). Assignment of inactive elements would be made by using a text editor on the RZ2D-generated **MESH** file, and moving groups of elements towards the end of the **ELEME** block, past a dummy element with zero volume. 'RZ2D' makes it easy to declare a vertical column inactive, facilitating assignment of boundary conditions in the vertical, such as a

gravitationally equilibrated pressure gradient. 'RZ2DL' on the other hand facilitates implementation of areal (top and bottom layer) boundary conditions.

The inputs for cylindrical systems are as follows:

Data Block GRID

The first record in this data block includes only the keyword (character variable) 'RZ2D' or 'RZ2DL'. This keyword is read using a Format (A5).

Record RADII.0

RADII: A keyword that introduces data for defining a set of interfaces (grid block boundaries) in the radial direction. It is read using a Format (A5).

Record RADII.1

Format (I5)
NRAD

NRAD: Number of radii that will be read. At least one radius must be provided, indicating the inner boundary of the mesh.

Record RADII.2, RADII.3, etc.

Format (8E10.4)
RC(i), i = 1,NRAD

RC(i): The radii defining the element boundaries in ascending order [m].

Record EQUID.0

EQUIDistant
Keyword indicating that the ensuing data describe a set of equal radial increments. This keyword is read using Format (A5).

Record EQUID.1

Format (I5, 5X, E10.4)
NEQU, DR

NEQU: The number of desired radial increments.

DR: The size of the radial increment [m].

NOTE: *At least one radius must have been defined via block* RADII *before* EQUID *can be invoked*.

Record LOGAR.0

LOGARithmic — This keyword introduces data on radial increments that increase from one to the next by the same factor (i.e., $\Delta r_{n+1} = f \bullet \Delta r_n$).

Record LOGAR.1

Format (I5, 5X, 2E10.4)
NLOG, RLOG, DR

NLOG: The number of desired radial increments.

RLOG: The desired radius r_{max} of the last (largest) of these radii.

DR: The reference radial increment Δr_0: the first Δr generated will be equal to $f \bullet \Delta r_0$, with f internally determined such that the last increment will bring total radius to RLOG = r_{max}. The factor $f < 1$ for decreasing radial increments is permissible. If Δr_0 is set equal to zero, or left blank, the last increment Δr generated before the keyword **LOGAR** is invoked will be used as default.

Additional blocks RADII, EQUID, and LOGAR can be specified in arbitrary order.

NOTE: *At least one radius must have been defined before the* LOGAR *option can be invoked. If* $\Delta r_0 = 0$, *at least two radii must have been defined*.

Record LAYER.0

LAYER: This keyword introduces information on horizontal layers, and signals closure of RZ2D input data. It is read using a Format (A5).

Record LAYER.1

Format (I5)
NLAY

NLAY: The number of horizontal layers in the cylindrical grid.

Record LAYER.2

Format (8E10.4)
H(i), i = 1, NLAY

H(i): The thicknesses of the horizontal layers in the cylindrical grid, from top layer downward. By default, zero or blank entries for layer thickness will result in assignment of the last preceding non-zero entry. Assignment of a zero layer thickness, as needed for inactive layers, can be accomplished by specifying a negative value.

Record GRID-END

Two blank records close the input data file *if* further MINC-processing of the grid is not needed. If such processing is specified, then **a single** blank record terminates the RZ2D or RZ2DL data block, and is then followed by the MINC data block (see Section 7.2.5).

7.4.4.2. Generation of rectilinear grids (keyword 'XYZ')

Data Block GRID

The first record in this data block includes only the keyword (character variable) 'XYZ' that is invokes generation of a Cartesian (rectilinear) mesh. This keyword is read using a Format (A5).

Record XYZ.1

DEG, X_ref, Y_ref, Z_ref (Free format)

DEG: The angle (in degrees) between the *y*-axis and the horizontal. If gravitational acceleration (parameter gravity in record PARAM.2, see Section 10) is positive, -90° < DEG < 90° corresponds to grid layers going from top down. Grids can be specified from the bottom layer up by setting gravity or ConxBeta (Section 7.3) to negative values. The default (DEG = 0.0e0) corresponds to horizontal *y*- and vertical *z*-axis. The *x*-axis is always horizontal.

