**1-Introduction:**

This package is an open-source nonlinear transient structural mechanics solver written in Python (FEniCS). The package couples growth and remodeling (G&R) of a biological tissue/structure with the transient dynamic of the structure. Specifically, we developed a solver to couple the transient dynamic of an aortic valve with the growth and remodeling of the aortic valve due to aging and calcification for multiscale modeling of disease growth.

The solver runs with MPI and interfaces, through FEniCS, to the state-of-the-art linear algebra backends like PETSc. To be able to work with this package, a user should know basic knowledge in how to solve PDEs through FEniCS, and basic programming skills in Python. A good introduction to the FEniCS could be obtained by following the tutorial:

<https://fenicsproject.org/tutorial/>

This code is used in the following publication:

**2- FEniCS installation:**

To run the solver, users need to install FEniCS, an open-source finite element software, written in Python: the best reference to install FEniCS is:

<https://fenicsproject.org/download/>

This solver is compatible with FEniCS 2017, 2018 and 2019. However, it is recommended to install this version of FEniCS.

$ conda create -n fenicsproject -c conda-forge fenics=2019.2

$ source activate fenicsproject

**3- Nonlinear transient solver installation:**

1. The solver can be installed simply by cloning the GitHub repository to your own computer:

$ https://github.com/amir-cardiolab/valve-growth.git

$ cd code/

1. You can just download the package from the link below:

$ https://github.com/amir-cardiolab/valve-growth

$ cd code

**4- Files and folders:**

The solver has a folder named “code”, which includes six python codes and the geometry of our problem.

All of the codes are implemented in Python and the main executable code name is *main\_loop.py.*

To run the solver, you need to preset the assumptions:

1. *Flags.py*  includes some of the settings used in the solver e.g., which version of FEniCS you need to use. By turning on or off the flags the settings may be changed. All of the flags have comments in the code and are explained in the code.
2. *functions.py*  includes some of the functions, which are used in the solver.
3. *Parameters.py* includes all of the parameters used in the solver. The time integration, the material properties, the circumferential and normal directions for growth and remodeling are implemented in this file. You can change the material properties such as the constitutive equation, the constitutive equation constants, time integration method, and parameters.
4. *continuum.py* includes the G&R equations and Cauchy’s equation of motion. Also, the weak form of Cauchy’s equation is assembled in this code.
5. *main\_loop.py* is the executable code, which is included the main loop where you can couple the transient structural dynamics with G&R.

To run the solver you need to run the *main\_loop.py:*

$ python main\_loop.py

If you want to run in parallel (recommended), you need to run the code as:

$ time mpi -n n\_processors python main\_loop.py

In the above “n\_processors” is the number of processors you want to use in parallel (MPI).

**5- Post processing:**

You can see the results in the open-source software **Paraview**. To download the software, you can use the link below:

<https://www.paraview.org/download/>

To see the displacement and the growth of the aortic valve, you need to use

“**warp by vector**” option in **Filters🡪alphabetical🡪warp by vector**:

Graphical user interface, text

Description automatically generated

Figure 1- Warp by vector is an option in Paraview which the user can see the displacement calculated by the solver.

Figure-2 is an example of the visualization of displacement of the valve which is the output of the solver:

Graphical user interface, text, application

Description automatically generated

Figure 2- Displacement visualization at max systolic