QA: L

MOL.19990519.0191

AMESH

A mesh creating program for the Integral Finite Difference Method

User's Manual

Charles B. Haukwa

Earth Sciences Division

Ernest Orlando Lawrence Berkeley National Laboratory

Berkeley, California

AMESH

A mesh creating program

Charles B. Haukwa

1.0 Introduction

Amesh program generates discrete grids for numerical modeling of flow and transport problems in which the formulation is based on integral finite difference method (IFDM). For example, the output of Amesh can be used directly as (part of) the input to TOUGH2 or TOUGH numerical Simulator (Pruess, 1987, 1990, Pruess, et al., 1996). The code *Amesh* can generate 1D, 2D or 3D numerical grids for a given set of locations, i.e. the centers of of each discrete sub-domain. In the 2D aerial plane the Voronoi tessellation method is used (Voronoi, 1908; Ahuja, 1982; Aurehammer, 1991; Fortune, 1987, 1988, 1993). In this method we can create a mesh of elements, within model domain, where the interfaces between neighbor elements are the perpendicular bisectors of the line connecting the element centers. The interface distances are simply the medians of the line connecting the centers. To create the 3D grid, the vertical direction interface areas are always treated as horizontal projections of the 2D areal plane. In the lateral direction the interface areas are always vertical projections. In both cases the direction of gravity vector is given by the cosine of angle formed by the line joining the element centers and the vertical. From the list of element locations (center points), the program determines element volumes, and the connection information, i.e. areas, connection distances and the angle. The default input file is \`in'. The output files are \`eleme' are \`conne' and 'segmt'.

The files 'eleme' and 'conne' contain all the data required to describe a TOUGH2 input and together they describe the input TOUGH2 input file called 'MESH', for the specified domain. The file 'segmt' can be used to plot the geometrical shape of each element in each layer of the input domain.

The input data into *Amesh* does not have to be ordered. AMESH uses a fast quaternary sorting algorithm (Fortune, 1988; Watson 1985) to sort and compute the adjacency relationships between nodes in the 2D plane.

2.0 Theoretical background

2.1 Introduction

The structure of a numerical grid depends on the mathematical formulation of the flow and transport equations. For the complex geometry of subsurface flow domains, no analytical solution of the flow problem is possible and numerical techniques are the only methods that can provide insight into the physical processes of interest. Numerical discretization of the mathematical formulation, the assumptions used, and the data that must be provided for the solution, dictate the type of numerical grid to be developed. With the rapidly developing computer technology, numerical models of even complex physical problems can be solved on desktop computers. In subsurface flow problems, numerical models are used in variety of related problems such as oil reservoir simulation, geothermal reservoir simulation, saturated and unsaturated fluid flow modeling, and in contaminant transport modeling. The numerical grid for a subsurface flow problem is normally selected according to several considerations. These considerations include the geology, size, and the type of data available for the site. The grids are also developed based on the fluid displacement or depletion process to be modeled. Other considerations include numerical accuracy desired, available software, and computer resources. For example, a numerical grid for fractured geological media must consider the transport properties of both the matrix and the fractures. The grid must also accurately represent the geological and hydrogeological information of the system.

2.2 Equations Governing Subsurface Flow and Transport through Porous Media

Flow of fluids through porous media is governed by physical and empirical relationships. The balances of mass, momentum and energy are partial differential equations that describe the variation of fluid phase saturation and composition, pressure, and temperature as functions of continuous space and time coordinates in the subsurface. The relevant formulation for multi-component mass and heat transport which was extensively discussed by Narasimhan and Witherspoon (1976), and Pruess (1990), is summarized below:

First we consider the mass and energy balance for component k at node m, with volume V. In the following equation, F is the flux, n is the unit inward normal to interface Γ,

q is the sink/source term, and M is the mass accumulation (see Pruess, 1990, and Fig. 2.3.1).

$$\int_{\Gamma_m} F^{(k)}.nd\Gamma + \int_{V_m} q^{(k)} dv = \frac{d}{dt}\int_{V_m} M^{(k)} dv, \tag{2.2.1}$$

$$\begin{matrix} \textit{normal} & + & \textit{Source /} & = & \textit{Storage} \\ \textit{Flux} & & \textit{Sink} & & \textit{Change} \end{matrix}$$

The individual components of the flux are given by Darcy's law.

$$F_\beta^{(k)} = -k\frac{k_{r\beta}}{\mu_\beta}\rho_\beta X_\beta^{(k)}(\nabla P_\beta - \rho_\beta g \cos\theta) - \delta_{\beta g} D_{va}\rho_\beta \nabla X_\beta^{(k)}. \tag{2.2.2}$$

Here k is the absolute matrix permeability; $k_{r\beta}$ and μ_β are the relative permeability and the viscosity of phase β respectively; $\nabla P_\beta = P - P_{cap,\beta}$ is the pressure in phase β (P is the reference pressure, and $P_{cap,\beta}$ is the capillary pressure); and $g\cos\theta$ is the gravity vector component in flow direction θ, measured from vertical. D_{va} is the binary diffusivity of vapor in air, and δ_{bg} accounts for tortuosity and vapor saturation in the flow, $\rho_\beta g$ is the density of phase β and X is the mass fraction of component k in phase β (see Pruess [1987, 1990] and Pruess, et al. [1996] for details)

2.3 Discretization of Flow and Transport Equations

To obtain a numerical solution, the balance equations must be discretized. Among the commonly employed methods for numerical discretization of the governing equations are the Finite Difference Method (FDM), Integral Finite Difference Method (IFDM), Control Volume Method (CVM) and Finite Element Method (FEM). The numerical discretization results in a series of coupled non-linear algebraic equations for each numerical node that can be solved by appropriate linearisation or by iterative methods. The numerical grid discretization employed in *Amesh* is based on the IFDM formulation which was also employed in TOUGH2 (Pruess [1987, 1990]; Narasimhan and Witherspoon, [1976]). The IFDM scheme also allows for easy inclusion of locally refined grids in the (Heineman et.al. [1983]; Pedrosa and Aziz [1986]; Quandalle and Bisset [1983], von Rosenberg [1982]). In the IFDM method, the spatial discretization is based on the numerical discretization of the flow equation (2.2.1). In the discretized from we can define the components of equation 2.2.1 as follows:

The change in mass accumulation is defined as:

$$\int_{V_m} M dv = V_m M_m \qquad 2.3.1$$

Where M_m is the average value of M in volume V_m. The total flux crossing the the interfaces (surface integral of flux) can be approximated by the discrete summation:

$$\int_{\Gamma_m} F^k . n d\Gamma = \sum_m A_{mn} F^k_{mn} \qquad 2.3.2$$

In equation 2.3.2 F_{mn} is the average value of the inward normal component of F over the surface segment area A_{nm} between volume elements V_n and V_m. The definition of the flow geometry for the IFDM formulation is illustrated in Fig. 2.3.1. The discretized form of the flux can be expressed using averages within the volume elements (V_n and V_m) and the average flux at the interface as follows. The flow term in, Darcy's law equation 2.2.2 can be approximated by:

$$F_{\beta,nm} = -k_{nm} \left[\frac{k_{r\beta}\, \rho_\beta}{\mu_\beta} \right]_{nm} \left[\frac{P_{\beta,n} - P_{\beta,m}}{D_{nm}} - \rho_{\beta,nm}\, g_{nm} \right]. \qquad 2.3.3$$

In equation 2.3.3, D_{nm} is the distance between the nodal points *n* and *m* (see Fig. 2.3.1), and g_{nm} is the component of gravitational acceleration in the direction from *m* to *n*. A suitable

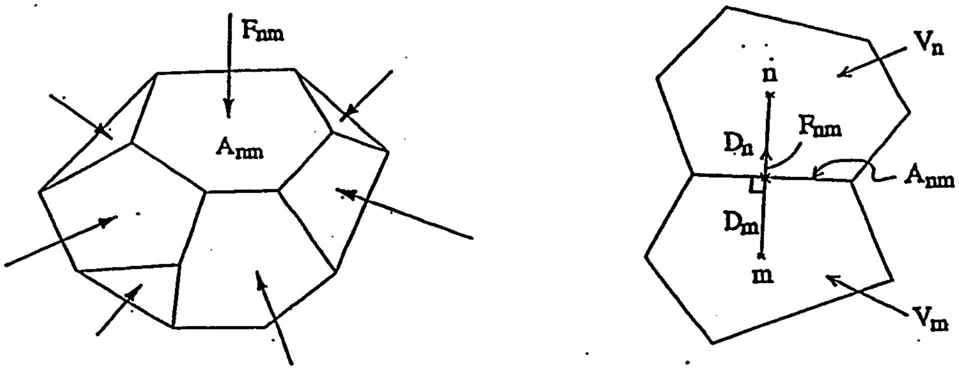

Fig. 2.3.1 Space discretization and geometry for the integral finite difference method (from Pruess, el. al. 1990)

In equation 2.3.3, the flux quantities are evaluated at the interface. Therefore, we require a definition for flow parameters and fluid state at the interface (see Fig. 2.3.1). To evaluate these parameters, several schemes for 'averaging' of the parameters at the interface between grid blocks n and m (for example: interpolation, harmonic weighting, upstream weighting) are used. The choice of the weighting scheme has a strong influence on the results of the numerical simulation. It must be tailored to the physical model of the flow field based on an evaluation of the results using independent observation data.

The discretized form of the binary diffusive flux in phase (β) of component k between elements n and m (equation 2.2.2) is given by:

$$\mathbf{J}^k_{\beta,nm} = -\left(\phi\, S_\beta\, \tau_\beta\right)_{nm} \left(D^k_\beta\right)_{nm} \rho_{\beta,nm} \frac{X^k_{\beta,n} - X^k_{\beta,m}}{D_{nm}} \qquad 2.3.4$$

In the above equation, harmonic weighting is used to compute ($\phi\ S\ \tau$) at the interface, while D_β and ρ_β are averaged between grid blocks n and m.

Substituting equations 2.3.1 and 2.3.2 into the governing equation 2.2.1, results in a set of first-order ordinary differential equations in time

$$\frac{dM^k_n}{dt} = \frac{1}{V_n}\sum_m A_{nm} F^k_{nm} + q^k_n . \qquad 2.3.5$$

Time is discretized as a first order finite difference. The flux and sink/source terms on the right hand side of Equation 2.3.5 are evaluated at the new time level, $t^{l+1} = t^l + \Delta t$, (*l*=*0,1,2,3,...*). This assures the numerical stability needed for an efficient calculation of multiphase flow. Such a treatment of flux terms is known as fully implicit because the fluxes are expressed in terms of the unknown thermodynamic parameters at time level t^{l+1}. These unknowns are only implicitly defined in the resulting equations (Peaceman 1977).

The fully implicit Integral Finite Difference form of the equation (2.1) becomes:

$$M_m^{(k)l+1} - M_m^{(k)l} - \frac{\Delta t^l}{V_m}\left\{\sum_n A_{mn} F_{mn}^{(k)l+1} + V_m q_m^{(k)l+1}\right\} = R_m^{(k)l+1} \cong 0 \qquad 2.3.6$$

Where $\Delta t^l = t^{l+1} - t^l$ is the variable time increment at time level l. $R_m^{(k)l+1}$ is the residual term at time step l+1 that should be minimized after a specified number of iterations, otherwise the time step is reduced.

The total flux term $F_{\beta,mn}^{(k)}$ is given by

$$F_{\beta,mn}^{(k)} = \begin{array}{l} k_{mn}\left(\frac{k_{r\beta}\rho_\beta}{\mu_\beta}\right)_{mn}\left(X_\beta^{(k)}\right)_{mn}\cdot\left[\frac{P_{\beta,m} - P_{\beta,n}}{D_{mn}} - \rho_{\beta,mn} g\cos\theta\right] \\ - \delta_{\beta g} D_{va,mn}\rho_{\beta,mn}\frac{X_{\beta,m}^{(k)} - X_{\beta,n}^{(k)}}{D_{mn}} \end{array} \qquad 2.3.7$$

The entire geometric information of the space discretization in (Equation 2.3.6 and 2.3.7, and Fig. 2.3.1) is provided as the input file 'MESH'. The 'MESH' gives a list of grid block volumes V_n, interface areas A_{nm}, nodal distances D_{nm}, and components g_{nm} of gravitational acceleration along nodal lines. There is no reference to a global system of coordinates, or to the dimensionality of a particular flow problem, in computation of flow. The discretized equations are in fact valid for any arbitrary irregular discretization in one, two or three dimensions, for porous as well as fractured media. However, the accuracy of solution depends on ways in which the various interface parameters in the equations (such as 2.3.3 or 2.3.4), are expressed in terms of average conditions in the grid blocks. A general requirement is that there exists approximate local thermodynamic equilibrium in all grid blocks at all times (Pruess and Narasimhan, 1985). For systems of regular grid blocks referenced to global coordinates, equation 2.3.6 is identical to a conventional finite difference formulation. In the TOUGH2 input the global coordinate systems is only specified, for use in output post-processing, and when MESH processing is required (for example computation of the equivalent dual-continua MESH, Pruess, 1990)

The major task in numeric grid generation is to obtain a grid that provides accurate and consistent flow computation parameters. The grid must also lead to a stable numerical solution and have optimal computation time. Finally the numerical grid should conform with the hydrogeological and conceptual models of the system.