X_ref, Y_ref, Z_ref:
The reference *x*-, *y*-, and *z*-coordinates at the origin of the axes.

Record XYZ.2

Format (A2, 3X, I5, E10.4)
NTYPE, NO, DEL

NTYPE: A character variable that can assume one of the values 'NX', 'NY' or 'NZ', specifying grid increments in the *x*-, *y*-, or *z*-direction, respectively.

NO: The number of grid increments.

DEL: The constant grid increment for NO grid blocks, if set to a non-zero value.

Record XYZ.3 (**Optional**, DEL = 0.0e0 or blank only)

Format (8E10.4)

DEL(i), i = 1, NO

DEL(i): A set of grid increments in the direction specified by NTYPE in record XYZ.2. Additional records with formats as XYZ.2 and XYZ.3 can be provided, with *x*-, *y*-, and *z*-data in arbitrary order.

Record GRID-END

Two blank records close the input data file if further MINC-processing of the grid is not needed. If such processing is specified, then **a single** blank record terminates the RZ2D or RZ2DL data block, and is then followed by the MINC data block (see Section 7.2.5).

2.2.5. **MINC** *processing for fractured media*

This is the third grid construction options available in **MeshMaker.f95.** It is applicable to fractured media an is invoked by the 'MINC' keyword, which calls a modified version of the GMINC program [*Pruess*, 1983] to sub-partition a primary porous medium mesh into a secondary mesh for fractured media, using the method of *Multiple INteracting Continua* [*Pruess and Narasimhan*, 1982; 1985]. The MINC-processing operates on the data in a file **MESH** that is either pre-existing or is created directly from the inputs of **MeshMaker.f95** when the 'RZ2D', 'RZ2DL' or 'XYZ' data blocks are followed by the 'MINC' datablock.

NOTE*: If the application of the* ***MeshMaker.f95*** *aims to process a pre-existing external* ***MESH*** *file, the input file begins with records* ***MESHMAKER.1*** *and* ***MESHMAKER.2*** *(see Section 2.2.1) as the first data block, followed by the data inputs described in detail below.*

```
Meshmaker test: Cartesian grid
1000 20  5 'Old' 'm'  .FALSE.
XYZ
       00.
NX      10    1.0e00
NY       5       2.0
NZ      20       1.0
```

Figure 1. An example of a **`MeshMaker.f95`** input file for the creation of a Cartesian 3D grid. Note that no heterogeneous regions or boundaries are defined in this grid.

```
Meshmaker test: Cylindrical grid
6000  6000  5 'Old' 'm'  .FALSE.
RZ2DL
RADII
    2
    1.0e-5    1.0e-3
EQUID
 4000         2.5e-2
LOGAR
  500         2.5e+3
EQUID
    1         1.0e-3
LAYER
    1
    1.0e00
```

