Workflow DFNMesh – iMRS

This document presents in an objective way the workflow to perform DFM simulations using the research codes DFNMesh – for mesh generation – and iMRS – for multiscale FEM simulation.

# Preliminaries: Installing DFNMesh and iMRS

Both these codes can be cloned from GitHub:

* DFNMesh: <https://github.com/PedroLima92/dfnMesh>
* iMRS: <https://github.com/labmec/iMRS/>

Both codes rely on the NeoPZ environment that needs to be installed first from GitHub:

* NeoPZ: <https://github.com/labmec/neopz/>

DFNMesh has the extra following dependency:

* Gmsh SDK: <https://gmsh.info/>

iMRS has the extra following dependencies:

* libinterpolate: <https://github.com/CD3/libInterpolate>
* DFNMesh: <https://github.com/PedroLima92/dfnMesh>

# Defining fracture location and initial coarse mesh for DFN

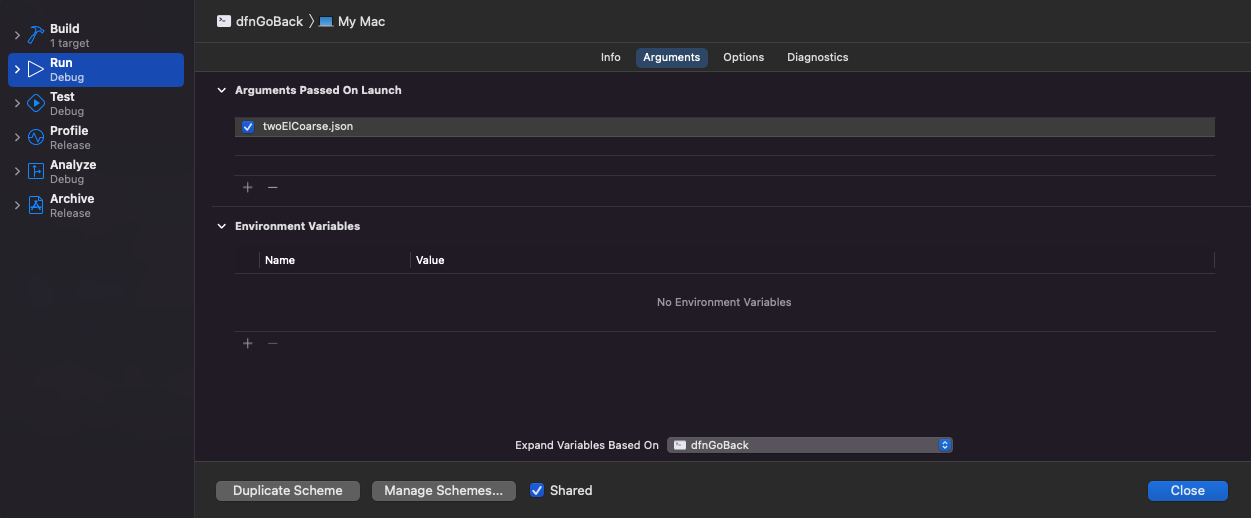
DFNMesh takes as input a coarse mesh in the GMSH format and the polygons that define the fractures in the domain. With that, it generates a fine mesh where the fractures are located at the interface of 3D elements. This procedure is done based on the intersection of the provided polygons with the initial coarse mesh.

Next is a figure showing a sample minimal input file for DFNMesh. See comments in the file itself for explanation of each variable:

Graphical user interface, text

Description automatically generated

Next, this file should be passed as argument to the executable of DFNMesh. In Xcode, press cmd+shift+, and fill the section with the JSON file name – in this case the file is called twoElCoarse.json. Also, don’t forget to set the working directory in the Option tab to the parent folder of DFNMesh repository.



By running the executable dfnGoBack, a GMSH file with .geo extension is generated with the name dfnExport.geo at the parent folder of DFNMesh. This file needs then to be opened in the software Gmsh and a screen similar to the next figure should appear:

Chart, radar chart

Description automatically generated with medium confidence

As indicated in the figure itself, the next steps are to generate the 3D mesh with the button “3D”, and save it to the computer with the button “Save”. This will generated a file named dfnExport.msh in the parent folder of DFNMesh.

Next, the necessary files will be exported to a folder in the iMRS project to be used for the multiscale FEM simulation.

# Importing DFNMesh generated files to iMRS and running the simulation

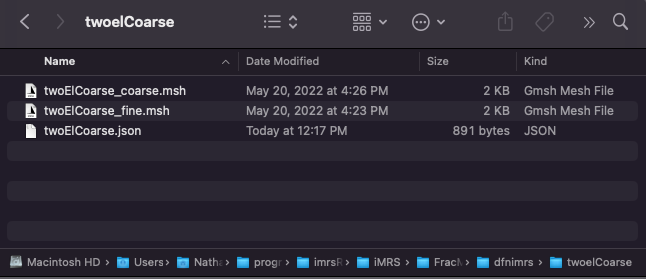
iMRS takes 3 files to perform a simulation. First is a coarse mesh GMSH type file which is usually the one provided to DFNMesh as coarse mesh. Second is a fine mesh which is the one saved on the computer as dfnExport.msh as explained in the end of the last section. And third is a JSON type of file that contains all the parameters for the simulation. This last file is usually built on top of the JSON file provided to DFNMesh.

First, let’s copy the 3 files to a folder in iMRS parent folder. The location I use in this tutorial is:

Graphical user interface, application

Description automatically generated

In this folder, let’s rename all files such that they all have the same initial string. In this case, I will use twoElCoarse. So, the file names are:



Next, let’s edit the JSON file to encompass all the necessary parameters. Next is a figure with the final version and explanations as comments next to the variables themselves. The new variables from the last section are: Domains, NCoarseGroups, Boundary, and the fracture properties K, phi, and width.

Text

Description automatically generated

Note that NCoarseGroups can be inferred by the number of physical names c0, c1, c2, … in the file twoElCoarse\_fine.msh. So, if a mesh has c0 and c1, it has 2 NCoarseGroups.

Last step is to set this set of files to be run in iMRS. For this, choose the target DFNIMRS in Xcode and edit the main() function in dfnimrs.cpp such that there is a simcase with the path to the base filename chosen for the simulation – in this case twoElCoarse. Thus:

Text

Description automatically generated