3.0 AMESH input and output

3.1 AMESH input

The file `in' contains the geometrical location of the centers of all nodes in the domain. It also provides the descriptions of the domain boundaries and the controls for the minimum length of the connections. This is done through the (optional) data blocks `*locat*', `*bound*', and `*toler*'. The `*locat*' block provides the name layer-index *xyz* coordinates as well as the thickness of each node in the domain. This is the primary input data for *Amesh*. The `bound' block describes the aerial boundary of the mesh. This is specified as *xy* coordinates in a clockwise direction. The boundary coordinates must be in sequence and they must form a convex polygon around the input domain. If it is not supplied, a rectangular boundary is chosen from the extreme values of the coordinates in the `*locat*' block. In this case, the nodal points with the extreme coordinate would be on the boundary of the mesh. When non-convex boundaries are specified, then *Amesh* may compute geometrical connections exterior to the domain (at the location of concave boundaries) and such connections must be ignored. Alternatively, non-convex domains may be decomposed into convex, sub-domains for purposes of generating accurate grids. These grids can then be combined to form the 'MESH' for the larger domain. The `*toler*' block is included to inform *Amesh* of the length of the minimum interface between elements. This is used to eliminate small interfaces.

Unrecognized text from `*in*' is ignored, except when within a block in which case *Amesh* signal an input error flag and stops further execution. Duplicate `*locat*', and `*bound*' blocks are concatenated, except that scanning terminates after all three blocks ('*locat*', '*bound*' and '*toler*') have been read. *Amesh* understands the (A3, I2) format for element names as used in TOUGH2 and fills in the leading zero on the I2 field. Thus, `e 1 1' and `e 101' are treated as identical by *Amesh*, but `e00 1' and `e 01' are not. *Amesh* is case insensitive for the block names, but it is case sensitive for element names (preserves the case in element names).

The first line of each block should begin with the 5 characters of the block's name. The blocks are terminated by a blank line (actually a line with blank characters in the first 5 spaces). The formats for the lines within each block are:

locat: (a5) block name for node center listed in any order

(a5, i5, 2f20.0, 2f10.0) element, id, *x, y, z* and *t* (Note *x,y,z* and *t* coordinates can be provided in free format).

....

.....

blank line

bound: Block name for enclosing 2d (*x,y*) polygon (in clockwise order)

(2f10.0) *x* and *y*

blank line

toler : Block name for tolerance or minimum connection length

(f10.0)

blank line

Note that :

1.) *xy* coordinates in '*locat*' that are outside '*bound*' will be ignored.

2.) connection interface areas or (line segment, for 2-D) less than '*toler*' will be ignored.

3.) For a 3-D input *id* = layer index. In the input, nodes with the same layer index are in the same layer and will be connected laterally if they are adjacent to each other. Vertical connections are made only between nodes with the same *xy* location (i.e. within the same column).

Examples of the input files are shown in section 3.3

3.2 AMESH output

The output of *Amesh* are three files '*eleme*', '*conne*' and '*segmt*'. The file '*eleme*' contains the element name, material, volume, and the *x*, *y*, and *z* locations of all the element centers as required in TOUGH2. Only a generic space holder for material name ('*rock1*') is written by *Amesh.* A separate post-processor is required to specify node specific material properties based on the layer information and the node location. Because, *Amesh* can generate geometrical grid from input data that have different special dimensions (from few centimeters, to several hundreds of meters apart), the output format should be chosen so as to accurately describe the domain. The format must also fit within the TOUGH2 requirements. The output format can be adjusted by changing the write format in module *'pmesh.c'* of *Amesh*. The format described below corresponds to that currently used in TOUGH2.

The format for *'eleme'* is as follows:

ELEME or **eleme** (a5) Block name for element information

a5,10x,a5,e10.0, 20x, 3f10.0 :*Element, rock type, volume, x, y,* and *z*

....

....

blank line

The file *'conne'* contains the element connection information is required for TOUGH2. The format of 'conne' is as follows:

CONNE or **conne** (a5) :Block name of connection information

a5,a5,18x, i2, 4e10.0 :*Element1, Element2, isot, d1,d2, area, beta*

....

....

blank line

where *element1*, *element2* are the connected element pair; '*isot*' is the direction of permeability factor; *d1* is the distance from the center of *element1* to the interface with *element2*; *d2* is distance from the center of element2 to the interface with *elemet1*; *area* is the interface area (minimum) between *elemnt1* and *element2*; and *beta* is the cosine of the angle formed by the line joining the centers of *element1* and *element2*, measured from the vertical at the center of *element1*.

The two files '*eleme*' and '*conne*' when concatenated provide the input file 'MESH' that defines all the input geometry required by TOUGH2.

The file '*segmt*' contains all the geometrical data required to plot the geometry for all the input nodes that are within the domain defined by '*bound*'. The format for '*segmt*' is:
4f15, i5,2a5 : *x1,y1, x2,y2 index, element1, element2*

When *element1* is on the boundary, *elemnt2* may be simply a space holder for a boundary element. In this case element2 name will be preceded by a star as in '*xxxx'. In the file *segmt*, (*x1*, *y1*) are the coordinates of vertex 1 and (*x2*, *y2*) are the coordinates vertex 2 for the line segment separating *element1* and *element2*

3.3 Examples

3.3.1 Uniform 2D grid spacing in a rectangular domain

3.3.1.1 Input data

Table 3.3.1.1 Input data - Rectangular domain, uniform 2D grid spacing

locat input with x-slope = 0.0 deg and y-slope = 0.0 deg					
Name	**lay_index**	**x-coordinate(m)**	**y-coordinate (m)**	**z-coordinate (m)**	**thickness (m)**
al1 1	101	12.5000000000	12.5000000000	50.00000	10.0000
al1 2	101	12.5000000000	37.5000000000	50.00000	10.0000
al1 3	101	12.5000000000	62.5000000000	50.00000	10.0000
al1 4	101	12.5000000000	87.5000000000	50.00000	10.0000
al1 5	101	37.5000000000	12.5000000000	50.00000	10.0000
al1 6	101	37.5000000000	37.5000000000	50.00000	10.0000
al1 7	101	37.5000000000	62.5000000000	50.00000	10.0000
al1 8	101	37.5000000000	87.5000000000	50.00000	10.0000
al1 9	101	62.5000000000	12.5000000000	50.00000	10.0000
al110	101	62.5000000000	37.5000000000	50.00000	10.0000
al111	101	62.5000000000	62.5000000000	50.00000	10.0000
al112	101	62.5000000000	87.5000000000	50.00000	10.0000
al113	101	87.5000000000	12.5000000000	50.00000	10.0000
al114	101	87.5000000000	37.5000000000	50.00000	10.0000
al115	101	87.5000000000	62.5000000000	50.00000	10.0000
al116	101	87.5000000000	87.5000000000	50.00000	10.0000
bound					
0.0	0.0				
0.0	100.0				
100.0	100.0				
100.0	0.0				
toler					
0.5					
Note: Nodes with same lay_index are within same layer					

3.3.1.2 Output data

Table 3.3.1.2 Element block - Rectangular domain, uniform 2D grid spacing

ELEME					
Name	rock type	volume m^3	x-coordinate (m)	y-coordinate (m)	z-coordinate (m)
a11 1	rock1	6.250e+03	12.500	12.500	50.0000
a11 2	rock1	6.250e+03	12.500	37.500	50.0000
a11 3	rock1	6.250e+03	12.500	62.500	50.0000
a11 4	rock1	6.250e+03	12.500	87.500	50.0000
a11 5	rock1	6.250e+03	37.500	12.500	50.0000
a11 6	rock1	6.250e+03	37.500	37.500	50.0000
a11 7	rock1	6.250e+03	37.500	62.500	50.0000
a11 8	rock1	6.250e+03	37.500	87.500	50.0000
a11 9	rock1	6.250e+03	62.500	12.500	50.0000
a1110	rock1	6.250e+03	62.500	37.500	50.0000
a1111	rock1	6.250e+03	62.500	62.500	50.0000
a1112	rock1	6.250e+03	62.500	87.500	50.0000
a1113	rock1	6.250e+03	87.500	12.500	50.0000
a1114	rock1	6.250e+03	87.500	37.500	50.0000
a1115	rock1	6.250e+03	87.500	62.500	50.0000
a1116	rock1	6.250e+03	87.500	87.500	50.0000

Table 3.3.1.3 Connection block - Rectangular domain, uniform 2D grid spacing

CONNE						
Element1	Element2	isot	d_1 (m)	d_2 (m)	A_{mn} m^2	Cos(β)
a11 1	a11 2	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a11 1	a11 5	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a11 2	a11 3	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a11 2	a11 6	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a11 3	a11 4	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a11 3	a11 7	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a11 4	a11 8	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a11 5	a11 6	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a11 5	a11 9	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a11 6	a11 7	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a11 6	a1110	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a11 7	a11 8	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a11 7	a1111	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a11 8	a1112	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a11 9	a1110	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a11 9	a1113	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a1110	a1111	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a1110	a1114	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a1111	a1112	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a1111	a1115	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a1112	a1116	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a1113	a1114	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a1114	a1115	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00
a1115	a1116	1	1.250e+01	1.250e+01	2.500e+02	0.000e+00

Table 3.3.1.4 Mesh geometry - Rectangular domain, uniform 2D grid spacing

Geometry						
X1	**Y1**	**X2**	**Y2**	**INDEX**	**Element2**	**Element2**
0	0	0	25	0	all 1	* 1
0	25	25	25	1	all 1	all 2
25	25	25	0	1	all 1	all 5
25	0	0	0	0	all 1	* 2
0	25	0	50	0	all 2	* 3
0	50	25	50	1	all 2	all 3
25	50	25	25	1	all 2	all 6
25	25	0	25	0	all 2	all 1
0	50	0	75	0	all 3	* 4
0	75	25	75	1	all 3	all 4
25	75	25	50	1	all 3	all 7
25	50	0	50	0	all 3	all 2
0	100	25	100	0	all 4	* 5
25	100	25	75	1	all 4	all 8
25	75	0	75	0	all 4	all 3
0	75	0	100	0	all 4	* 6
25	0	25	25	0	all 5	all 1
25	25	50	25	1	all 5	all 6
50	25	50	0	1	all 5	all 9
50	0	25	0	0	all 5	* 7
50	50	50	75	0	all11	all 7
50	75	75	75	1	all11	all12
75	75	75	50	1	all11	all15
....						
....						
75	0	75	25	0	all13	all 9
75	25	100	25	1	all13	all14
100	25	100	0	0	all13	* 12
75	25	75	50	0	all14	all10
75	50	100	50	1	all14	all15
100	50	100	25	0	all14	* 13
100	25	75	25	0	all14	all13
100	50	75	50	0	all15	all14
75	50	75	75	0	all15	all11
75	75	100	75	1	all15	all16
100	75	100	50	0	all15	* 14
100	100	100	75	0	all16	* 15
100	75	75	75	0	all16	all15
75	75	75	100	0	all16	all12
75	100	100	100	0	all16	* 16

Note: element2 beginning with '*' implies line segment is along the boundary, else Element1 and Element2 are separated by line segmt X1,Y1 to X2,Y2. INDEX of '0' implies line segment is a duplicate, except when it is on the boundary. In processing this geometry file for plotting, line segments with '0' index should be ignored except when they are on the boundary.

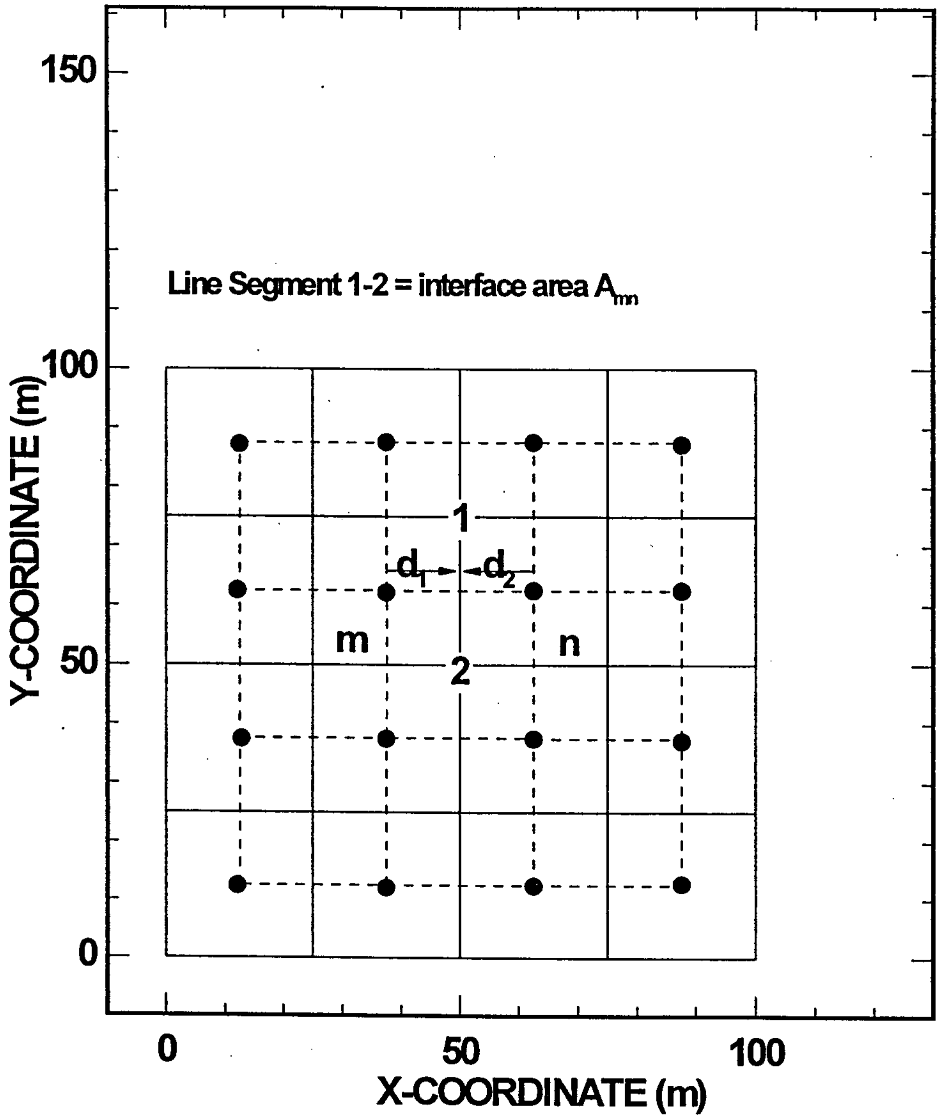

Fig. 3.3.1 Space discretization and geometry for the integral finite difference method grid. Uniform 2D grid spacing in a rectangular domain; Solid lines are Voronoi tessellation space; Dashed lines show valid connections between adjacent elements. Solid circles are the element centers.