Figure 2. An example of a **`MeshMaker.f95`** input file for the creation of a single-layer (1D) cylindrical grid. Note that no heterogeneous regions or boundaries are defined in this grid

```
Input file for a large 3D cartesian grid       ! Title
500000 100 5  'Old' 'm' .FALSE.
Regions                                        ! Keyword denoting heterogeneous subdomains
6                                              ! ==> # of heterogeneous media regions
'HydrL'                                        ! Region #1: Name of dominant medium
'Aquif'                                        ! Region #2: medium name
'cartesian'     'm'                            !    coordinates, units
0.0e0 1.5e3 0.0e0 1.5e3 -6.3e1 -4.825e1        !    Xmin, Xmax, Ymin, Ymax, Zmin, Zmax
'OverB'                                        ! Region #3: medium name
'cartesian'   'm'                              !    coordinates, units
0.0e0 1.5e3 0.0e0 1.5e3  -3.0e1  0.0e1         !    Rmin, Rmax, Zmin, Zmax
'UndrB'                                        ! Region #4: medium name
'cartesian'   'm'                              !    coordinates, units
0.0e0 1.5e3 0.0e0 1.5e3  -9.4e1 -6.3e1         !    Rmin, Rmax, Zmin, Zmax
'WellA'                                        ! Region #5: medium name
'cartesian'   'm'                              !    coordinates, units
0.0e0 5.0e-2 0.0e0 5.0e-2 -5.4e1 -3.0e1        !    Rmin, Rmax, Zmin, Zmax
'WellA'                                        ! Region #6: medium name
'cartesian'   'm'                              !    coordinates, units
4.9995e2 5.0e2 4.9995e2 5.0e2 -5.4e1 -3.0e1    !    Rmin, Rmax, Zmin, Zmax
Boundaries
2                                              ! ==> # of boundaries
'I' 'TopBB'                                    ! Boundary #1: type and medium name
'cartesian'   'm'                              !    coordinates, units
0.0e0 1.5e3 0.0e0 1.5e3 -1.0e-2  0.0e0         !    Rmin, Rmax, Zmin, Zmax
'I' 'BotBB'                                    ! Boundary #2: type and medium name
'cartesian'   'm'                              !    coordinates, units
0.0e0 1.5e3 0.0e0 1.5e3 -9.4e1 -9.3e1          !    Rmin, Rmax, Zmin, Zmax
XYZ
       00.
NX      76
    5.0e-2 2.5000e-1 2.8599e-1 3.2716e-1 3.7426e-1 4.2813e-1 4.8976e-1 5.6027e-1
 6.4092e-1 7.3319e-1 8.3873e-1 9.5947e-1 1.0976e+0 1.2556e+0 1.4363e+0 1.6431e+0
 1.8797e+0 2.1502e+0 2.4598e+0 2.8139e+0 3.2190e+0 3.6823e+0 4.2124e+0 4.8188e+0
 5.5125e+0 6.3061e+0 7.2139e+0 8.2524e+0 9.4404e+0 1.0799e+1 1.2354e+1 1.4132e+1
 1.6167e+1 1.8494e+1 2.1157e+1 2.4202e+1 2.7686e+1 3.1672e+1 3.1672e+1 2.7686e+1
 2.4202e+1 2.1157e+1 1.8494e+1 1.6167e+1 1.4132e+1 1.2354e+1 1.0799e+1 9.4404e+0
 8.2524e+0 7.2139e+0 6.3061e+0 5.5125e+0 4.8188e+0 4.2124e+0 3.6823e+0 3.2190e+0
 2.8139e+0 2.4598e+0 2.1502e+0 1.8797e+0 1.6431e+0 1.4363e+0 1.2556e+0 1.0976e+0
 9.5947e-1 8.3873e-1 7.3319e-1 6.4092e-1 5.6027e-1 4.8976e-1 4.2813e-1 3.7426e-1
 3.2716e-1 2.8599e-1 2.5000e-1    5.0e-2
NY      76
    5.0e-2 2.5000e-1 2.8599e-1 3.2716e-1 3.7426e-1 4.2813e-1 4.8976e-1 5.6027e-1
 6.4092e-1 7.3319e-1 8.3873e-1 9.5947e-1 1.0976e+0 1.2556e+0 1.4363e+0 1.6431e+0
 1.8797e+0 2.1502e+0 2.4598e+0 2.8139e+0 3.2190e+0 3.6823e+0 4.2124e+0 4.8188e+0
 5.5125e+0 6.3061e+0 7.2139e+0 8.2524e+0 9.4404e+0 1.0799e+1 1.2354e+1 1.4132e+1
 1.6167e+1 1.8494e+1 2.1157e+1 2.4202e+1 2.7686e+1 3.1672e+1 3.1672e+1 2.7686e+1
 2.4202e+1 2.1157e+1 1.8494e+1 1.6167e+1 1.4132e+1 1.2354e+1 1.0799e+1 9.4404e+0
 8.2524e+0 7.2139e+0 6.3061e+0 5.5125e+0 4.8188e+0 4.2124e+0 3.6823e+0 3.2190e+0
 2.8139e+0 2.4598e+0 2.1502e+0 1.8797e+0 1.6431e+0 1.4363e+0 1.2556e+0 1.0976e+0
 9.5947e-1 8.3873e-1 7.3319e-1 6.4092e-1 5.6027e-1 4.8976e-1 4.2813e-1 3.7426e-1
 3.2716e-1 2.8599e-1 2.5000e-1    5.0e-2
NZ      77
    1.0e-3    7.00e0    5.00e0    4.0e+0    3.2e+0    2.5e+0    2.0e00    1.6e00
   1.25e00    1.0e00    8.0e-1    6.5e-1    5.0e-1    5.0e-1    4.0e-1    4.0e-1
    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1
    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1
    4.0e-1    4.0e-1    4.5e-1    4.5e-1    4.5e-1    4.5e-1    4.5e-1    4.0e-1
    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1
    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1    4.0e-1
    4.0e-1    4.0e-1    4.0e-1    5.0e-1    5.0e-1    6.5e-1    8.0e-1    1.0e00
    1.0e00   1.25e00    1.6e00    2.0e00    2.5e00    3.2e00    3.0e00    4.00e0
    4.0e00    5.0e00    6.0e00    8.0e00    1.0e-3


====> ====> ====> ====>
```

