AMIE configuration guide

Alain B. Giorla

November 20, 2015

This document shows how to write initiation files for AMIE. These files can be used to set up 2D simulations with the `2d_composite` executable.

This document only presents the basic capabilities of `2d_composite`. Advanced features will be described in another guide.

Contents

List of Tables

1 Syntax

Each line in the initiation file contains a variable for the AMIE simulation, using the following format:

```
.parameter = value # comment
```

Special characters:

`parameter` must be a character string which contains no space, dot, `@`, `#`, quotes, or brackets of any kind. Each variable must be on its own line.

`value` can either be a number (using `e` to express powers of 10) or a character string following the same rule as `parameter`. Some variables require no value, in which case both the `=` and the right-hand side can be omitted.

An entire line can be commented using the `#` character.

All character strings are case-sensitive.

Hierarchy:

The variables are organized using a tree-like hierarchy. The number of leading dots before `parameter` indicates the level of the current line in the tree. A `parameter` at level n+1 is attached to the `parameter` at level n placed directly above (hereafter called its father).

The order in which parameters of the same level and sharing the same father are defined is not relevant. The exception is when a parameter is defined multiple times, in which case only the last definition applies.

Paths and file names:

Several items can be defined using additional files. These files are always defined using the path relative to the folder from which AMIE was called. Folder separators are always `"/"` even in a Windows environment. As spaces are ignored, files or folders must not include spaces in their name.

Units:

S.I. units are generally used, except for times which are expressed in days. Notably, stresses are in Pascal.

2 Structure

The following first-level parameters must be defined in the initiation file. Some of them are required, as indicated in the Req. column.

	Req.	Description
`.define`	no	Advanced input parameters (not covered in this guide)
`.discretization`	yes	Parameters for the finite element mesh
`.stepping`	yes	Time steps and solver parameters
`.sample`	yes	Geometry and mechanical behaviour of the matrix
`.inclusions`	no	Geometry and mechanical behaviour of the inclusions
`.boundary_conditions`	yes	External mechanical boundary conditions
`.output`	no	Tables of average values (strain, stress, etc) at different time steps
`.export`	no	Mesh files to visualize certain values (strain, stress, etc)

Table 1: First-level parameters

Each of these items are described in a different section of this guide.

3 Discretization

The `discretization` item defines a parameters that control the mesh density and element type. All its parameters are optional. If a parameter is not found, its default value will be used instead.

`.discretization`	Def.	Description
`..minimum_mesh_density`	0.4	Controls the minimum distance between two mesh points.
`..order`	`LINEAR`	Order of the shape functions for the finite element discretization. In space-time finite elements, the order is automatically set to `LINEAR_TIME_LINEAR`.
`..sampling_number`	4	Number of mesh points on the side of the sample.
`..sampling_restriction`	0	Radius (in meters) below which inclusions are not meshed.
`..sampling_surface_factor`	2	Increases the density of the mesh along the surfaces of inclusions.

Table 2: Discretization parameters

4 Stepping

The `stepping` item defines parameters required for time-stepping and controls parameters of the solver. All its parameters are optional. If a parameter is not found, its default value will be used instead.

There are four different ways to define the time steps, depending on which parameters are defined. By order of priority:

.stepping	Def.	Description
..first_time_step	1	Value of the first time step in the absolute time scale. This is only required if `logarithmic` is set to `TRUE`.
..list_of_time_steps		List of coma-separated value or path to a text file containing the list of time steps written in a single column, starting with 0 and containing all instants in increasing order.
..logarithmic	`FALSE`	If this flag is set to `TRUE`, then the time steps will be spaced equally in the logarithmic scale.
..maximum_iterations_per_step	256	Maximum number of iterations of the damage algorithm at each time step
..minimum_time_step	1e-9	Smallest amount of time during which coupled damage and visco-elasticity are exactly calculated. Viscous effects are ignored below this value.
..number_of_time_steps	1	Number of time steps to perform.
..solver_precision	1e-8	Precision of the conjugate gradient solver
..ssor_iterations	20	Number of SSOR iterations used to stabilize the conjugate gradient solver
..time_step	1	Value of a time step in day (or in the logarithmic scale if `logarithmic` is set to `TRUE`).