3.3.2 Uniform 2D grid spacing in a rhombohedron domain.

3.3.2.1 Input data

Table 3.3.2.1 Input data - rhombohedron domain, uniform 2D grid spacing

locat input with x-slope = 0.0 deg and y-slope = 0.0 deg					
Name	**Lay_index**	**x-coordinate (m)**	**y-coordinate (m)**	**z-coordinate(m)**	**thickness (m)**
al1 1	101	12.5000000000	18.7500000000	50.00000	10.0000
al1 2	101	12.5000000000	43.7500000000	50.00000	10.0000
al1 3	101	12.5000000000	68.7500000000	50.00000	10.0000
al1 4	101	12.5000000000	93.7500000000	50.00000	10.0000
al1 5	101	37.5000000000	31.2500000000	50.00000	10.0000
al1 6	101	37.5000000000	56.2500000000	50.00000	10.0000
al1 7	101	37.5000000000	81.2500000000	50.00000	10.0000
al1 8	101	37.5000000000	106.2500000000	50.00000	10.0000
al1 9	101	62.5000000000	43.7500000000	50.00000	10.0000
al110	101	62.5000000000	68.7500000000	50.00000	10.0000
al111	101	62.5000000000	93.7500000000	50.00000	10.0000
al112	101	62.5000000000	118.7500000000	50.00000	10.0000
al113	101	87.5000000000	56.2500000000	50.00000	10.0000
al114	101	87.5000000000	81.2500000000	50.00000	10.0000
al115	101	87.5000000000	106.2500000000	50.00000	10.0000
al116	101	87.5000000000	131.2500000000	50.00000	10.0000
bound					
0.0	0.0				
0.0	100.0				
100.0	150.0				
100.0	50.0				
toler					
0.5					
Note: Nodes with same lay_index are within same layer					

3.3.2.2 Output data

Table 3.3.2.2 Element block - rhombohedron domain, uniform 2D grid spacing.

ELEME					
Name	**rock type**	**volume m^3**	**x-coordinate (m)**	**y-coordinate (m)**	**z-coordinate (m)**
al1 1	rock1	6.465e+03	12.500	18.750	50.0000
al1 2	rock1	6.250e+03	12.500	43.750	50.0000
al1 3	rock1	6.250e+03	12.500	68.750	50.0000
al1 4	rock1	6.035e+03	12.500	93.750	50.0000
al1 5	rock1	6.250e+03	37.500	31.250	50.0000
al1 6	rock1	6.250e+03	37.500	56.250	50.0000
al1 7	rock1	6.250e+03	37.500	81.250	50.0000
al1 8	rock1	6.250e+03	37.500	106.250	50.0000
al1 9	rock1	6.250e+03	62.500	43.750	50.0000
al110	rock1	6.250e+03	62.500	68.750	50.0000
al111	rock1	6.250e+03	62.500	93.750	50.0000
al112	rock1	6.250e+03	62.500	118.750	50.0000
al113	rock1	6.035e+03	87.500	56.250	50.0000
al114	rock1	6.250e+03	87.500	81.250	50.0000
al115	rock1	6.250e+03	87.500	106.250	50.0000
al116	rock1	6.465e+03	87.500	131.250	50.0000

Table 3.3.2.3 Connection block - rhombohedron domain, uniform 2D grid spacing.

CONNE						
Element1	**Element2**	**isot**	**d_1 (m)**	**d_2 (m)**	**A_{mn} m^2**	**Cos(β)**
al1 1	al1 2	1	1.250e+01	1.250e+01	2.188e+02	0.000e+00
al1 1	al1 5	1	1.398e+01	1.398e+01	1.817e+02	0.000e+00
al1 2	al1 3	1	1.250e+01	1.250e+01	2.188e+02	0.000e+00
al1 2	al1 6	1	1.398e+01	1.398e+01	1.398e+02	0.000e+00
al1 2	al1 5	1	1.398e+01	1.398e+01	1.398e+02	0.000e+00
al1 3	al1 4	1	1.250e+01	1.250e+01	2.188e+02	0.000e+00
al1 3	al1 7	1	1.398e+01	1.398e+01	1.398e+02	0.000e+00
al1 3	al1 6	1	1.398e+01	1.398e+01	1.398e+02	0.000e+00
al1 4	al1 8	1	1.398e+01	1.398e+01	1.817e+02	0.000e+00
al1 4	al1 7	1	1.398e+01	1.398e+01	1.398e+02	0.000e+00
al1 5	al1 6	1	1.250e+01	1.250e+01	1.875e+02	0.000e+00
al1 5	al1 9	1	1.398e+01	1.398e+01	1.817e+02	0.000e+00
al1 6	al1 7	1	1.250e+01	1.250e+01	1.875e+02	0.000e+00
al1 6	al110	1	1.398e+01	1.398e+01	1.398e+02	0.000e+00
al1 6	al1 9	1	1.398e+01	1.398e+01	1.398e+02	0.000e+00
al1 7	al1 8	1	1.250e+01	1.250e+01	1.875e+02	0.000e+00
al1 7	al111	1	1.398e+01	1.398e+01	1.398e+02	0.000e+00
al1 7	al110	1	1.398e+01	1.398e+01	1.398e+02	0.000e+00
al1 8	al112	1	1.398e+01	1.398e+01	1.817e+02	0.000e+00
al1 8	al111	1	1.398e+01	1.398e+01	1.398e+02	0.000e+00
al1 9	al110	1	1.250e+01	1.250e+01	1.875e+02	0.000e+00
al1 9	al113	1	1.398e+01	1.398e+01	1.817e+02	0.000e+00
al110	al111	1	1.250e+01	1.250e+01	1.875e+02	0.000e+00
al110	al114	1	1.398e+01	1.398e+01	1.398e+02	0.000e+00
al110	al113	1	1.398e+01	1.398e+01	1.398e+02	0.000e+00
al111	al112	1	1.250e+01	1.250e+01	1.875e+02	0.000e+00
al111	al115	1	1.398e+01	1.398e+01	1.398e+02	0.000e+00
al111	al114	1	1.398e+01	1.398e+01	1.398e+02	0.000e+00
al112	al116	1	1.398e+01	1.398e+01	1.817e+02	0.000e+00
al112	al115	1	1.398e+01	1.398e+01	1.398e+02	0.000e+00
al113	al114	1	1.250e+01	1.250e+01	2.188e+02	0.000e+00
al114	al115	1	1.250e+01	1.250e+01	2.188e+02	0.000e+00
al115	al116	1	1.250e+01	1.250e+01	2.188e+02	0.000e+00

Table 3.3.2.4 Mesh geometry - rhombohedron domain, uniform 2D grid spacing.

Geometry						
X1 (m)	**Y1 (m)**	**X2 (m)**	**Y2 (m)**	**Index**	**Element2**	**Element2**
0	0	0	31.25	0	al1 1	* 1
0	31.25	21.875	31.25	1	al1 1	al1 2
21.875	31.25	30	15	1	al1 1	al1 5
30	15	0	0	0	al1 1	* 2
0	31.25	0	56.25	0	al1 2	* 3
0	56.25	21.875	56.25	1	al1 2	al1 3
21.875	56.25	28.125	43.75	1	al1 2	al1 6
28.125	43.75	21.875	31.25	1	al1 2	al1 5
21.875	31.25	0	31.25	0	al1 2	al1 1
0	56.25	0	81.25	0	al1 3	* 4
0	81.25	21.875	81.25	1	al1 3	al1 4
21.875	81.25	28.125	68.75	1	al1 3	al1 7
28.125	68.75	21.875	56.25	1	al1 3	al1 6
21.875	56.25	0	56.25	0	al1 3	al1 2
....						
....						
100	50	80	40	0	al113	* 11
80	40	71.875	56.25	0	al113	al1 9
71.875	56.25	78.125	68.75	0	al113	al110
78.125	68.75	100	68.75	1	al113	al114
100	68.75	100	50	0	al113	* 12
71.875	81.25	78.125	93.75	0	al114	al111
78.125	93.75	100	93.75	1	al114	al115
100	93.75	100	68.75	0	al114	* 13
100	68.75	78.125	68.75	0	al114	al113
78.125	68.75	71.875	81.25	0	al114	al110
100	93.75	78.125	93.75	0	al115	al114
78.125	93.75	71.875	106.25	0	al115	al111
71.875	106.25	78.125	118.75	0	al115	al112
78.125	118.75	100	118.75	1	al115	al116
100	118.75	100	93.75	0	al115	* 14
100	150	100	118.75	0	al116	* 15
100	118.75	78.125	118.75	0	al116	al115
78.125	118.75	70	135	0	al116	al112
70	135	100	150	0	al116	* 16
Note: element2 beginning with '*' implies line segment is along the boundary, else Element1 and Element2 are separated by line segmt X1,Y1 to X2,Y2. Index of '0' implies line segment is a duplicate, except when it is on the boundary. In processing this geometry file for plotting, line segments with '0' index should be ignored except when they are on the boundary.						

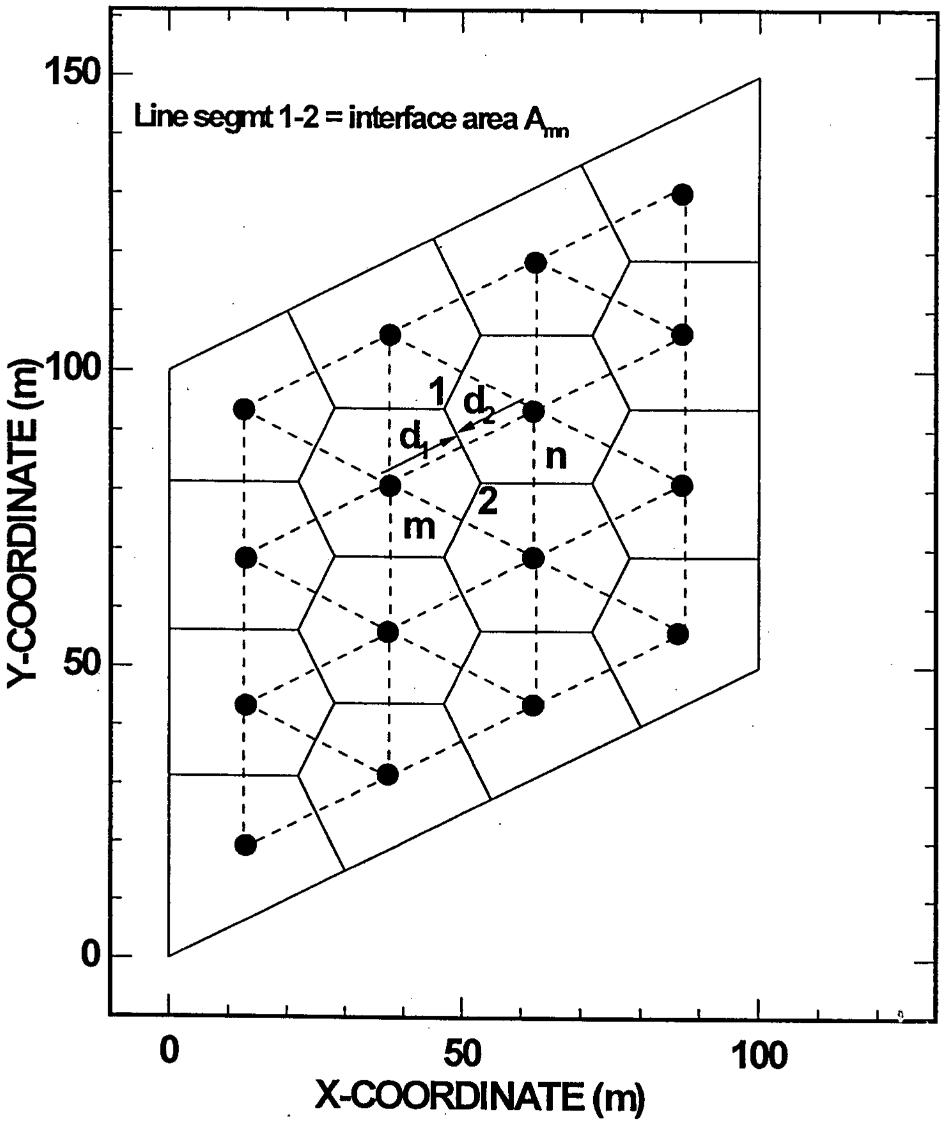

Fig. 3.3.2 Space discretization and geometry for the integral finite difference method grid. Uniform 2D grid spacing in a rhombohedron domain; Solid lines are Voronoi tessellation space; Dashed lines show valid connections between adjacent elements. Solid circles are the element centers.

