NAME

corduroy − element mesh generation description file format

DESCRIPTION

The *corduroy*(4fe) file format is used by the *corduroy*(1fe) application for describing the desired element mesh generation. In general white space is unimportant in a file, arbitrary numeric expressions may be used, and case of keywords is unimportant. As per standard convention, boldface items represent keywords, italicized items represent the syntax of the grammar, and items in brackets are optional. The file syntax is shown below.

[*initialization*]
[*generators*]
end

Initialization

The *initialization* section occurs first in the file if present. It defines parameters common to all generators and has the following syntax.

start-node = *expression*
start-element = *expression*
constraint = *name*
material = *name*

The default *start-node* and *start-element* is one. If a *constraint* is given then the *name* will be assigned to the generated nodes. Similarly, if a *material* is given then the *name* will be assigned to the generated elements.

Generators

The *generators* section contains specifications for generating the elements and the associated nodes. The section has the following syntax.

[*line-generator*]
[*grid-generator*]
[*quadrilateral-grid-generator*]
[*brick-grid-generator*]
[*triangular-mesh-generator*]

Lines

A *line-generator* specifies the generation of elements along a line in three dimensions. The generator has the following syntax.

[**start =** *triple*]
[**end =** *triple*]
[**number =** *expression*]
[**rule = linear** | **log**]
[**element-type =** *name*]

where a *triple* has one of the following forms:

(*expression* **,** *expression* **,** *expression* **)**
(*expression* **,** *expression* **)**

If the z-coordinate of a triple is not specified then the previously specified z-coordinate is used. The initial default z-coordinate is zero. The *start* and *end* triples define the starting and ending coordinates of the line respectively. The number of elements along the line is specified by *number*. The *rule* assignment controls whether the elements are linearly or logarithmically distributed along the line. The *element-type* specifies the type of elements to be generated. The type of the element defined by *name* must be a linear element.

Grids

A *grid-generator* specifies the generation of elements within a three-dimensional grid. The generator has the following syntax.

[**start =** *triple*]

[**end =** *triple*]
[**x-number =** *expression*]
[**y-number =** *expression*]
[**z-number =** *expression*]
[**x-rule = linear** | **log**]
[**y-rule = linear** | **log**]
[**z-rule = linear** | **log**]
[**element-type =** *name*]

The *start* and *end* triples define the diagonally opposite corners of the grid. The *x-number*, *y-number*, and *z-number* assignments specify the number of elements along the x, y, and z dimensions respectively. Similarly, the *x-rule*, *y-rule*, and *z-rule* assignments specify whether the elements are to be generated linearly or logarithmically along the x, y, and z dimensions respectively. The type of the element specified by *name* must be a linear element.

Quadrilateral Grids

A *quadrilateral-grid-generator* specifies the generation of four-node planar elements within a two-dimensional grid. The generator has the following syntax.

[**start =** *pair*]
[**end =** *pair*]
[**x-number =** *expression*]
[**y-number =** *expression*]
[**x-rule = linear** | **log**]
[**y-rule = linear** | **log**]
[**element-type =** *name*]

The *start* and *end* pairs define the diagonally opposite corners of the grid. The *x-number*, and *y-number* assignments specify the number of elements along the x and y dimensions respectively. Similarly, the *x-rule*, and *y-rule* assignments specify whether the elements are to be generated linearly or logarithmically along the x and y dimensions, respectively. The type of the element specified by *name* must be a four-node planar element.

Brick Grids

A *brick-grid-generator* specifies the generation of solid brick elements within a three-dimensional grid. The generator has the following syntax.

[**start =** *triple*]
[**end =** *triple*]
[**x-number =** *expression*]
[**y-number =** *expression*]
[**z-number =** *expression*]
[**x-rule = linear** | **log**]
[**y-rule = linear** | **log**]
[**z-rule = linear** | **log**]
[**element-type =** *name*]

The *start* and *end* triples define the diagonally opposite corners of the grid. The *x-number*, *y-number*, and *z-number* assignments specify the number of elements along the x, y, and z dimensions respectively. Similarly, the *x-rule*, *y-rule*, and *z-rule* assignments specify whether the elements are to be generated linearly or logarithmically along the x, y, and z dimensions respectively. The type of the element specified by *name* must be an eight node solid element.

Triangular meshes

A *triangular-mesh-generator* specifies the generation of triangular elements within a two-dimensional mesh. The generator has the following syntax.

[**tolin =** *expression*]
[**angspc =** *expression*]
[**angtol =** *expression*]

[**dmin =** *expression*]
[**kappa =** *expression*]
[**min =** *expression*]
[**max =** *expression*]
[**boundary = [** *pair-list* **]**]
[**hole = [** *pair-list* **]**]
[**element-type =** *name*]

where a *pair* has the following form:

(*expression* **,** *expression* **)**

The *tolin*, *angspc*, *angtol*, *dmin*, *kappa*, *min*, and *max* parameters control the specifics of the mesh generation and are discussed in the user's guide. The element type specified by *name* must be a planar, triangular element. The *boundary* assignment specifies the boundary points of the mesh which must be given in counter-clockwise order. The *pair-list* is a sequence of *pairs*. A *hole* assignment specifies a hole within the mesh and similarly the points must be given in counter-clockwise order. Unlike other assignments, a *hole* assignment does not overwrite a previous assignment but instead adds to it. Thus, more than hole can be specified by repeating the *hole* assignment as many times as necessary.

AUTHOR

The *corduroy* file format was developed by Jason I. Gobat (jgobat@mit.edu) and Darren C. Atkinson (atkinson@ucsd.edu).

SEE ALSO

corduroy(1fe), felt(4fe).