Table 3: Stepping parameters

1. **With a direct list of time steps:**

   ```
   ..list_of_time_steps = 0,1,2,3
   ```

   The right-hand side must be a list of coma-separated values that will be used as the time steps of the simulation. The list must start with 0 and strictly increasing.

2. **From an external file:**

   ```
   ..list_of_time_steps = path/to/time_step_file.txt
   ```

   The values stored in the text file are used for the time steps of the simulation. The file must contain a single column of numbers, starting with 0, and strictly increasing. The file name must contain no coma, otherwise the first option will be detected.

3. **Using a logarithmic time step:**

   ```
   ..logarithmic = TRUE
   ..first_time_step = 1
   ..time_step = 1
   ..number_of_time_steps = 1
   ```

   The simulation will consists in `number_of_time_steps` time steps, starting with `first_time_steps` (in days), and separated by `time_step` (in the logarithmic space).

4. **Using a constant time step:**

```
..time_step = 1
..number_of_time_steps = 1
```

The simulation will perform `number_of_time_steps` time steps, each having the same value `time_step` (in days).

5 Sample

The `sample` item defines the geometry of the main box in which the simulation is performed as well as its mechanical behaviour. Geometrical parameters are optional (default values will be used if they are not found), but the mechanical behaviour must be defined in the file.

`.stepping`	Def.	Description
`..behaviour`		A **Behaviour** object representing the mechanical behaviour of the sample. Behaviours are described in their own section.
`..center`		
`...x`	0	x coordinate of the `center` of the box.
`...y`	0	y coordinate of the `center` of the box.
`..height`	0.1	Height of the box (y axis).
`..width`	0.1	Width of the box (x axis)

Table 4: Sample parameters

6 Inclusions

The `inclusions` item defines the inclusions embedded in the matrix. Most parameters are optional (default values will be used if not found). Parameters without default values are required.

6.1 Geometry

The `geometry` item defines basic properties of the inclusions. It consists into a `type` and several optional parameters, depending on the actual type of the inclusions. The following list details parameters common for different types of geometry. **Req.** indicates a required parameters (only for geometry types for which this is relevant).

The different types available are detailed below:

.inclusions	Def.	Description
..behaviour		A **Behaviour** object representing the mechanical behaviour of the inclusions. Behaviours are described in their own section.
..geometry	Circular	A **Geometry** objects which defines basic shape properties of the inclusions (see below).
..particle_size_distribution	Constant	A **Particle Size Distribution** object which represents the size distribution of the inclusions (see below).
..placement		A **Placement** object defining rules for the random placement of the inclusions (see below).
..copy_grain_behaviour	FALSE	If this is TRUE, then the mechanical behaviour will be randomized for each inclusion (but will remain constant in each inclusion). Otherwise it is randomized for each element in the inclusions.
..number		Number of inclusions to generate.
..radius_maximum		Maximum radius of the inclusions.
..sampling_factor	1	Increases the density of the mesh in the inclusions.
..surface_fraction		The fraction of the area of the sample covered by the inclusions.

Table 5: Inclusions parameters

..geometry = type	Def.	Description
...orientation	0	Default angle of the major axis of the inclusion with the x axis.
...orientation_variability	π	Variation of the random distribution of the angle of the major axis around its average value.
...placement_rotation	0	Maximum angle by which the inclusions can be rotated during the placement. Setting this value higher than 0 can result in microstructures in which the inclusions are not oriented as defined by orientation and orientation_variability
...shape_factor	Req.	Average ratio between major and minor axis of the inclusions.
...shape_factor_variability	0	Variation of the random distribution of the shape factor around its average value.
...vertex	Req.	Average number of vertexes for polygonal inclusions.
...vertex_variability	0	Variation of the number of vertexes for polygonal inclusions.