3.3.3 Uniform 3D grid spacing in a rectangular domain

3.3.3.1 Input data

Table 3.3.3.1 Input data - rectangular domain, uniform 3D grid spacing

locat input with x-slope = 0.0 deg and y-slope = 0.0 deg					
Name	**Lay_index**	**x-coordinate (m)**	**y-coordinate (m)**	**z-coordinate(m)**	**thickness (m)**
a11 1	101	12.5000000000	12.5000000000	50.00	10.00
a21 1	102	12.5000000000	12.5000000000	40.00	10.00
a31 1	103	12.5000000000	12.5000000000	30.00	10.00
a41 1	104	12.5000000000	12.5000000000	20.00	10.00
a11 2	101	12.5000000000	37.5000000000	50.00	10.00
a21 2	102	12.5000000000	37.5000000000	40.00	10.00
a31 2	103	12.5000000000	37.5000000000	30.00	10.00
a41 2	104	12.5000000000	37.5000000000	20.00	10.00
a11 3	101	12.5000000000	62.5000000000	50.00	10.00
a21 3	102	12.5000000000	62.5000000000	40.00	10.00
a31 3	103	12.5000000000	62.5000000000	30.00	10.00
a41 3	104	12.5000000000	62.5000000000	20.00	10.00
a11 4	101	12.5000000000	87.5000000000	50.00	10.00
a21 4	102	12.5000000000	87.5000000000	40.00	10.00
a31 4	103	12.5000000000	87.5000000000	30.00	10.00
a41 4	104	12.5000000000	87.5000000000	20.00	10.00
a11 5	101	37.5000000000	12.5000000000	50.00	10.00
a21 5	102	37.5000000000	12.5000000000	40.00	10.00
a31 5	103	37.5000000000	12.5000000000	30.00	10.00
a41 5	104	37.5000000000	12.5000000000	20.00	10.00
a11 6	101	37.5000000000	37.5000000000	50.00	10.00
a21 6	102	37.5000000000	37.5000000000	40.00	10.00
a31 6	103	37.5000000000	37.5000000000	30.00	10.00
a41 6	104	37.5000000000	37.5000000000	20.00	10.00
a11 7	101	37.5000000000	62.5000000000	50.00	10.00
a21 7	102	37.5000000000	62.5000000000	40.00	10.00
a31 7	103	37.5000000000	62.5000000000	30.00	10.00
a41 7	104	37.5000000000	62.5000000000	20.00	10.00
a11 8	101	37.5000000000	87.5000000000	50.00	10.00
a21 8	102	37.5000000000	87.5000000000	40.00	10.00
a31 8	103	37.5000000000	87.5000000000	30.00	10.00
a41 8	104	37.5000000000	87.5000000000	20.00	10.00
a11 9	101	62.5000000000	12.5000000000	50.00	10.00
a21 9	102	62.5000000000	12.5000000000	40.00	10.00
a31 9	103	62.5000000000	12.5000000000	30.00	10.00
a41 9	104	62.5000000000	12.5000000000	20.00	10.00
a1110	101	62.5000000000	37.5000000000	50.00	10.00
a2110	102	62.5000000000	37.5000000000	40.00	10.00
a3110	103	62.5000000000	37.5000000000	30.00	10.00
a4110	104	62.5000000000	37.5000000000	20.00	10.00
a1111	101	62.5000000000	62.5000000000	50.00	10.00

a2111	102	62.5000000000	62.5000000000	40.00	10.00
a3111	103	62.5000000000	62.5000000000	30.00	10.00
a4111	104	62.5000000000	62.5000000000	20.00	10.00
a1112	101	62.5000000000	87.5000000000	50.00	10.00
a2112	102	62.5000000000	87.5000000000	40.00	10.00
a3112	103	62.5000000000	87.5000000000	30.00	10.00
a4112	104	62.5000000000	87.5000000000	20.00	10.00
a1113	101	87.5000000000	12.5000000000	50.00	10.00
a2113	102	87.5000000000	12.5000000000	40.00	10.00
a3113	103	87.5000000000	12.5000000000	30.00	10.00
a4113	104	87.5000000000	12.5000000000	20.00	10.00
a1114	101	87.5000000000	37.5000000000	50.00	10.00
a2114	102	87.5000000000	37.5000000000	40.00	10.00
a3114	103	87.5000000000	37.5000000000	30.00	10.00
a4114	104	87.5000000000	37.5000000000	20.00	10.00
a1115	101	87.5000000000	62.5000000000	50.00	10.00
a2115	102	87.5000000000	62.5000000000	40.00	10.00
a3115	103	87.5000000000	62.5000000000	30.00	10.00
a4115	104	87.5000000000	62.5000000000	20.00	10.00
a1116	101	87.5000000000	87.5000000000	50.00	10.00
a2116	102	87.5000000000	87.5000000000	40.00	10.00
a3116	103	87.5000000000	87.5000000000	30.00	10.00
a4116	104	87.5000000000	87.5000000000	20.00	10.00
bound					
0.0	0.0				
0.0	100.0				
100.0	100.0				
100.0	0.0				
toler					
0.5					
Note: Nodes with same lay_index are within same layer					

3.3.3.2 Output data

Table 3.3.3.2 Element block - rectangular domain, uniform 3D grid spacing

ELEME					
Name	rock type	volume m^3	x-coordinate (m)	y-coordinate (m)	z-coordinate (m)
a11 1	rock1	6.250e+03	12.500	12.500	50.0000
a21 1	rock1	6.250e+03	12.500	12.500	40.0000
a31 1	rock1	6.250e+03	12.500	12.500	30.0000
a41 1	rock1	6.250e+03	12.500	12.500	20.0000
a11 2	rock1	6.250e+03	12.500	37.500	50.0000
a21 2	rock1	6.250e+03	12.500	37.500	40.0000
a31 2	rock1	6.250e+03	12.500	37.500	30.0000
a41 2	rock1	6.250e+03	12.500	37.500	20.0000
a11 3	rock1	6.250e+03	12.500	62.500	50.0000
a21 3	rock1	6.250e+03	12.500	62.500	40.0000
a31 3	rock1	6.250e+03	12.500	62.500	30.0000
a41 3	rock1	6.250e+03	12.500	62.500	20.0000
a11 4	rock1	6.250e+03	12.500	87.500	50.0000
a21 4	rock1	6.250e+03	12.500	87.500	40.0000
a31 4	rock1	6.250e+03	12.500	87.500	30.0000
a41 4	rock1	6.250e+03	12.500	87.500	20.0000
a11 5	rock1	6.250e+03	37.500	12.500	50.0000
a21 5	rock1	6.250e+03	37.500	12.500	40.0000
a31 5	rock1	6.250e+03	37.500	12.500	30.0000
a41 5	rock1	6.250e+03	37.500	12.500	20.0000
a11 6	rock1	6.250e+03	37.500	37.500	50.0000
a21 6	rock1	6.250e+03	37.500	37.500	40.0000
a31 6	rock1	6.250e+03	37.500	37.500	30.0000
a41 6	rock1	6.250e+03	37.500	37.500	20.0000
....					
....					
a1114	rock1	6.250e+03	87.500	37.500	50.0000
a2114	rock1	6.250e+03	87.500	37.500	40.0000
a3114	rock1	6.250e+03	87.500	37.500	30.0000
a4114	rock1	6.250e+03	87.500	37.500	20.0000
a1115	rock1	6.250e+03	87.500	62.500	50.0000
a2115	rock1	6.250e+03	87.500	62.500	40.0000
a3115	rock1	6.250e+03	87.500	62.500	30.0000
a4115	rock1	6.250e+03	87.500	62.500	20.0000
a1116	rock1	6.250e+03	87.500	87.500	50.0000
a2116	rock1	6.250e+03	87.500	87.500	40.0000
a3116	rock1	6.250e+03	87.500	87.500	30.0000
a4116	rock1	6.250e+03	87.500	87.500	20.0000

Table 3.3.3.3 Connection block - rectangular domain, uniform 3D grid spacing

CONNE						
Element1	**Element2**	**isot**	**d_1 (m)**	**d_2 (m)**	**A_{mn} m^2**	**Cos(β)**
a11 1	a11 2	1	1.250e+01	1.250e+01	2.500e+02	0.0
a11 1	a11 5	1	1.250e+01	1.250e+01	2.500e+02	0.0
a21 1	a21 2	1	1.250e+01	1.250e+01	2.500e+02	0.0
a21 1	a21 5	1	1.250e+01	1.250e+01	2.500e+02	0.0
a21 1	a11 1	3	5.000e+00	5.000e+00	6.250e+02	1.0
a31 1	a31 2	1	1.250e+01	1.250e+01	2.500e+02	0.0
a31 1	a31 5	1	1.250e+01	1.250e+01	2.500e+02	0.0
a31 1	a21 1	3	5.000e+00	5.000e+00	6.250e+02	1.0
a41 1	a41 2	1	1.250e+01	1.250e+01	2.500e+02	0.0
a41 1	a41 5	1	1.250e+01	1.250e+01	2.500e+02	0.0
a41 1	a31 1	3	5.000e+00	5.000e+00	6.250e+02	1.0
a11 2	a11 3	1	1.250e+01	1.250e+01	2.500e+02	0.0
a11 2	a11 6	1	1.250e+01	1.250e+01	2.500e+02	0.0
a21 2	a21 3	1	1.250e+01	1.250e+01	2.500e+02	0.0
a21 2	a21 6	1	1.250e+01	1.250e+01	2.500e+02	0.0
a21 2	a11 2	3	5.000e+00	5.000e+00	6.250e+02	1.0
a31 2	a31 3	1	1.250e+01	1.250e+01	2.500e+02	0.0
a31 2	a31 6	1	1.250e+01	1.250e+01	2.500e+02	0.0
a31 2	a21 2	3	5.000e+00	5.000e+00	6.250e+02	1.0
a41 2	a41 3	1	1.250e+01	1.250e+01	2.500e+02	0.0
a41 2	a41 6	1	1.250e+01	1.250e+01	2.500e+02	0.0
a41 2	a31 2	3	5.000e+00	5.000e+00	6.250e+02	1.0
a11 3	a11 4	1	1.250e+01	1.250e+01	2.500e+02	0.0
a11 3	a11 7	1	1.250e+01	1.250e+01	2.500e+02	0.0
a21 3	a21 4	1	1.250e+01	1.250e+01	2.500e+02	0.0
....						
....						
a3114	a3115	1	1.250e+01	1.250e+01	2.500e+02	0.0
a3114	a2114	3	5.000e+00	5.000e+00	6.250e+02	1.0
a4114	a4115	1	1.250e+01	1.250e+01	2.500e+02	0.0
a4114	a3114	3	5.000e+00	5.000e+00	6.250e+02	1.0
a1115	a1116	1	1.250e+01	1.250e+01	2.500e+02	0.0
a2115	a2116	1	1.250e+01	1.250e+01	2.500e+02	0.0
a2115	a1115	3	5.000e+00	5.000e+00	6.250e+02	1.0
a3115	a3116	1	1.250e+01	1.250e+01	2.500e+02	0.0
a3115	a2115	3	5.000e+00	5.000e+00	6.250e+02	1.0
a4115	a4116	1	1.250e+01	1.250e+01	2.500e+02	0.0
a4115	a3115	3	5.000e+00	5.000e+00	6.250e+02	1.0
a2116	a1116	3	5.000e+00	5.000e+00	6.250e+02	1.0
a3116	a2116	3	5.000e+00	5.000e+00	6.250e+02	1.0
a4116	a3116	3	5.000e+00	5.000e+00	6.250e+02	1.0

Table 3.3.3.4 Mesh geometry - rectangular domain, uniform 3D grid spacing.

Geometry						
X1 (m)	**Y1(m)**	**X2(m)**	**Y2(m)**	**Index**	**Element1**	**Element2**
0	0	0	25	0	a11 1	* 1
0	25	25	25	1	a11 1	a11 2
25	25	25	0	1	a11 1	a11 5
25	0	0	0	0	a11 1	* 2
0	0	0	25	0	a21 1	* 3
0	25	25	25	1	a21 1	a21 2
25	25	25	0	1	a21 1	a21 5
25	0	0	0	0	a21 1	* 4
0	0	0	25	0	a31 1	* 5
0	25	25	25	1	a31 1	a31 2
25	25	25	0	1	a31 1	a31 5
25	0	0	0	0	a31 1	* 6
0	0	0	25	0	a41 1	* 7
0	25	25	25	1	a41 1	a41 2
25	25	25	0	1	a41 1	a41 5
25	0	0	0	0	a41 1	* 8
0	25	0	50	0	a11 2	* 9
0	50	25	50	1	a11 2	a11 3
25	50	25	25	1	a11 2	a11 6
....						
....						
75	50	75	75	0	a4115	a4111
75	75	100	75	1	a4115	a4116
100	75	100	50	0	a4115	* 56
100	100	100	75	0	a1116	* 57
100	75	75	75	0	a1116	a1115
75	75	75	100	0	a1116	a1112
75	100	100	100	0	a1116	* 58
100	100	100	75	0	a2116	* 59
100	75	75	75	0	a2116	a2115
75	75	75	100	0	a2116	a2112
75	100	100	100	0	a2116	* 60
100	100	100	75	0	a3116	* 61
100	75	75	75	0	a3116	a3115
75	75	75	100	0	a3116	a3112
75	100	100	100	0	a3116	* 62
100	100	100	75	0	a4116	* 63
100	75	75	75	0	a4116	a4115
75	75	75	100	0	a4116	a4112
75	100	100	100	0	a4116	* 64

Note: element2 beginning with '*' implies line segment is along the boundary, else Element1 and Element2 are separated by line segmt X1,Y1 to X2,Y2. INDEX of '0' implies line segment is a duplicate, except when it is on the boundary. In processing this geometry file for plotting, line segments with '0' index should be ignored except when they are on the boundary.