Figure 3. An example of a **`MeshMaker.f95`** input file for the creation of a large Cartesian 3D grid with heterogeneous regions and defined boundaries.

```
Input file for a large cylindrical grid  ……! Title
15000 30000 5 'Old' 'm' .FALSE.
Regions                        ! Keyword denoting heterogeneous subdomains
  6                            ! => Num_HetRegions ( = number of heterogeneous regions)
'HydrL'                        !    Region #1: dominant_medium
'Aquif'                        !    Region #2: H_RegionName
'cylindrical'   'm'            !       H_RegionCoordinates, H_RegionUnits
0.0e0 1.5e3 -6.3e1 -4.825e1    !       Rmin, Rmax, Zmin, Zmax
'OverB'                        !    Region #3: H_RegionName
'cylindrical'   'm'            !       H_RegionCoordinates, H_RegionUnits
0.0e0 1.5e3 -3.0e1  0.0e1      !       Rmin, Rmax, Zmin, Zmax
'UndrB'                        !    Region #4: H_RegionName
'cylindrical'   'm'            !       H_RegionCoordinates, H_RegionUnits
0.0e0 1.5e3 -9.4e1 -6.3e1      !       Rmin, Rmax, Zmin, Zmax
'Casng'                        !    Region #5: H_RegionName
'cylindrical'   'm '           !       H_RegionCoordinates, H_RegionUnits
0.0e0 1.0e-1 -4.6e1 -3.0e1     !       Rmin, Rmax, Zmin, Zmax
'Perfo'                        !    Region #5: H_RegionName
'cylindrical'   'm '           !       H_RegionCoordinates, H_RegionUnits
0.0e0  1.0e-1 -5.225e1 -4.6e1  !       Rmin, Rmax, Zmin, Zmax
Boundaries                     ! Keyword denoting boundaries
2                              ! ==> Num_Boundaries
'I' 'TopBB'                    !    Boundary #1: BoundID, BoundRegionName
'cylindrical'   'm'            !       BoundRegionCoordinates, BoundRegionUnits
0.0e0 1.5e3 -1.0e-2  0.0e0     !       Rmin, Rmax, Zmin, Zmax
'I' 'BotBB'                    !    Boundary #2: BoundID, BoundRegionName
'cylindrical'   'm'            !       BoundRegionCoordinates, BoundRegionUnits
0.0e0 1.5e3 -9.4e1 -9.3e1      !       Rmin, Rmax, Zmin, Zmax
RZ2DL
RADII
    2
     0.001   0.10795
EQUID
    1         2.0e-2
LOGAR
  199         1.0e+3
LAYER
  68
    1.0e-3    6.50e0    5.00e0    5.0e+0    3.0e+0    3.0e+0    2.0e00    2.0e00
    1.0e00    1.0e00    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1
    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1
    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1
    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1
    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.5e-1    5.5e-1    5.5e-1    5.5e-1
    5.5e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    5.0e-1    1.0e00
    1.0e00    2.0e00    2.0e00    3.0e+0    3.0e+0    4.0e+0    4.0e+0    5.00e0
    5.0e+0    6.00e0    6.0e+0    1.0e-3


====> ====> ====> ====>
```