Table 6: Common geometry parameters

Circles:

..geometry = Circular	Def.	Description

Table 7: Geometry parameters for circles

Alternatively, using extended finite elements (constant radius):

..geometry = XFEM	Def.	Description

Table 8: Geometry parameters for extended finite elements circles

And for extended finite elements in space and time (constant radius):

..geometry = SpaceTimeXFEM	Def.	Description

Table 9: Geometry parameters for space-time extended finite elements circles

Ellipses:

..geometry = Ellipsoidal	Def.	Description
...orientation	0	
...orientation_variability	π	
...placement_rotation	0	
...shape_factor	Req.	
...shape_factor_variability	0	

Table 10: Geometry parameters for ellipses

Rectangles:

..geometry = Rectangular	Def.	Description
...orientation	0	
...orientation_variability	π	
...placement_rotation	0	
...shape_factor	Req.	
...shape_factor_variability	0	

Table 11: Geometry parameters for rectangles

Regular polygons:

..geometry = Polygonal	Def.	Description
...orientation	0	
...orientation_variability	π	
...placement_rotation	0	
...vertex	Req.	
...vertex_variability	0	

Table 12: Geometry parameters for rectangles

Gravel-like polygons:

..geometry = GravelPolygonal	Def.	Description
...amplitude_factor	0.9	Controls the regularity of the inclusions.
...amplitude_exponent	1.9	Controls the regularity of the inclusions
...degree	2	Higher degree may result in rougher surfaces.
...orientation	0	
...orientation_variability	π	
...placement_rotation	0	
...vertex	Req.	
...vertex_variability	0	

Table 13: Geometry parameters for gravel-like polygons

Crushed-like polygons:

As opposed to other geometries, crushed aggregates already result in inclusions with variable elongations. Therefore, `shape_factor_variability` is not a parameter of that specific geometry.

..geometry = CrushedPolygonal	Def.	Description
...orientation	0	
...orientation_variability	π	
...placement_rotation	0	
...shape_factor	Req.	
...vertex	Req.	
...vertex_variability	0	

Table 14: Geometry parameters for crushed-like polygons

Alternatively:

..geometry = CrushedSubtendedPolygonal	Def.	Description
...angle_variability		Variation of the angle between two consecutive segments of the polygons.
...orientation	0	
...orientation_variability	π	
...placement_rotation	0	
...shape_factor	Req.	
...vertex	Req.	
...vertex_variability	0	

Table 15: Geometry parameters for crushed-like polygons

Voronoi-generated polygons:

This specific method creates polygons extracted from a Voronoi diagram. **The resulting microstructure will NOT be a Voronoi diagram.**

..geometry = VoronoiPolygonal	Def.	Description
...box_width	Req.	Size of the box that is used to generate the Voronoi polygons (independent of the size of the sample or the inclusions).
...grains	Req.	Number of points used for the Delaunay triangulation upon which the Voronoi polygons will be based. This number does not correlate with the number of inclusions in the simulation. Instead, it dictates the number of different shapes available in the distribution.
...orientation	0	
...orientation_variability	π	
...placement_rotation	0	
...spacing	Req.	Distance between the points used for the Delaunay triangulation. That distance is related to box_width and not the size of the sample or the inclusions itself.

Table 16: Geometry parameters for Voronoi-generated polygons

6.2 Particle Size Distribution

The particle_size_distribution can either be defined as a pre-generated function, or using a text file. Most distributions have no parameters.

Constant distribution:

This is the default distribution if none is found.

..particle_size_distribution = ConstantSizeDistribution	Def	Description

Table 17: Parameters for the constant size distribution

Bolome distribution:

..particle_size_distribution	Def	Description
..particle_size_distribution = PSDBolomeA		Curve for concrete.
..particle_size_distribution = PSDBolomeB		Curve for concrete.
..particle_size_distribution = PSDBolomeC		Curve for concrete.
..particle_size_distribution = PSDBolomeD		Curve for mortar.

Table 18: Parameters for the Bolome particle size distributions

Fuller distribution:

..particle_size_distribution = PSDFuller	Def.	Description
...exponent	0.5	Defines the slope of the distribution.
...radius_minimum	0	Defines the smallest radius in the distribution.

Table 19: Parameters for the Fuller particle size distribution

Distribution from an external file:

This method reads a text file containing the discretized particle size distribution as a two-column table. The first column contains the fraction, the second the radii.

..particle_size_distribution = GranuloFromCumulativePSD	Def.	Description
...factor	1	Multiplies all radii in the distribution by the same factor.
...file_name		Path to the text file containing the particle size distribution.
...radius_maximum	-1	Cuts off the distribution above the specified radius (if positive).
...radius_minimum	-1	Cuts off the distribution below the specified radius (if positive).
...specification		Indicates how to read the file.