3.3.4 Uniform 3D grid spacing in a rectangular domain, layers slope 10° in x-direction.

3.3.4.1 Input data

Table 3.3.4.1 Input data - rectangular domain, uniform 3D grid spacing, 10° slope in x-direction.

locat input with x-slope = 10.0 deg and y-slope = 0.0 deg					
Name	**Lay_index**	**x-coordinate (m)**	**y-coordinate (m)**	**z-coordinate(m)**	**thickness (m)**
a11 1	101	12.5000000000	12.5000000000	50.00000	10.0000
a21 1	102	12.5000000000	12.5000000000	40.00000	10.0000
a31 1	103	12.5000000000	12.5000000000	30.00000	10.0000
a41 1	104	12.5000000000	12.5000000000	20.00000	10.0000
a11 2	101	12.5000000000	37.5000000000	50.00000	10.0000
a21 2	102	12.5000000000	37.5000000000	40.00000	10.0000
a31 2	103	12.5000000000	37.5000000000	30.00000	10.0000
a41 2	104	12.5000000000	37.5000000000	20.00000	10.0000
a11 3	101	12.5000000000	62.5000000000	50.00000	10.0000
a21 3	102	12.5000000000	62.5000000000	40.00000	10.0000
a31 3	103	12.5000000000	62.5000000000	30.00000	10.0000
a41 3	104	12.5000000000	62.5000000000	20.00000	10.0000
a11 4	101	12.5000000000	87.5000000000	50.00000	10.0000
a21 4	102	12.5000000000	87.5000000000	40.00000	10.0000
a31 4	103	12.5000000000	87.5000000000	30.00000	10.0000
a41 4	104	12.5000000000	87.5000000000	20.00000	10.0000
a11 5	101	37.5000000000	12.5000000000	45.59183	10.0000
a21 5	102	37.5000000000	12.5000000000	35.59183	10.0000
a31 5	103	37.5000000000	12.5000000000	25.59183	10.0000
a41 5	104	37.5000000000	12.5000000000	15.59183	10.0000
a11 6	101	37.5000000000	37.5000000000	45.59183	10.0000
a21 6	102	37.5000000000	37.5000000000	35.59183	10.0000
a31 6	103	37.5000000000	37.5000000000	25.59183	10.0000
a41 6	104	37.5000000000	37.5000000000	15.59183	10.0000
a11 7	101	37.5000000000	62.5000000000	45.59183	10.0000
a21 7	102	37.5000000000	62.5000000000	35.59183	10.0000
a31 7	103	37.5000000000	62.5000000000	25.59183	10.0000
a41 7	104	37.5000000000	62.5000000000	15.59183	10.0000
a11 8	101	37.5000000000	87.5000000000	45.59183	10.0000
a21 8	102	37.5000000000	87.5000000000	35.59183	10.0000
a31 8	103	37.5000000000	87.5000000000	25.59183	10.0000
a41 8	104	37.5000000000	87.5000000000	15.59183	10.0000
a11 9	101	62.5000000000	12.5000000000	41.18365	10.0000
a21 9	102	62.5000000000	12.5000000000	31.18365	10.0000
a31 9	103	62.5000000000	12.5000000000	21.18365	10.0000
a41 9	104	62.5000000000	12.5000000000	11.18365	10.0000
a1110	101	62.5000000000	37.5000000000	41.18365	10.0000
a2110	102	62.5000000000	37.5000000000	31.18365	10.0000

a3110	103	62.5000000000	37.5000000000	21.18365	10.0000
a4110	104	62.5000000000	37.5000000000	11.18365	10.0000
a1111	101	62.5000000000	62.5000000000	41.18365	10.0000
a2111	102	62.5000000000	62.5000000000	31.18365	10.0000
a3111	103	62.5000000000	62.5000000000	21.18365	10.0000
a4111	104	62.5000000000	62.5000000000	11.18365	10.0000
a1112	101	62.5000000000	87.5000000000	41.18365	10.0000
a2112	102	62.5000000000	87.5000000000	31.18365	10.0000
a3112	103	62.5000000000	87.5000000000	21.18365	10.0000
a4112	104	62.5000000000	87.5000000000	11.18365	10.0000
a1113	101	87.5000000000	12.5000000000	36.77548	10.0000
a2113	102	87.5000000000	12.5000000000	26.77548	10.0000
a3113	103	87.5000000000	12.5000000000	16.77548	10.0000
a4113	104	87.5000000000	12.5000000000	6.77548	10.0000
a1114	101	87.5000000000	37.5000000000	36.77548	10.0000
a2114	102	87.5000000000	37.5000000000	26.77548	10.0000
a3114	103	87.5000000000	37.5000000000	16.77548	10.0000
a4114	104	87.5000000000	37.5000000000	6.77548	10.0000
a1115	101	87.5000000000	62.5000000000	36.77548	10.0000
a2115	102	87.5000000000	62.5000000000	26.77548	10.0000
a3115	103	87.5000000000	62.5000000000	16.77548	10.0000
a4115	104	87.5000000000	62.5000000000	6.77548	10.0000
a1116	101	87.5000000000	87.5000000000	36.77548	10.0000
a2116	102	87.5000000000	87.5000000000	26.77548	10.0000
a3116	103	87.5000000000	87.5000000000	16.77548	10.0000
a4116	104	87.5000000000	87.5000000000	6.77548	10.0000
bound					
0.0	0.0				
0.0	100.0				
100.0	100.0				
toler					
0.5					
Note: Nodes with same lay_index are within same layer					

3.3.4.2 Output data

Table 3.3.4.2 Element block - rectangular domain, uniform 3D grid spacing, 10° slope in x-direction.

ELEME					
Name	rock type	volume m^3	x-coordinate (m)	y-coordinate (m)	z-coordinate (m)
a11 1	rock1	6.250e+03	12.500	12.500	50.0000
a21 1	rock1	6.250e+03	12.500	12.500	40.0000
a31 1	rock1	6.250e+03	12.500	12.500	30.0000
a41 1	rock1	6.250e+03	12.500	12.500	20.0000
a11 2	rock1	6.250e+03	12.500	37.500	50.0000
a21 2	rock1	6.250e+03	12.500	37.500	40.0000
a31 2	rock1	6.250e+03	12.500	37.500	30.0000
a41 2	rock1	6.250e+03	12.500	37.500	20.0000
a11 3	rock1	6.250e+03	12.500	62.500	50.0000
a21 3	rock1	6.250e+03	12.500	62.500	40.0000
a31 3	rock1	6.250e+03	12.500	62.500	30.0000
a41 3	rock1	6.250e+03	12.500	62.500	20.0000
a11 4	rock1	6.250e+03	12.500	87.500	50.0000
a21 4	rock1	6.250e+03	12.500	87.500	40.0000
a31 4	rock1	6.250e+03	12.500	87.500	30.0000
a41 4	rock1	6.250e+03	12.500	87.500	20.0000
a11 5	rock1	6.250e+03	37.500	12.500	45.5918
a21 5	rock1	6.250e+03	37.500	12.500	35.5918
a31 5	rock1	6.250e+03	37.500	12.500	25.5918
a41 5	rock1	6.250e+03	37.500	12.500	15.5918
a11 6	rock1	6.250e+03	37.500	37.500	45.5918
a21 6	rock1	6.250e+03	37.500	37.500	35.5918
a31 6	rock1	6.250e+03	37.500	37.500	25.5918
a41 6	rock1	6.250e+03	37.500	37.500	15.5918
....					
....					
a1114	rock1	6.250e+03	87.500	37.500	36.7755
a2114	rock1	6.250e+03	87.500	37.500	26.7755
a3114	rock1	6.250e+03	87.500	37.500	16.7755
a4114	rock1	6.250e+03	87.500	37.500	6.7755
a1115	rock1	6.250e+03	87.500	62.500	36.7755
a2115	rock1	6.250e+03	87.500	62.500	26.7755
a3115	rock1	6.250e+03	87.500	62.500	16.7755
a4115	rock1	6.250e+03	87.500	62.500	6.7755
a1116	rock1	6.250e+03	87.500	87.500	36.7755
a2116	rock1	6.250e+03	87.500	87.500	26.7755
a3116	rock1	6.250e+03	87.500	87.500	16.7755
a4116	rock1	6.250e+03	87.500	87.500	6.7755

Table 3.3.4.3 Connection block - rectangular domain, uniform 3D grid spacing, 10° slope in x-direction.

CONNE						
Element1	**Element2**	**isot**	**d_1 (m)**	**d_2 (m)**	**A_{mn} m^2**	**Cos(β)**
a11 1	a11 2	1	1.250e+01	1.250e+01	2.500e+02	0.0
a11 1	a11 5	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a21 1	a21 2	1	1.250e+01	1.250e+01	2.500e+02	0.0
a21 1	a21 5	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a21 1	a11 1	3	5.000e+00	5.000e+00	6.250e+02	1.0
a31 1	a31 2	1	1.250e+01	1.250e+01	2.500e+02	0.0
a31 1	a31 5	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a31 1	a21 1	3	5.000e+00	5.000e+00	6.250e+02	1.0
a41 1	a41 2	1	1.250e+01	1.250e+01	2.500e+02	0.0
a41 1	a41 5	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a41 1	a31 1	3	5.000e+00	5.000e+00	6.250e+02	1.0
a11 2	a11 3	1	1.250e+01	1.250e+01	2.500e+02	0.0
a11 2	a11 6	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a21 2	a21 3	1	1.250e+01	1.250e+01	2.500e+02	0.0
a21 2	a21 6	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a21 2	a11 2	3	5.000e+00	5.000e+00	6.250e+02	1.0
a31 2	a31 3	1	1.250e+01	1.250e+01	2.500e+02	0.0
a31 2	a31 6	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a31 2	a21 2	3	5.000e+00	5.000e+00	6.250e+02	1.0
a41 2	a41 3	1	1.250e+01	1.250e+01	2.500e+02	0.0
a41 2	a41 6	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a41 2	a31 2	3	5.000e+00	5.000e+00	6.250e+02	1.0
a11 3	a11 4	1	1.250e+01	1.250e+01	2.500e+02	0.0
a11 3	a11 7	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a21 3	a21 4	1	1.250e+01	1.250e+01	2.500e+02	0.0
....						
....						
a3114	a2114	3	5.000e+00	5.000e+00	6.250e+02	1.0
a4114	a4115	1	1.250e+01	1.250e+01	2.500e+02	0.0
a4114	a3114	3	5.000e+00	5.000e+00	6.250e+02	1.0
a1115	a1116	1	1.250e+01	1.250e+01	2.500e+02	0.0
a2115	a2116	1	1.250e+01	1.250e+01	2.500e+02	0.0
a2115	a1115	3	5.000e+00	5.000e+00	6.250e+02	1.0
a3115	a3116	1	1.250e+01	1.250e+01	2.500e+02	0.0
a3115	a2115	3	5.000e+00	5.000e+00	6.250e+02	1.0
a4115	a4116	1	1.250e+01	1.250e+01	2.500e+02	0.0
a4115	a3115	3	5.000e+00	5.000e+00	6.250e+02	1.0
a2116	a1116	3	5.000e+00	5.000e+00	6.250e+02	1.0
a3116	a2116	3	5.000e+00	5.000e+00	6.250e+02	1.0
a4116	a3116	3	5.000e+00	5.000e+00	6.250e+02	1.0

Table 3.3.4.4 Mesh geometry - rectangular domain, uniform 3D grid spacing, 10° slope in x-direction.

X1(m)	Y1(m)	X2(m)	Y2(m)	Index	Element1	Element2
0	0	0	25	0	a11 1	* 1
0	25	25	25	1	a11 1	a11 2
25	25	25	0	1	a11 1	a11 5
25	0	0	0	0	a11 1	* 2
0	0	0	25	0	a21 1	* 3
0	25	25	25	1	a21 1	a21 2
25	25	25	0	1	a21 1	a21 5
25	0	0	0	0	a21 1	* 4
0	0	0	25	0	a31 1	* 5
0	25	25	25	1	a31 1	a31 2
25	25	25	0	1	a31 1	a31 5
25	0	0	0	0	a31 1	* 6
0	0	0	25	0	a41 1	* 7
0	25	25	25	1	a41 1	a41 2
25	25	25	0	1	a41 1	a41 5
25	0	0	0	0	a41 1	* 8
0	25	0	50	0	a11 2	* 9
0	50	25	50	1	a11 2	a11 3
25	50	25	25	1	a11 2	a11 6
....						
....						
100	50	75	50	0	a4115	a4114
75	50	75	75	0	a4115	a4111
75	75	100	75	1	a4115	a4116
100	75	100	50	0	a4115	* 56
100	100	100	75	0	a1116	* 57
100	75	75	75	0	a1116	a1115
75	75	75	100	0	a1116	a1112
75	100	100	100	0	a1116	* 58
100	100	100	75	0	a2116	* 59
100	75	75	75	0	a2116	a2115
75	75	75	100	0	a2116	a2112
75	100	100	100	0	a2116	* 60
100	100	100	75	0	a3116	* 61
100	75	75	75	0	a3116	a3115
75	75	75	100	0	a3116	a3112
75	100	100	100	0	a3116	* 62
100	100	100	75	0	a4116	* 63
100	75	75	75	0	a4116	a4115
75	75	75	100	0	a4116	a4112
75	100	100	100	0	a4116	* 64

Note: element2 beginning with '*' implies line segment is along the boundary, else Element1 and Element2 are separated by line segmt X1,Y1 to X2,Y2. INDEX of '0' implies line segment is a duplicate, except when it is on the boundary. In processing this geometry file for plotting, line segments with '0' index should be ignored except when they are on the boundary.

