Main

README file for the void generating algorithms referenced in Najmeddine and Shakiba (2020): Impact of Void Morphology on the Mechanical Response of Time-Dependent Heterogeneous Media: A Numerical Investigation Approach, ASCE Journal of Materials in Civil Engineering. https://doi.org/10.1061/(ASCE)MT.1943-5533.0003252

This repository contains codes that are designed to generate random voids with differing geometry and content level within heterogeneous microstructures depending on the desired output.

Each opens and reads the COORDSALLAGG.xls file containing the nodes' (x,y) coordinates of the entire sample microstructure to be plotted. Random void geometries are then subsequently generated until the target void content is reached. The COORDSALLAGG.xls can be obtained from either one of the available methods to extract particles' coordinates (e.g., AUTOCAD).

Voids are checked to not cross the aggregate particles and to not intercross one another.

Upon completion, the code will output a file, the name of which is included in the README files for the respective codes.

Copy the contents of the file(s) and paste them into the sample python script provided.

This python script will then be run in Abaqus CAE to generate the corresponding microstructure sketch.

Circles

This code generates voids of circular geometry.

The input parameters for the AirVoidGenerationScript for circular geometry are as follows in order:

M: Coordinates of particles read by Matlab from the COORDSALLGG.xls file.

Nstep: a parameter controlling the number of vertices in each circular void. A higher selected value of Nstep will result in more circular shapes, whereas a lower value will tend to create a polygon of Nstep # of vertices.

Target Air Void Percentage: The percentage of voids the code will generate up to.

The functions in the script are as follows:

FindPolygons: This function plots the aggregates from the COORDALLGG.xls file. The function ensures that each polygon drawn from the coordinates file is a separate entity.

GenerateCircleLogNormalDistribution: Generates the random voids.

isAnyPointinPolygon: Check if the generated void intersects with an aggregate. The output of the function is either "True", indicating that the void does intersect with the aggregates and should

therefore be neglected, or "False" indicating that the void does not intersect with any aggregate and should therefore be run through the second void intersection check.

IsAnyAirVoidInResultAirVoids: Check if the generated void intersects with any of the already generated voids. The output of the function is either "True", indicating that the void does intersect with the other voids and should therefore be neglected, or "False", indicating that the void does not intersect with any void and should therefore be selected.

The code will output a file named "***Origin_of_Void_Circles***" containing the origins and radii of all the successfully generated voids written in Python syntax. This is the script that will be run in Abaqus CAE to sketch the generated void morphology. This is done because Abaqus CAE only understands syntax written within Python libraries.

Ellipses

This code generates voids of elliptical geometry.

The input parameters for the AirVoidGenerationScript for Elliptical Geometry are as follows in order:

M: Coordinates of particles read by Matlab from the COORDSALLGG.xls file.

Nstep: a parameter controlling the number of vertices in each elliptical void.

Target Air Void Percentage: The percentage of voids the code will generate up to.

The functions in the script are as follows:

FindPolygons: This function plots the aggregates from the COORDALLGG.xls file. The function ensures that each polygon drawn from the coordinates file is a separate entity.

GenerateEllipseLogNormalDistribution: Generates the random voids.

IsanyPointinPolygon: Checks if a point is plotted within an aggregate and will delete the generated point if so.

IsAnyAirVoidInResultAirVoids: Checks if a point is plotted within a generated air void and will delete it if so.

This code will output a file named "***Origin_of_Void_Ellipses*** " containing the origins and the two semi-major and semi-minor radii of all the successfully generated voids written in Python syntax. This is the script that will be run in Abaqus CAE to sketch the generated void morphology. This is done because Abaqus CAE only understands syntax written within Python libraries.

Polygons

This code generates voids of polygonal geometry.

The input parameters for the AirVoidGenerationScript for Polygonal Geometry are as follows in order:

M: Coordinates of particles read by Matlab from the COORDSALLGG.xls file.

Irregularity: Controls how irregular each polygon is. Takes values between 0 and 1. A value of 0 will yield a perfectly regular polygon, whereas a higher irregularity value will increase the variance in the distance between the origin of the corresponding regular polygon and the polygon's vertices.

Spikyness: Controls the angles at the vertices. A higher value will create smaller angles at the vertices.

Alpha: Controls the elongation of the voids. A higher value will make the voids less elongated and more compact.

Numverts: Controls the number of vertices each void has. For instance, a value of *numvert* equal to 6 will yield a hexagonal shape.

Offset: Controls the limit to which the voids can be close to each other.

The functions in the script are as follows:

FindPolygons: This function plots the aggregates from the COORDALLGG.xls file. The function ensures that each polygon drawn from the coordinates file is a separate entity.

GenerateRandomPolygonVoid: Main function. Generates the random voids and outputs the files to be imported into Abaqus.

IsanyPointinPolygon: Checks if the void particle intersects with an aggregate and will delete the generated void if so.

IsAnyAirVoidInResultAirVoids: Checks if the void particle intersects with an already generated air void and will delete it if so.

generatePolygonFunction: Create the (X,Y) coordinate vectors of a void particle with the geometric properties outlined above.

generatePolygon: Modifies the generated void so as to eliminate any extremely-elongated shapes. Considers the *alpha* parameter.

Target: Air Void Percentage: The percentage of voids the code will generate up to.

This code will generate 2 output files named Vertices_of_Polygones_Voids and Line_connector_of_Polygones_of_Vertices_of_voids in the folder.

This code will output two files named "***Vertices_of_Polygones_Voids***" and "**Line_connector_of_Polygones_of_Vertices_of_voids**" containing the vertices (x,y) coordinates as well as the coordinates for the line segments connecting the vertices together of all the successfully generated voids written in Python syntax. This is the script that will be run in Abaqus CAE to sketch the generated void morphology. This is done because Abaqus CAE only understands syntax written within Python libraries.