Table 20: Parameters for the file-defined particle size distribution

`specification` indicates if the fraction and radii are defined in increasing or decreasing order. It can be one of the following:

...specification =	Description
CUMULATIVE_PERCENT	Radii are sorted by decreasing order, and fractions are expressed in percentage, from 100 to 0.
CUMULATIVE_FRACTION	Radii are sorted by decreasing order, and fractions range from 1 to 0.
CUMULATIVE_ABSOLUTE	Radii are sorted by decreasing order, and fractions are in absolute value of the volume of aggregate, finishing with 0.
CUMULATIVE_PERCENT_REVERSE	Radii are sorted by increasing order, and fractions are expressed in percentage, from 0 to 100.
CUMULATIVE_FRACTION_REVERSE	Radii are sorted by increasing order, and fractions range from 0 to 1.
CUMULATIVE_ABSOLUTE_REVERSE	Radii are sorted by increasing order, and fractions are in absolute value of the volume of aggregate, starting with 0.

Table 21: Specification types for the file-defined particle size distribution

6.3 Placement

This item defines how particles are placed in the sample. All parameters are optional. If the geometry properties (`center`, `height` and `width`) are not defined, the inclusions will be placed in the `sample` of the simulation instead.

..placement	Def.	Description
...center		
....x		x coordinate of the `center` of the box in which the inclusions are placed.
....y		y coordinate of the `center` of the box in which the inclusions are placed.
...height		Height of the box in which the inclusions are placed (y axis).
...random_seed	1	Seed to generate different random microstructures based on the same inclusions.
...spacing	0	Minimum distance between inclusions, or between the inclusions and the edges of the placement box.
...tries	1000	Number of random tries for the placement.
...width		Width of the box in which the inclusions are placed (x axis)

Table 22: Placement parameters

7 Boundary Conditions

The `boundary_conditions` item contains a list of `boundary_condition` sub-items. As many `boundary_condition` subitems can be defined. Each `boundary_condition` item can have the following parameters. **Req** indicates required parameters; Others are optional.

..boundary_condition	Def.	Description
...condition	Req.	Type of boundary condition to apply.
...interpolation		Path to a text file containing the value of the boundary condition at different instants. The file must be written as a two-column table, with the time in the first column and the value in the second. A linear interpolation will be carried between the specified points.
...point		
....x		x coordinate of the node on which the boundary condition is applied.
....y		x coordinate of the node on which the boundary condition is applied.
...position	Req.	Edge of the sample on which the boundary condition is applied.
...restriction		
....top_right		
.....x	1	Maximum x coordinate of the nodes on which the boundary condition is applied.
.....y	1	Maximum y coordinate of the nodes on which the boundary condition is applied.
....bottom_left		
.....x	-1	Minimum x coordinate of the nodes on which the boundary condition is applied.
.....y	-1	Minimum y coordinate of the nodes on which the boundary condition is applied.
...rate	0	Rate of the boundary condition to apply. The value will be linear in time with the specified rate.
...value	0	Value of the boundary condition to apply.

Table 23: Boundary condition parameters

In its basic form, a `boundary_condition` applies a `condition` with a constant `value` on an edge or vertex of the sample defined with `position`.

Geometric restrictions:

If `restriction` is defined, the `condition` will only be applied on the nodes located on the specified `position` and in the rectangular box defined by the `top_right` and `bottom_left` points.

If `point` is defined, the `condition` will be applied to the node located on the specified `position` and closest to `point`.

If both `restriction` and `point` are defined, only `restriction` will be applied.

Time evolution:

In its basic form, the value of the `condition` is constant. However, it can be set as linear in time using `rate`, or use a linear interpolation in time using `interpolation`.

In case of an `interpolation`, the boundary condition will be interpolated linearly in-between the specified instants. Therefore, special care must be taken to represent step-wise loadings.

Conditions:

Common mechanical boundary conditions are:

`..condition =`	Description
`FIX_ALONG_XI`	Fixed horizontal (x) displacements.
`FIX_ALONG_ETA`	Fixed vertical (y) displacements.
`SET_ALONG_XI`	Imposed horizontal (x) displacements.
`SET_ALONG_ETA`	Imposed vertical (y) displacements.
`SET_STRESS_XI`	Imposed horizontal (x) stress.
`SET_STRESS_ETA`	Imposed vertical (y) stress.