3.3.5 Uniform 3D grid spacing in a rectangular domain, layers slope 10° in x-direction and 10° in y-direction

3.3.5.1 Input data

Table 1.5.1 Input data - rectangular domain, uniform 3D grid spacing, slope =10° in x-direction and 10° in y-direction

locat input with x-slope = 10.0 deg and y-slope = 10.0 deg					
Name	**Lay_index**	**x-coordinate (m)**	**y-coordinate (m)**	**z-coordinate(m)**	**thickness (m)**
a11 1	101	12.5000000000	12.5000000000	50.00000	10.0000
a21 1	102	12.5000000000	12.5000000000	40.00000	10.0000
a31 1	103	12.5000000000	12.5000000000	30.00000	10.0000
a41 1	104	12.5000000000	12.5000000000	20.00000	10.0000
a11 2	101	12.5000000000	37.5000000000	45.59183	10.0000
a21 2	102	12.5000000000	37.5000000000	35.59183	10.0000
a31 2	103	12.5000000000	37.5000000000	25.59183	10.0000
a41 2	104	12.5000000000	37.5000000000	15.59183	10.0000
a11 3	101	12.5000000000	62.5000000000	41.18365	10.0000
a21 3	102	12.5000000000	62.5000000000	31.18365	10.0000
a31 3	103	12.5000000000	62.5000000000	21.18365	10.0000
a41 3	104	12.5000000000	62.5000000000	11.18365	10.0000
a11 4	101	12.5000000000	87.5000000000	36.77548	10.0000
a21 4	102	12.5000000000	87.5000000000	26.77548	10.0000
a31 4	103	12.5000000000	87.5000000000	16.77548	10.0000
a41 4	104	12.5000000000	87.5000000000	6.77548	10.0000
a11 5	101	37.5000000000	12.5000000000	45.59183	10.0000
a21 5	102	37.5000000000	12.5000000000	35.59183	10.0000
a31 5	103	37.5000000000	12.5000000000	25.59183	10.0000
a41 5	104	37.5000000000	12.5000000000	15.59183	10.0000
a11 6	101	37.5000000000	37.5000000000	41.18365	10.0000
a21 6	102	37.5000000000	37.5000000000	31.18365	10.0000
a31 6	103	37.5000000000	37.5000000000	21.18365	10.0000
a41 6	104	37.5000000000	37.5000000000	11.18365	10.0000
a11 7	101	37.5000000000	62.5000000000	36.77548	10.0000
a21 7	102	37.5000000000	62.5000000000	26.77548	10.0000
a31 7	103	37.5000000000	62.5000000000	16.77548	10.0000
a41 7	104	37.5000000000	62.5000000000	6.77548	10.0000
a11 8	101	37.5000000000	87.5000000000	32.36730	10.0000
a21 8	102	37.5000000000	87.5000000000	22.36730	10.0000
a31 8	103	37.5000000000	87.5000000000	12.36730	10.0000
a41 8	104	37.5000000000	87.5000000000	2.36730	10.0000
a11 9	101	62.5000000000	12.5000000000	41.18365	10.0000
a21 9	102	62.5000000000	12.5000000000	31.18365	10.0000
a31 9	103	62.5000000000	12.5000000000	21.18365	10.0000
a41 9	104	62.5000000000	12.5000000000	11.18365	10.0000
a1110	101	62.5000000000	37.5000000000	36.77548	10.0000

a2110	102	62.5000000000	37.5000000000	26.77548	10.0000
a3110	103	62.5000000000	37.5000000000	16.77548	10.0000
a4110	104	62.5000000000	37.5000000000	6.77548	10.0000
a1111	101	62.5000000000	62.5000000000	32.36730	10.0000
a2111	102	62.5000000000	62.5000000000	22.36730	10.0000
a3111	103	62.5000000000	62.5000000000	12.36730	10.0000
a4111	104	62.5000000000	62.5000000000	2.36730	10.0000
a1112	101	62.5000000000	87.5000000000	27.95913	10.0000
a2112	102	62.5000000000	87.5000000000	17.95913	10.0000
a3112	103	62.5000000000	87.5000000000	7.95913	10.0000
a4112	104	62.5000000000	87.5000000000	-2.04087	10.0000
a1113	101	87.5000000000	12.5000000000	36.77548	10.0000
a2113	102	87.5000000000	12.5000000000	26.77548	10.0000
a3113	103	87.5000000000	12.5000000000	16.77548	10.0000
a4113	104	87.5000000000	12.5000000000	6.77548	10.0000
a1114	101	87.5000000000	37.5000000000	32.36730	10.0000
a2114	102	87.5000000000	37.5000000000	22.36730	10.0000
a3114	103	87.5000000000	37.5000000000	12.36730	10.0000
a4114	104	87.5000000000	37.5000000000	2.36730	10.0000
a1115	101	87.5000000000	62.5000000000	27.95913	10.0000
a2115	102	87.5000000000	62.5000000000	17.95913	10.0000
a3115	103	87.5000000000	62.5000000000	7.95913	10.0000
a4115	104	87.5000000000	62.5000000000	-2.04087	10.0000
a1116	101	87.5000000000	87.5000000000	23.55095	10.0000
a2116	102	87.5000000000	87.5000000000	13.55095	10.0000
a3116	103	87.5000000000	87.5000000000	3.55095	10.0000
a4116	104	87.5000000000	87.5000000000	-6.44905	10.0000
bound					
0.0	0.0				
0.0	100.0				
100.0	100.0				
100.0	0.0				
toler					
0.5					
Note: Nodes with same lay_index are within same layer					

3.3.5.2 Output data

Table 3.3.5.2 Element block - rectangular domain, uniform 3D grid spacing, layer slope = 10° in x-direction and 10° in the y-direction.

ELEME					
Name	rock type	volume m^3	x-coordinate (m)	y-coordinate (m)	z-coordinate (m)
a11 1	rock1	6.250e+03	12.500	12.500	50.0000
a21 1	rock1	6.250e+03	12.500	12.500	40.0000
a31 1	rock1	6.250e+03	12.500	12.500	30.0000
a41 1	rock1	6.250e+03	12.500	12.500	20.0000
a11 2	rock1	6.250e+03	12.500	37.500	45.5918
a21 2	rock1	6.250e+03	12.500	37.500	35.5918
a31 2	rock1	6.250e+03	12.500	37.500	25.5918
a41 2	rock1	6.250e+03	12.500	37.500	15.5918
a11 3	rock1	6.250e+03	12.500	62.500	41.1837
a21 3	rock1	6.250e+03	12.500	62.500	31.1837
a31 3	rock1	6.250e+03	12.500	62.500	21.1837
a41 3	rock1	6.250e+03	12.500	62.500	11.1837
a11 4	rock1	6.250e+03	12.500	87.500	36.7755
a21 4	rock1	6.250e+03	12.500	87.500	26.7755
a31 4	rock1	6.250e+03	12.500	87.500	16.7755
a41 4	rock1	6.250e+03	12.500	87.500	6.7755
a11 5	rock1	6.250e+03	37.500	12.500	45.5918
a21 5	rock1	6.250e+03	37.500	12.500	35.5918
a31 5	rock1	6.250e+03	37.500	12.500	25.5918
a41 5	rock1	6.250e+03	37.500	12.500	15.5918
....					
....					
a1113	rock1	6.250e+03	87.500	12.500	36.7755
a2113	rock1	6.250e+03	87.500	12.500	26.7755
a3113	rock1	6.250e+03	87.500	12.500	16.7755
a4113	rock1	6.250e+03	87.500	12.500	6.7755
a1114	rock1	6.250e+03	87.500	37.500	32.3673
a2114	rock1	6.250e+03	87.500	37.500	22.3673
a3114	rock1	6.250e+03	87.500	37.500	12.3673
a4114	rock1	6.250e+03	87.500	37.500	2.3673
a1115	rock1	6.250e+03	87.500	62.500	27.9591
a2115	rock1	6.250e+03	87.500	62.500	17.9591
a3115	rock1	6.250e+03	87.500	62.500	7.9591
a4115	rock1	6.250e+03	87.500	62.500	-2.0409
a1116	rock1	6.250e+03	87.500	87.500	23.5510
a2116	rock1	6.250e+03	87.500	87.500	13.5510
a3116	rock1	6.250e+03	87.500	87.500	3.5509
a4116	rock1	6.250e+03	87.500	87.500	-6.4490

Table 3.3.5.3 Connection block - rectangular domain, uniform 3D grid spacing, layer slope = 10° in x-direction and 10° in y-direction.

CONNE						
Element1	**Element2**	**isot**	**d_1 (m)**	**d_2 (m)**	**A_{mn} m^2**	**Cos(β)**
a11 1	a11 2	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a11 1	a11 5	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a21 1	a21 2	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a21 1	a21 5	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a21 1	a11 1	1	5.000e+00	5.000e+00	6.250e+02	1.
a31 1	a31 2	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a31 1	a31 5	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a31 1	a21 1	1	5.000e+00	5.000e+00	6.250e+02	1.
a41 1	a41 2	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a41 1	a41 5	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a41 1	a31 1	1	5.000e+00	5.000e+00	6.250e+02	1.
a11 2	a11 3	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a11 2	a11 6	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a21 2	a21 3	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a21 2	a21 6	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a21 2	a11 2	1	5.000e+00	5.000e+00	6.250e+02	1.
a31 2	a31 3	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a31 2	a31 6	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a31 2	a21 2	1	5.000e+00	5.000e+00	6.250e+02	1.
a41 2	a41 3	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a41 2	a41 6	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a41 2	a31 2	1	5.000e+00	5.000e+00	6.250e+02	1.
a11 3	a11 4	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a11 3	a11 7	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a21 3	a21 4	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
....						
....						
a3114	a2114	1	5.000e+00	5.000e+00	6.250e+02	1.
a4114	a4115	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a4114	a3114	1	5.000e+00	5.000e+00	6.250e+02	1.
a1115	a1116	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a2115	a2116	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a2115	a1115	1	5.000e+00	5.000e+00	6.250e+02	1.
a3115	a3116	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a3115	a2115	1	5.000e+00	5.000e+00	6.250e+02	1.
a4115	a4116	1	1.250e+01	1.250e+01	2.500e+02	-1.736e-01
a4115	a3115	1	5.000e+00	5.000e+00	6.250e+02	1.
a2116	a1116	1	5.000e+00	5.000e+00	6.250e+02	1.
a3116	a2116	1	5.000e+00	5.000e+00	6.250e+02	1.
a4116	a3116	1	5.000e+00	5.000e+00	6.250e+02	1.

3.3.6 Non-uniform 2D grid spacing in a rhombohedron domain.

3.3.6.1 Input data.

Table 3.3.6.1 Input data - rhombohedron domain, non-uniform 2D grid spacing

locat input with x-slope = 0.0 deg and y-slope = 0.0 deg					
Name	**Lay_index**	**x-coordinate (m)**	**y-coordinate (m)**	**z-coordinate(m)**	**thickness (m)**
al1 1	101	10.5000000000	19.9502999793	50.00000	10.0000
al1 2	101	14.5000000000	41.1505999586	50.00000	10.0000
al1 3	101	11.5000000000	67.1505999586	50.00000	10.0000
al1 4	101	8.5000000000	95.9502999793	50.00000	10.0000
al1 5	101	39.9622305477	33.4502999793	50.00000	10.0000
al1 6	101	35.9622305477	53.6505999586	50.00000	10.0000
al1 7	101	37.9622305477	81.6505999586	50.00000	10.0000
al1 8	101	33.9622305477	103.4502999793	50.00000	10.0000
al1 9	101	66.9622305477	46.9502999793	50.00000	10.0000
al110	101	63.9622305477	72.1505999586	50.00000	10.0000
al111	101	68.9622305477	90.1505999586	50.00000	10.0000
al112	101	62.9622305477	120.9502999793	50.00000	10.0000
al113	101	87.7311152738	51.4502999793	50.00000	10.0000
al114	101	95.7311152738	96.6505999586	50.00000	10.0000
al115	101	87.7311152738	111.6505999586	50.00000	10.0000
al116	101	92.7311152738	133.4502999793	50.00000	10.0000
bound					
0.0	0.0				
0.0	100.0				
100.0	150.0				
100.0	50.0				
toler					
0.5					
Note: Nodes with same lay_index are within same layer					

3.3.6.2 Output data

Table 3.3.6.2 Element block - rhombohedron domain, non-uniform 2D grid spacing

ELEME					
Name	**rock type**	**volume m^3**	**x-coordinate (m)**	**y-coordinate (m)**	**z-coordinate (m)**
al1 1	rock1	6.476e+03	10.500	19.950	50.0000
al1 2	rock1	6.036e+03	14.500	41.151	50.0000
al1 3	rock1	6.731e+03	11.500	67.151	50.0000
al1 4	rock1	5.074e+03	8.500	95.950	50.0000
al1 5	rock1	6.560e+03	39.962	33.450	50.0000
al1 6	rock1	6.387e+03	35.962	53.651	50.0000
al1 7	rock1	6.999e+03	37.962	81.651	50.0000
al1 8	rock1	7.071e+03	33.962	103.450	50.0000
al1 9	rock1	6.291e+03	66.962	46.950	50.0000
al110	rock1	6.746e+03	63.962	72.151	50.0000
al111	rock1	7.212e+03	68.962	90.151	50.0000
al112	rock1	7.189e+03	62.962	120.950	50.0000
al113	rock1	6.279e+03	87.731	51.450	50.0000
al114	rock1	4.881e+03	95.731	96.651	50.0000
al115	rock1	5.050e+03	87.731	111.651	50.0000
al116	rock1	5.018e+03	92.731	133.450	50.0000

Table 3.3.6.3 Connection block - rhombohedron domain, non-uniform 2D grid spacing