Figure 4. An example of a **`MeshMaker.f95`** input file for the creation of a large cylindrical 2D grid with multiple layers, heterogeneous regions and defined boundaries.

Data Block GRID

The first record in this data block includes only the keyword (character variable) 'MINC' that invokes post-processing of a primary porous medium mesh from a previously developed file **MESH**. This keyword is read using a Format (A5) or (A8). The input formats in data block MINC are identical to those of the GMINC program [*Pruess*, 1983], with two enhancements: (a) there is an additional facility for specifying global matrix-matrix connections (*dual permeability* option); (b) only active elements (see *Moridis and Pruess* [2014]) will be subjected to MINC-processing, the remainder of the **MESH** remaining unaltered as porous medium grid blocks.

Record MINC.1

Format (2A5, 5X, A5)
PART, TYPE, DUAL

PART: This is the first keyword following the 'MINC' keyword. It will be followed on the same line by parameters TYPE and DUAL with information on the nature of fracture distributions and matrix-matrix connections.

PART: An identifier of the data block with partitioning parameters for secondary mesh.

TYPE: A five-character variable for selecting one of the following six different proximity functions provided in MINC [*Pruess*, 1983].

= 'ONE-D': A set of plane parallel infinite fractures with matrix block thickness between neighboring fractures equal to PAR(1).

= 'TWO-D': Two sets of plane parallel infinite fractures, with arbitrary angle between them. Matrix block thickness is PAR(1) for the first set, and PAR(2) for the second set. If PAR(2) is not specified explicitly, it will be set equal to PAR(1).

= 'THRED': Three sets of plane parallel infinite fractures at right angles, with matrix block dimensions of PAR(1), PAR(2), and PAR(3), respectively. If PAR(2) and/or PAR(3) are not explicitly specified, they will be set equal to PAR(1) and/or PAR(2), respectively.

= 'STANA': Average proximity function for rock loading of Stanford large reservoir model [*Lam et al.*, 1988].

= 'STANB': Proximity function for the five bottom layers of Stanford large reservoir model.

= 'STANT': Proximity function for top layer of Stanford large reservoir model.

DUAL A five-character word for selecting the treatment of global matrix flow.

= ' ' (Blank – default): The global flow occurs only through the fracture continuum, while rock matrix and fractures interact locally by means of interporosity flow (*double-porosity* model).

= 'MMVER': The global matrix-matrix flow is permitted only in the vertical; otherwise like the double-porosity model; for internal consistency this choice should only be made for flow systems with one or two predominantly vertical fracture sets.

= 'MMALL': The global matrix-matrix flow in all directions; for internal consistency only two continua, representing matrix and fractures, should be specified (dual-permeability model).

NOTE: *A user wishing to employ a different proximity function other than the options provided through the* TYPE *variable in* ***MINC*** *needs to replace the function subprogram* PROX(x) *in* **MeshMaker.f95** *with a routine of the form:*

```
FUNCTION PROX(x)
PROX = (arithmetic expression in x)
RETURN
END
```

It is necessary that PROX(x) *be defined even when x exceeds the maximum possible distance from the fractures, in which case* PROX = 1. *Additionally, when the users supply their own proximity function subprogram, the parameter* TYPE *must be set equal to* 'ONE-D', 'TWO-D', *or* 'THRED', *depending on the dimensionality of the proximity function. This will assure proper definition of the innermost nodal distances* [*Pruess,* 1983].