Table 24: Mechanical conditions

Positions:

Common positions in two dimensions are:

`..position =`	Description
`LEFT`	Minimum horizontal (x) coordinates.
`RIGHT`	Maximum horizontal (x) coordinates.
`BOTTOM`	Minimum vertical (y) coordinates.
`TOP`	Maximum vertical (y) coordinates.
`BOTTOM_LEFT`	
`TOP_LEFT`	
`BOTTOM_RIGHT`	
`TOP_RIGHT`	

Table 25: Bounding box positions

In space-time finite elements, these all required the `_AFTER` suffix (as in `LEFT_AFTER`, etc).

8 Output

The `output` of the simulation is exported as a table containing for each time step average values of different fields.

The `inclusions`, `edge` and `point` sub-items can all be defined multiple times (for example to get average strains or stresses in both the matrix and the inclusions).

In the main `output` item (respectively any of the `inclusions`, `edge` and `point` sub-items) the `field` sub-item can be defined multiple times.

.output	Def.	Description
..instant	NOW	Instant at which the values are extracted. Use `AFTER` in space-time finite elements.
..file_name	output	Path to the file in which the results are stored. **The file will be overwritten at each simulation.**
..field		Name of the field.
..inclusions		Allows to output the average value of some fields over a certain phase.
...index	0	Index of the phase over which the average is carried. 0 is the matrix (`.sample`), 1 the inclusions (`.inclusions`).
...field		Name of the field.
..edge		Allows to output the average value of some fields over an edge of the sample.
...position	BOTTOM	Indicates which edge is selected.
...field		Name of the field.
..point		Finds the local value of some fields at a given point.
...x	0	x coordinate where to measure the selected fields.
...y	0	y coordinate where to measure the selected fields.
...field		Name of the field.

Table 26: Output parameters

The fields are ordered in the output table in the same order as they are defined in the initiation file. Fields can be any standard AMIE field. The most common are:

..field =	Columns	Comments
DISPLACEMENT_FIELD	2	
STRAIN_FIELD	3	
STRAIN_RATE_FIELD	3	
MECHANICAL_STRAIN_FIELD	3	Strain field not accounting for visco-elastic and imposed strains
REAL_STRESS_FIELD	3	
EFFECTIVE_STRESS_FIELD	3	Stress field not accounting for the damage
PRINCIPAL_STRAIN_FIELD	2	
PRINCIPAL_STRAIN_RATE_FIELD	2	
PRINCIPAL_MECHANICAL_STRAIN_FIELD	2	
PRINCIPAL_REAL_STRESS_FIELD	2	
PRINCIPAL_STRESS_FIELD	2	
SCALAR_DAMAGE_FIELD	1	
GENERALIZED_VISCOELASTIC_STRAIN_FIELD	variable	Size depends on the viscoelastic model (3 + 3 for each dashpot in the rheological assembly).

Table 27: List of common output field types

9 Export

The `export` consists in one mesh file per time step numbered from 1 to the number of time steps in the simulation. Each file describes a series of fields (the same series is used in each file).

`.export`	Def.	Description
`..instant`	`NOW`	Instant at which the values are extracted. Use `AFTER` in space-time finite elements.
`..file_name`	`export`	Basic path to the files in which the results are written. `"_i"` will be added to the name of each file, with `i` the index of the time step. **The files will be overwritten at each simulation.**
`..field`		Name of the field.

Table 28: Export parameters

The `field` subitem can be defined multiple times. It can be a field from the list of the output field types (see above), or one of the list below.

`..field =`	Columns	Comments
`TWFT_STIFFNESS`	1	Value of the 1111 component of the stiffness tensor.
`TWFT_VISCOSITY`	1	Value of the 1111 component of the viscosity tensor.
`TWFT_CRITERION`	1	Value of the fracture criterion used to evaluate the damage.

Table 29: List of common export field types

10 Mechanical behaviours

The `behaviour` items describes the mechanical behaviour of the matrix and the inclusions. It is defined by its `type` and a certain number of parameters or subitems.

`.behaviour`	Def.	Description
`..damage_model`		A **DamageModel** item which indicates which algorithm is used to compute the damage.
`..fracture_criterion`		A **FractureCriterion** items which define the failure surface of the material. The material does not fail if the criterion is not defined.
`..plane_type`	`PLANE_STRESS`	Indicates which 2D approximation is used.
`..poisson_ratio`	Req.	The Poisson ratio of the material.
`..young_modulus`	Req.	The Young's modulus of the material.