CONNE						
Element1	**Element2**	**isot**	**d_1 (m)**	**d_2 (m)**	**A_{mn} m^2**	**Cos(β)**
al1 1	al1 2	1	1.079e+01	1.079e+01	2.494e+02	0.0
al1 1	al1 5	1	1.620e+01	1.620e+01	1.435e+02	0.0
al1 2	al1 3	1	1.309e+01	1.309e+01	2.092e+02	0.0
al1 2	al1 6	1	1.242e+01	1.242e+01	1.546e+02	0.0
al1 2	al1 5	1	1.330e+01	1.330e+01	1.400e+02	0.0
al1 3	al1 4	1	1.448e+01	1.448e+01	2.034e+02	0.0
al1 3	al1 7	1	1.509e+01	1.509e+01	1.634e+02	0.0
al1 3	al1 6	1	1.397e+01	1.397e+01	1.512e+02	0.0
al1 4	al1 8	1	1.327e+01	1.327e+01	1.966e+02	0.0
al1 4	al1 7	1	1.637e+01	1.637e+01	8.600e+01	0.0
al1 5	al1 6	1	1.030e+01	1.030e+01	2.241e+02	0.0
al1 5	al1 9	1	1.509e+01	1.509e+01	1.858e+02	0.0
al1 6	al1 7	1	1.404e+01	1.404e+01	1.928e+02	0.0
al1 6	al110	1	1.678e+01	1.678e+01	1.058e+02	0.0
al1 6	al1 9	1	1.586e+01	1.586e+01	1.232e+02	0.0
al1 7	al1 8	1	1.108e+01	1.108e+01	2.735e+02	0.0
al1 7	al111	1	1.607e+01	1.607e+01	1.101e+02	0.0
al1 7	al110	1	1.384e+01	1.384e+01	1.890e+02	0.0
al1 8	al112	1	1.694e+01	1.694e+01	2.131e+02	0.0
al1 8	al111	1	1.872e+01	1.872e+01	8.462e+01	0.0
al1 9	al110	1	1.269e+01	1.269e+01	2.187e+02	0.0
al1 9	al113	1	1.063e+01	1.063e+01	2.145e+02	0.0
al110	al111	1	9.341e+00	9.341e+00	3.413e+02	0.0
al110	al114	1	2.006e+01	2.006e+01	8.622e+00	0.0
al110	al113	1	1.576e+01	1.576e+01	1.880e+02	0.0
al111	al112	1	1.569e+01	1.569e+01	1.817e+02	0.0
al111	al115	1	1.427e+01	1.427e+01	1.245e+02	0.0
al111	al114	1	1.377e+01	1.377e+01	2.362e+02	0.0
al112	al116	1	1.614e+01	1.614e+01	1.268e+02	0.0
al112	al115	1	1.323e+01	1.323e+01	1.979e+02	0.0
al113	al114	1	2.295e+01	2.295e+01	1.299e+02	0.0
al114	al115	1	8.500e+00	8.500e+00	2.141e+02	0.0
al115	al116	1	1.118e+01	1.118e+01	2.187e+02	0.0

Table 3.3.6.4 Mesh geometry -rhombohedron domain, non-uniform 2D grid spacing.

Geometry						
X1 (m)	Y1(m)	X2(m)	Y2(m)	Index	Element1	Element2
0	0	0	32.9089072	0	al1 1	* 1
0	32.9089072	24.5047879	28.2854277	1	al1 1	al1 2
24.5047879	28.2854277	30.4819385	15.2409693	1	al1 1	al1 5
30.4819385	15.2409693	0	0	0	al1 1	* 2
0	32.9089072	0	52.6506	0	al1 2	* 3
0	52.6506	20.7771494	55.0479634	1	al1 2	al1 3
20.7771494	55.0479634	28.5580819	41.68827	1	al1 2	al1 6
28.5580819	41.68827	24.5047879	28.2854277	1	al1 2	al1 5
24.5047879	28.2854277	0	32.9089072	0	al1 2	al1 1
0	52.6506	0	80.5087725	0	al1 3	* 4
0	80.5087725	20.2294638	82.6160302	1	al1 3	al1 4
20.2294638	82.6160302	28.0822255	68.284886	1	al1 3	al1 7
28.0822255	68.284886	20.7771494	55.0479634	1	al1 3	al1 6
20.7771494	55.0479634	0	52.6506	0	al1 3	al1 2
0	100	18.4287126	109.214356	0	al1 4	* 5
18.4287126	109.214356	23.9844963	90.3526703	1	al1 4	al1 8
....						
....						
58.8499044	29.4249522	30.4819385	15.2409693	0	al1 5	* 7
20.7771494	55.0479634	28.0822255	68.284886	0	al1 6	al1 3
28.0822255	68.284886	47.3122975	66.9113095	1	al1 6	al1 7
47.3122975	66.9113095	53.1445742	58.0840798	1	al1 6	al110
53.1445742	58.0840798	50.5416836	46.0413938	1	al1 6	al1 9
50.5416836	46.0413938	28.5580819	41.68827	0	al1 6	al1 5
28.5580819	41.68827	20.7771494	55.0479634	0	al1 6	al1 2
87.2117964	74.8503241	100	72.5869403	1	al113	al114
100	72.5869403	100	50	0	al113	* 12
86.6852466	75.5330955	81.1116279	98.4868734	0	al114	al111
81.1116279	98.4868734	100	108.560672	1	al114	al115
100	108.560672	100	72.5869403	0	al114	* 13
100	72.5869403	87.2117964	74.8503241	0	al114	al113
87.2117964	74.8503241	86.6852466	75.5330955	0	al114	al110
100	108.560672	81.1116279	98.4868734	0	al115	al114
81.1116279	98.4868734	71.7325414	106.674547	0	al115	al111
71.7325414	106.674547	78.6873869	125.198131	0	al115	al112
78.6873869	125.198131	100	120.309849	1	al115	al116
100	120.309849	100	108.560672	0	al115	* 14
100	150	100	120.309849	0	al116	* 15
100	120.309849	78.6873869	125.198131	0	al116	al115
78.6873869	125.198131	73.7783083	136.889154	0	al116	al112
73.7783083	136.889154	100	150	0	al116	* 16
Note: element2 beginning with '*' implies line segment is along the boundary, else Element1 and Element2 are separated by line segmt X1,Y1 to X2,Y2. INDEX of '0' implies line segment is a duplicate, except when it is on the boundary. In processing this geometry file for plotting, line segments with '0' index should be ignored except when they are on the boundary.						

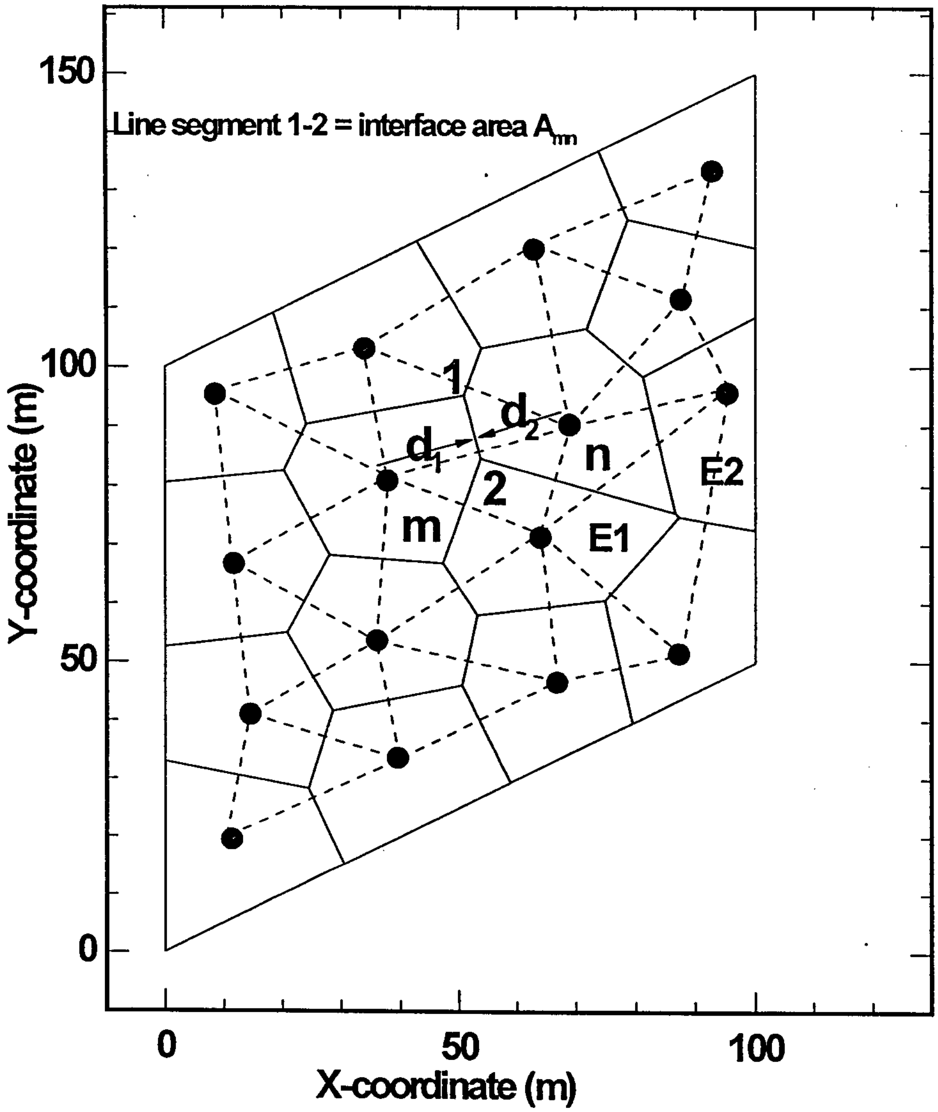

Fig. 3.3.3 Space discretization and geometry for the integral finite difference method grid. Non-uniform 2D grid spacing in a rhombohedron domain; Solid lines are Voronoi tessellation space; Dashed lines show valid connections between adjacent elements. Solid circles are the element centers.

Note that elements E1 and E2 share a small interface although the direction of flow between E1 and E2 does not intersect this interface. Such small interfaces in irregular grids can be corrected by adjusting the coordinates of **E1**, **E2** and **n** so as to increase the interface area of E1/E2 or eliminate it.

REFERENCES

1. Ahuja, N. (1982). Dot pattern processing using Voronoi polygons as neighborhoods. IEEE *Trans. Patt. Anal. Mach. Int.* PAMI-4 pp. 336-343.

2. Aurenhammer, F., (1991). Voronoi diagrams; A survey of a fundamental geometrical structure: ACM Computing Survey, 23(3), p. 345-405.

3. Fortune, S. (1993). Computational geometry: In, directions in computation information geometers, R. Martin, editor., 47 Stockers Avenue, Winchester, S022 5LB, UK, ISBN-1-87428-02-X.

4. Fortune, S. (1988). Sorting helps for Voronoi diagrams. Manuscript, ATT Bell Lab., Murray, New Jersey.

5. Fortune, S. (1987). A sweepline algorithm for Voronoi diagrams. *Alogoritmica*, 2, pp.153-174.

6. Heinman. Z.E, G. Gerken and G. Hantelmann (1983). Using grid refinement in multiple-application reservoir simulator. SPE 12225, Annual Technical Conference and Exhibition, Dallas, Oct 6-9th.

7. Narasimhan, T.N., and P.A. Witherspoon, (1976). An Integrated Finite Difference Method For Analyzing Fluid Flow In Porous Media, Water Resources Research, NNA.19890522.0253, , 12(1), pp. 57-64.

8. Peaceman, D.W. (1977). Fundamentals of Numerical Reservoir Simulation Elsevier Scientific Publishing Company, North Holland, Amsterdam.

9. Pedrosa, O.A. and K., Aziz (1986). The use hybrid grids in Reservoir Simulation. SPE, Symposium on Reservoir simulation Nov. pp. 611-621.

10. Pruess, K., A. Simmons, Y.S. Wu, and G.J. Moridis, (1996). TOUGH2 software qualification, Lawrence Berkeley Laboratory Report LBL-38383, Lawrence Berkeley Laboratory, Berkeley, CA.

11. Pruess, K. (1990). TOUGH2 A general purpose Numerical Simulator for multiphase fluid and heat flow. Lawrence Berkeley Laboratory report, LBL-29400.

12. Pruess, K., (1987). TOUGH user's guide, Nuclear Regulatory Commission LBL-20700, Lawrence Berkeley Laboratory, Berkeley, CA, 1987.

13. Quandalle, P. and Bisset, P. (1983). The use of flexible grids for improved reservoir modeling. SPE 12239, Seventh Symposium on Reservoir simulation, San Francisco, Nov. 16-18th.

14. Watson, D. F., (1985). Natural neighbor sorting. The Australian computer Journal, Vol. 17 No. 4. P. 189-193.

15. von Rosenberg, D.U. (1982). Local grid refinement for finite difference methods. SPE 10974; Annual Technical conference and Exhibition, New Orleans, September 26-29th.

16. Voronoi, M.G. (1908). Nouvelles applications des parametres continus a la theorie des formes quadratiques. *J. Reine Angew. Math.* 134, pp. 198-287.

APPENDIX

SOURCE LISTING FOR AMESH MODULES.

The modules in C that constitute AMESH are: *Amesh.c, Amesh.h, bleme.c, bstar.c, extre.c, pmesh.c and rmesh.c*. The source listing for the seven modules, as well as a Unix Makefile are appended below.