Record PART.l

Format (2I3, A4, 7E10.4)
J, NVOL, WHERE, (PAR(i), i = 1, 7)

J: The total number of multiple interacting continua (J < 36).

NVOL: The total number of explicitly provided volume fractions (NVOL < J). If NVOL < J, the volume fractions with indices NVOL+1, ..., J will be

internally generated; all being equal and chosen such as to yield proper normalization to 1.

`WHERE`: Character variable specifying whether the sequentially specified volume fractions begin with the fractures (`WHERE = 'OUT'`) or in the interior of the matrix blocks (`WHERE = 'IN'`).

`PAR(i)`: Real array that stores the parameters describing the fracture spacing (see discussion of previous record).

Record `PART.2.1, PART.2.2, etc.`

Format (8E10.4)
`(VOL(i), i = 1, NVOL)`

`VOL(i)`: The volume fraction (having a value between 0 and 1) of a continuum with index `i` (for `WHERE = 'OUT'`) or index `J+1-i` (for `WHERE = 'IN'`). `NVOL` volume fractions will be read. For `WHERE = 'OUT'`, `i=1` is the fracture continuum, `i=2` is the matrix continuum closest to the fractures, `i=3` is the matrix continuum adjacent to `i=2`, etc. The sum of all volume fractions must not exceed 1.

Record `GRID-END`

Two blank records close the input data file.

```
MINC test: Cartesian grid
1010 21  5 'Old' 'm' .TRUE.
MINC
PART THRED
  5  4OUT        50.       50.       50.
       .02       .08       .20       .35
```

Figure 5. An example of a **`MeshMaker.f95`** input file for the MINC-processing of an external **`MESH`** file.

Acknowledgements

This work was supported by the Assistant Secretary for Fossil Energy, Office of Natural Gas and Petroleum Technology, through the National Energy Technology Laboratory, under the U.S. Department of Energy, Contract No DE-AC02-05CH11231.

PAGE LEFT INTENTIONALLY BLANK

References

Lam, S.T., A. Hunsbedt, P. Kruger and K. Pruess, Analysis of the Stanford Geothermal Reservoir Model Experiments Using the LBL Reservoir Simulator, Geothermics, **17**(4), 595- 605, LBL-25957, 1988.

Moridis, G.J., M.B. Kowalsky and K. Pruess, TOUGH+HYDRATE v1.0 User's Manual: A Code for the Simulation of System Behavior in Hydrate-Bearing Geologic Media, Report LBNL-0149E, Lawrence Berkeley National Laboratory, Berkeley, CA (2008).

Narasimhan, T.N. and P.A. Witherspoon, An Integrated Finite Difference Method for Analyzing Fluid Flow in Porous Media, *Water Resour. Res.*, **12**(1), 57 – 64, 1976.

Pan, L., User's Information for WinGridder V3.0. Report LBNL-273E, Lawrence Berkeley National Laboratory, Berkeley, CA, 2008.

Pruess, K., GMINC - A Mesh Generator for Flow Simulations in Fractured Reservoirs, Lawrence Berkeley Laboratory Report LBL-15227, Berkeley, CA, March 1983.

Pruess, K. and G.S. Bodvarsson, A Seven-Point Finite Difference Method for Improved Grid Orientation Performance in Pattern Steam Floods, *Proceedings*, Seventh Society of Petroleum Engineers Symposium on Reservoir Simulation, Paper SPE-12252, 175 - 184, San Francisco, CA, 1983.

Pruess, K., C. Oldenburg, and G. Moridis, TOUGH2 User's Guide, Version 2.0, Report LBNL-43134, Lawrence Berkeley National Laboratory, Berkeley, Calif., 1999.

Pruess, K., C. Oldenburg, and G. Moridis, TOUGH2 User's Guide, Version 2.1, Report LBNL-43134, Lawrence Berkeley National Laboratory, Berkeley, Calif., 2012.