Linear Elasticity Tutorial 2

Mohd Afeef Badri

Abstract

This document details a single tutorials of 'linear elasticity' module of PSD in a more verbos manner.

To showcase the usage of Linear elasticity, we shall discuss here an example of a 2D bar, which bends under its own load. The bar 5×1 m^2 in area is made up of material with $\rho = 8 \times 10^3$, $E = 200 \times 10^9$, and $\nu = 0.3$. Contrary to tutorial 1, now both ends of the bar are clamped.

Step 1: Preprocessing

First step in a PSD simulation is PSD preprocessing , at this step you tell PSD what kind of physics, boundary conditions, approximations, mesh, etc are you expecting to solve.

In the terminal cd to the folder /home/PSD-tutorials/linear-elasticity. Launch PSD_PreProcess from the terminal, to do so run the following command.

```
PSD_PreProcess -problem linear-elasticity -dimension 2 -bodyforceconditions 1 \
-dirichletconditions 2 -postprocess u
```

After the PSD_PreProcess runs successfully you should see many .edp files in your current folder.

What do the arguments mean ?

- -problem linear-elasticity means that we are solving linear elasticity problem;
- -dimension 2 means it is a 2D simulation;
- -bodyforceconditions 1 with applied body force acting on the domain;
- -dirichletconditions 2 says we have two Dirichlet border;
- -postprocess u means we would like to have ParaView post processing files.

Since basic nature of both the problems (the one from tutorial 1 and 2) is same the almost the same command for preprocessing used in previous tutorial 1 is used here. The only difference,is that an additional Dirichlet condition needs to be supplied, notified to PSD by -dirichletconditions 2. To provide Dirichlet conditions of the left clamped end ($u_x = u_y = 0$) in ControlParameters.edp set Dbc0On 2, Dbc0Ux 0., and Dbc0Uy 0.. Similarly, for the right end set variables Dbc1On 4, Dbc1Ux 0., and Dbc1Uy 0. Each one of these is a clamped border respectively labeled as 2 (Dbc0On 2) and 4 (Dbc1On 4) in the mesh ../Meshes/2D/bar.msh.

Just like the previous tutorial the input properties E, ν should be mentioned in ControlParameters.edp, use E = 200.e9, and nu = 0.3;. The volumetric body force condition is mentioned in the same file via variable Fbc0Fy -78480.0, i.e ($\rho * g = 8.e3 * (-9.81) = -78480.0$). One can also provide the mesh to be used in ControlParameters.edp , via ThName = "../Meshes/2D/bar.msh" (*note that mesh can also be provided in the next step*) .In addition variable Fbc0On 1 has to be provided in order to indicate the volume (region) for which the body force is acting, here 1 is the integer volume tag of the mesh.

Step 2: Solving

As PSD is a parallel solver, let us use 3 parallel processes to solve this 2D bar case. To do so enter the following command:

```
PSD_Solve -np 3 Main.edp -mesh ./../Meshes/2D/bar.msh -v 0
```

Here -np 3 denote the argument used to enter the number of parallel processes (MPI processes) used while solving. -mesh ./../Meshes/2D/bar.msh is used to provide the mesh file to the solver. -v 0 denotes the verbosity level on screen.PSD_Solve is a wrapper around FreeFem++ or FreeFem++-mpi. Note that if your problem is large use more cores. PSD has been tested upto 13,000 parallel processes and problem sizes with billions of unknowns, surely you will now need that many for the 2D bar problem.

Step 3: Postprocessing

PSD allows postprocessing of results in ParaView. After the step 2 mentioned above finishes. Launch ParaView and have a look at the .pvd file in the VTUs_DATE_TIME folder. Using ParaView for postprocessing the results that are provided in the VTUs... folder, results such as those shown in figure~1 can be extracted.