1. Amesh.c

Locates input file and calls all associated modules

```
#include <stdio.h>

struct  locat   *locat;
struct  bound   *bound;
double  toler = 0.;

void
main()
{
        FILE    *fp;

        if((fp = fopen("in","r")) == NULL) {
                perror("in-open"); exit(1);
        }
        rmesh(fp);
        (void)fclose(fp);
        if(!locat) exit(1);
        extre();
        pmesh();
}
```

2. Amesh.h

Sets input/output data structure and types

```
struct  locat   {
        char    l_name[6];
        short   l_layer;
        double  l_x,l_y,l_z;
        double  l_thick;
        struct  locat   *l_next;
};
struct  bound   {
        struct  bound   *b_next;
        struct  locat   *b_other;
        double  b_x,b_y;
};
```

3. beleme.c

Sorts all the adjacency relationships and computes the bounding connections for any interior node

```
#include "amesh.h"

#define NULL  ((char *)0)

extern  struct  locat   *locat;

struct  bound *bstar(/* struct locat * */);

struct  bound *
belem(l,abp)
        struct  locat   *l, **abp;
{
        struct  locat   *o;
        struct  bound   *b0, *b1, *bc, *b;
        int     flag;
        double  dx, dy, c, c1, c2, c3;

        *abp = (struct locat *)NULL;
/*
 * create a known bounding polygon
 */
        b0 = bstar(l);
        for(o = locat; o; o = o->l_next) {
                if(o == l) continue;
                dx = (o->l_x -l->l_x);
                dy = (o->l_y -l->l_y);
                c = (dx*dx + dy*dy)/2;
                if(c == 0.0) {
                        if(o->l_z <= l->l_z) continue;
                        if(*abp && (*abp)->l_z <= o->l_z) continue;
                        *abp = o;
                        continue;
                }
                c += dx*l->l_x + dy*l->l_y;
/*
 * locate a vertex inside the connection
 */
                b = b0;
                bc = (struct bound *)NULL;
                flag = 0;
                do {
                        if(b->b_other &&
                          b->b_other->l_x == o->l_x &&
                          b->b_other->l_y == o->l_y) {
                                flag = 1;
                                if(o->l_layer == l->l_layer)
                                        b->b_other = o;
                        }
```

```
                        if(!bc && dx*b->b_x + dy*b->b_y <= c)
                                bc = b;
                        b = b->b_next;
                } while(b != b0);
                if(flag) continue;
                if(!bc) continue;
                for(b = b0 = bc; (b = b->b_next) != b0; bc = b) {
/*
 * locate a boundary segment intersected by the line
 */
                        if(dx*b->b_x + dy*b->b_y <= c) continue;
                        if((b1 = (struct bound *)malloc(sizeof(*b))) == (struct bound *)NULL) {
                                perror("malloc-beleme"); exit(1);
                        }
                        bc->b_next = b1;
                        c1 = dx*bc->b_x + dy*bc->b_y;
                        c2 = dx*b->b_x  + dy*b->b_y;
                        c3 = (c - c1)/(c2 - c1);
                        b1->b_other = o;
                        b1->b_x = bc->b_x + (b->b_x - bc->b_x)*c3;
                        b1->b_y = bc->b_y + (b->b_y - bc->b_y)*c3;
                        for(bc = b; (b = b->b_next) != b0; bc = b) {
                                if(dx*b->b_x + dy*b->b_y <= c) break;
                                (void)free((char *) bc);
                        }
                        b1->b_next = bc;
                        c1 = dx*bc->b_x + dy*bc->b_y;
                        c2 = dx*b->b_x  + dy*b->b_y;
                        c3 = (c - c1)/(c2 - c1);
                        bc->b_x = bc->b_x + (b->b_x - bc->b_x)*c3;
                        bc->b_y = bc->b_y + (b->b_y - bc->b_y)*c3;
                        if(b == b0) break;
                }
        }
        return(b0);
}
```

4. bstar.c

For computing interfaces that lie on the boundary (Names with '*' in the output)

```
#include "amesh.h"

#define NULL    ((char *)0)

extern  struct  bound   *bound;

char    *malloc(/* unsigned int */);

/*
 * create a known bounding polygon
 */
struct  bound   *
```

```
bstar(l)
        struct    locat     *l;
{
        struct    bound     *b0, *b, *b1, *bc;
        struct    bound     **bp;
        double    d0, d, dx, dy;

        bp = &b0;
        bc = (struct bound *)NULL;
        b = bound;
        do {
                if(((*bp) = (struct bound *)malloc(sizeof(*b))) == (struct bound *)NULL) {
                        perror("bstar-malloc"); exit(1);
                }
                **bp = *b;
                dx = b->b_x - l->l_x;
                dy = b->b_y - l->l_y;
                d = dx*dx + dy*dy;
                if(!bc || d0 > d) {
                        bc = *bp;
                        d0 = d;
                }
                bp = &(*bp)->b_next;
                b = b->b_next;
        } while(b != bound);
        *bp = b0;
        return(bc);
}
```

5. Extre.c

Finds extreme values from the input xy coordinates

```
#include "amesh.h"

#define NULL   ((char *)0)

extern   struct   locat    *locat;
extern   struct   bound    *bound;

void
extre()
{
        double    xl, yl, xh, yh;
        struct    locat     *l;

        if(bound) return;
        bound = (struct bound *)malloc(4*sizeof(struct bound));
        if(bound == (struct bound *)NULL) {
                perror("extre-malloc"); exit(1);
        }
        xh = xl = locat->l_x;
        yh = yl = locat->l_y;
        for(l = locat->l_next; l; l = l->l_next) {
```

```
                if(xl > l->l_x) xl = l->l_x;
                if(xh < l->l_x) xh = l->l_x;
                if(yl > l->l_y) yl = l->l_y;
                if(yh < l->l_y) yh = l->l_y;
        }
        bound[0].b_next = &bound[1];
        bound[0].b_other = (struct locat *)NULL;
        bound[0].b_x = xl;
        bound[0].b_y = yl;
        bound[1].b_next = &bound[2];
        bound[1].b_other = (struct locat *)NULL;
        bound[1].b_x = xh;
        bound[1].b_y = yl;
        bound[2].b_next = &bound[3];
        bound[2].b_other = (struct locat *)NULL;
        bound[2].b_x = xh;
        bound[2].b_y = yh;
        bound[3].b_next = bound;
        bound[3].b_other = (struct locat *)NULL;
        bound[3].b_x = xl;
        bound[3].b_y = yh;
}
```

6. pmesh.c

Computes the Voronoi tessellation and writes out the 'eleme', 'conne' and 'segmt' output files.

```
#include <stdio.h>
#include <math.h>

#include "amesh.h"

extern  struct  locat    *locat;
extern  double  toler;

struct  bound   *belem(/* struct locat *, struct locat ** */);

void
pmesh()
{
        FILE    *ef,*cf,*sf;
        struct  bound   *b, *b1, *b2;
        struct  locat   *above, *l, *j;
        double  vol, area, dx, dy, dz, beta, a;
        double  d1, d2;
        int     nb = 0;
        int     k;
        char    *cp;
        char    bnd[6];

        if((ef = fopen("eleme","w")) == NULL) {
                perror("open-eleme"); exit(1);
```

```
}
if((cf = fopen("conne","w")) == NULL) {
        perror("open-conne"); exit(1);
}
if((sf = fopen("segmt","w")) == NULL) {
        perror("open-eleme"); exit(1);
}
(void)fprintf(ef,"eleme\n");
(void)fprintf(cf,"conne\n");
for(l = locat; l; l= l->l_next) {
        vol = 0.0;
        area = 0.0;
        b = belem(l,&above);
        for(b1 = b, b2 = NULL; !b2 || b1 != b; b1 = b2) {
                double  vp;
                b2 = b1->b_next;
                dx = b2->b_x - b1->b_x;
                dy = b2->b_y - b1->b_y;
                d2 = dx*dx + dy*dy;
                if(d2 == 0.0) continue;
                if(d2 < toler) {
                        d2 = 0;
                        continue;
                }
                d2 = sqrt(d2);
                d1 = dy*(b1->b_x - l->l_x) - dx*(b1->b_y - l->l_y);
                if(d1 < 0.0) d1 = -d1;
                d1 /= d2;
                j = b1->b_other;
                beta = 0;
                k = 0;
                if(j && j->l_layer ==  l->l_layer) {
                        a = d2*(l->l_thick + j->l_thick)/2;
                        vp = a*d1/3 + d1*d2*l->l_thick/6;
                        area += d1*d2/2;
                        cp = j->l_name;
                        if(j > l) {
                                k = 1;
                                dz = j->l_z - l->l_z;
                                beta = dz / sqrt(4*d1*d1 + dz*dz);
                                d2 = d1;
                        }
                }
                else {
                        if(d1 == 0.0 || j) {
                                cp = "*   0";
                                a = 0.;
                                vp = d1*d2*l->l_thick/6;
                                area += d1*d2/2;
                                d2 = 0.;
                        }
                        else {
                                a = d2*l->l_thick;
                                vp = a*d1/2;
```

```
                                        area += d1*d2/2;
                                        d2 = 0;
                                        (void)sprintf(cp = bnd,"*%4d",++nb);
                                }
                                j = NULL;
                        }
                        vol += vp;
                        (void)fprintf(sf,
                          "%15.9g%15.9g%15.9g%15.9g %d %-5.5s%-5.5s\n",
                          b1->b_x,b1->b_y,
                          b2->b_x,b2->b_y,
                          k,l->l_name,cp);
                        if(j && !k) continue;
                        (void)fprintf(cf,
                          "%-5.5s%-5.5s          1%10.3e%10.3e%10.3e",
                          l->l_name,
                          cp,
                          d1,d2,a);
                        if(beta != 0.0) (void)fprintf(cf,"%10.3e\n",beta);
                        else (void)fprintf(cf,"\n");
                }
                if(above) {
                        (void)fprintf(cf,
                          "%-5.5s%-5.5s          1%10.3e%10.3e%10.3e 1.\n",
                          l->l_name,
                          above->l_name,
                          l->l_thick/2,above->l_thick/2,area);
                }
                (void)fprintf(ef,
                  "%-5.5s       rock1%10.3e            %10.3f%10.3f%10.4f\n",
                  l->l_name,
                  vol,l->l_x,l->l_y,l->l_z);
                for(b1 = b, b2 = NULL; b1 != b && !b2; b1 = b2) {
                        b2 = b1->b_next;
                        (void)free(b1);
                }
        }
        (void)fprintf(ef,"\n");
        (void)fprintf(cf,"\n");
        (void)fclose(ef);
        (void)fclose(cf);
        (void)fclose(sf);
}
```

7. rmesh.c

Reads all the required input parameters from the file 'in'.

```
#include <stdio.h>
#include <ctype.h>

#include "amesh.h"
```

```
extern  struct  locat   *locat;
extern  struct  bound   *bound;
extern  double  toler;

void    rmesh(fp)
        FILE    *fp;
{
        char    buf[1024];
        char    *cp;
        struct  locat   **l = &locat;
        struct  bound   **b = &bound;

/*
 * this subroutine reads in the mesh information i.e. the element
 * coordinates, connection distances, and any boundary description.
 */

/*
 * what kind of block have we encountered (ignore any unknown)
 */
        while((cp = fgets(buf, 1024, fp)) != NULL) {
/*
 * put into lower case (upper case gives me a headache)
 */
                for(; *cp && cp < buf + 5; cp++)
                        if(isupper(*cp)) *cp = tolower(*cp);
/*
 * the location block -- read until you find a blank element name
 */
                if(!locat && strncmp(buf,"locat",5) == 0) for(;;) {
                        double  x,y,z,thick;
                        int     layer;

                        if((cp = fgets(buf,1024,fp)) == NULL) return;
                        for(; isspace(*cp); cp++) ;
                        if(cp >= buf + 5 || *cp == '\0') break;
                        if((*l = (struct locat *)malloc(sizeof(**l))) == NULL) {
                                perror("locat-malloc"); exit(1);
                        }
                        (*l)->l_next = NULL;
                        strncpy((*l)->l_name,buf,5);
                        sscanf(buf+5,"%d %lf %lf %lf %lf",
                                &layer,
                                &x,
                                &y,
                                &z,
                                &thick);
                        (*l)->l_layer = layer;
                        (*l)->l_x = x;
                        (*l)->l_y = y;
                        (*l)->l_z = z;
                        (*l)->l_thick = thick;
                        l = &(*l)->l_next;
                }
```

```
/*
 * the boundary block -- how do i draw the mesh
 */
                else if(!bound && strncmp(buf,"bound",5) == 0) for(;;) {
                        if((cp = fgets(buf,1024,fp)) == NULL) return;
                        for(; isspace(*cp); cp++) ;
                        if(cp >= buf + 10 || *cp == '\0') break;
                        if(((*b) = (struct bound *)malloc(sizeof(**b))) == NULL) {
                                perror("bound-malloc"); exit(1);
                        }
                        (*b)->b_next = bound;
                        (*b)->b_other = NULL;
                        sscanf(buf,"%lf %lf",
                                &(*b)->b_x,
                                &(*b)->b_y);
                        b = &(*b)->b_next;
                }
                else if(!toler && strncmp(buf,"toler",5) == 0) {
                        if(fgets(buf,1024,fp) == NULL) return;
                        sscanf(buf,"%lf",&toler);
                        toler *= toler;
                }
/*
 * if we have all the information we want, we can stop reading
 */
                if(locat && bound && toler) return;
        }
}
```

8. Makefile

A Unix c compiler make to create an excutable grid generator program

```
OBJS=   gmesh.o \
        belem.o \
        bstar.o \
        extre.o \
        pmesh.o \
        rmesh.o

LIBS=   -lm

CFLAGS=         -g

gmesh:  $(OBJS)
        $(CC) -o amesh.`hostname` $(CFLAGS) $(OBJS) $(LIBS)
```