Table 30: Behaviour parameters

The different mechanical behaviours implemented are listed below. The default material behaviour is the **logarithmic creep** behaviour.

Logarithmic creep:

This behaviour requires the use of space-time finite elements. It is able to consider purely-elastic materials (no logarithmic creep), imposed deformation, as well as damage, depending on the parameters used for its definition.

This is the default behaviour, so it does not need a type. If you need to specify it, use `.behaviour = LogarithmicCreepWithExternalParameters`.

.behaviour	Def.	Description
..creep_characteristic_time		Optional. Indicates the delay before the creep reaches its logarithmic regime (in days). This parameter is required to activate the creep properties of the material (otherwise it will be considered as elastic).
..creep_modulus		Optional. Describes the rate of the uni-axial creep deformation in the logarithmic scale.
..creep_poisson		Optional. Defines the ratio between axial and lateral creep deformations. If this parameter is not found, `poisson_ratio` is used instead.
..damage_model		Optional. The material will not exhibit damage if either `damage_model` or `fracture_criterion` are omitted.
..fracture_criterion		Optional. The material will not exhibit damage if either `damage_model` or `fracture_criterion` are omitted.
..imposed_deformation	0	Value of the linear expansion in absence of external mechanical restraints. Use 0 for a non-expanding material.
..plane_type	PLANE_STRESS	
..poisson_ratio	Req.	
..young_modulus	Req.	

Table 31: Parameters for logarithmic creep material behaviour

Empty behaviour:

Elements with a `VoidForm` behaviour are not part of the solid.

.behaviour = VoidForm	Def.	Description

Table 32: Parameters for empty material behaviour

Elastic behaviour:

This behaviour is not compatible with space-time finite elements.

.behaviour = Stiffness	Def.	Description
..plane_type	PLANE_STRESS	
..poisson_ratio	Req.	
..young_modulus	Req.	

Table 33: Parameters for elastic material behaviour

Elastic quasi-brittle behaviour:

This behaviour is not compatible with space-time finite elements.

.behaviour = Stiffness	Def.	Description
..damage_model	Req.	
..fracture_criterion	Req.	
..plane_type	PLANE_STRESS	
..poisson_ratio	Req.	
..young_modulus	Req.	

Table 34: Parameters for elastic quasi-brittle material behaviour

Elastic behaviour with imposed deformation:

This behaviour is not compatible with space-time finite elements.

.behaviour = StiffnessWithImposedDeformation	Def.	Description
..imposed_deformation	Req.	Value of the linear expansion in absence of external mechanical restraints.
..plane_type	PLANE_STRESS	
..poisson_ratio	Req.	
..young_modulus	Req.	

Table 35: Parameters for elastic with imposed deformation material behaviour

Elastic behaviour with imposed stress:

This behaviour is not compatible with space-time finite elements.

.behaviour = StiffnessWithImposedDeformation	Def.	Description
..imposed_stress	Req.	Value of the volumetric stress induced in the material.
..plane_type	PLANE_STRESS	
..poisson_ratio	Req.	
..young_modulus	Req.	

Table 36: Parameters for elastic with imposed stress material behaviour

10.1 Damage model

Two damage models are used depending on the type of finite elements.

Isotropic damage model:

This damage model is used for standard finite element calculations

..damage_model = Isotropic	Def.	Description
...residual_stiffness_fraction	0	Stiffness of entirely damaged elements (expressed as a fraction of the stiffness of non-damaged elements).

Table 37: Parameters for the isotropic damage model

Space-time fiber-based damage model:

This damage model is used for space-time finite element calculations

..damage_model = SpaceTimeFiberBasedIsotropic	Def.	Description
...damage_increment	0.1	Value by which the damage is increased at each step of the algorithm.
...maximum_damage	0.99	Value above which elements are considered entirely broken.
...residual_stiffness_fraction	0	Stiffness of entirely damaged elements (expressed as a fraction of the stiffness of non-damaged elements).
...time_tolerance	0.001	Minimum time interval between two damage events. This value is relative to the current time step (unit-less) and must be lower than 1.

Table 38: Parameters for the space-time fiber-based damage model

10.2 Fracture criterion