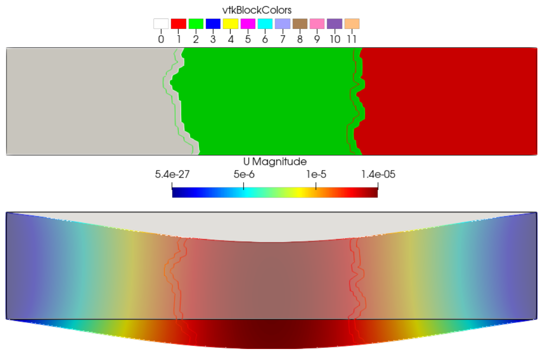

Figure 1: The 2D clamped bar problem: partitioned mesh and displacement field visualization in ParaView.

You are all done with your 2D linear-elasticty simulation.

What else should you try to become an advanced user

Try running the 3D problem. Keep in mind to rerun the PSD_PreProcess with -dimension 3 flag and using the appropriate mesh via -mesh flag with PSD_Solve. It goes without saying you will need to adjust the Dirichlet border labels in ControlParameters.edp.

Since gravity is the main force involved in the problem, try redoing the test with different gravitational constant. Imagine, you wish to know how the test would compare if performed on Moon and Jupiter. The only thing that will change now is the gravitational pull, for Moon $g = 1.32$ and for Jupiter $g = 24.79$. To perform the moon test simply change Fbc0Fy -10560.0 in ControlParameters.edp and redo step 2 and step 3. Similarly, for the Jupiter test Fbc0Fy -198320.0 in ControlParameters.edp and redo step 2 and step 3.

Optionally try using -withmaterialtensor flag with PSD_PreProcess , and run the simulation. You are encouraged to have a look at ControlParameters.edp and VariationalFormulations.edp file produced with -withmaterialtensor flag and without this flag.

Add -sequential flag to PSD_PreProcess for sequential solver, but remember to use PSD_Solve_Seq instead of PSD_Solve and no -np flag.

```
PSD_PreProcess -problem linear-elasticity -dimension 2 -sequential \
-bodyforceconditions 1 -dirichletconditions 2 -postprocess u
```

```
PSD_Solve_Seq Main.edp -mesh ./../Meshes/2D/bar.msh -v 0
```

Advance exercise 1

There is a solver run level flag for mesh refinement [1]. This flag is called -split [int] which splits the triangles (resp. tetrahedrons) of your mesh into four smaller triangles (resp. tetrahedrons). As such -split 2 will produce a mesh with 4 times the elements of the input mesh. Similarly, -split n where n is a positive integer produces 2^n times more elements than the input mesh. You are encouraged to use this -split flag to produce refined meshes and check, mesh convergence of a problem, computational time, etc. Use of parallel computing is recommended. You could try it out with PSD_Solve or PSD_Solve_Seq, for example:

```
PSD_Solve -np 4 Main.edp -mesh ./../Meshes/3D/bar.msh -v 0 -split 2
```

for splitting each triangle of the mesh bar.msh into 4.

[1] Mesh refinement is performed after partitioning.

Advance exercise 2

There is a preprocess level flag -debug, which as the name suggests should be used for debug proposes by developers. However, this flag will activate OpenGL live visualization of the problems displacement field. You are encouraged to try it out

```
PSD_PreProcess -problem linear-elasticity -dimension 3 -bodyforceconditions 1 \
-dirichletconditions 2 -postprocess u -timelog -debug
```

Then to run the problem we need aditional -wg flag

```
PSD_Solve -np 4 Main.edp -mesh ./../Meshes/3D/bar.msh -v 0 -wg
```

Advance exercise 3

There is a preprocess level flag -withmaterialtensor, which introduces the full material tensor into the finite element variational formulation. You are encouraged to use this flag and see how the sollver performs.

```
PSD_PreProcess -problem linear-elasticity -dimension 3 -bodyforceconditions 1 \
-dirichletconditions 2 -postprocess u -timelog -withmaterialtensor
```

Then to run the problem we need aditional -wg flag

```
PSD_Solve -np 4 Main.edp -mesh ./../Meshes/3D/bar.msh -v 0
```

To understand what the flag does, try to find out the difference between the files created by PSD_PreProcess when used with and without -withmaterialtensor flag. Especially, compare ControlParameters.edp and VariationalFormulations.edp files produced by PSD_PreProcess step.

Advance exercise 4

You are encouraged to use more complex meshes for this same problem, but do not forget to update the ControlParameters.edp file, with your desired boundary conditions.