Linear Elasticity additional exercises

Mohd Afeef Badri

Abstract

This document details a additional exercises of linear elasticity. These exercises are recomeneded to be followed only after other tutorials of linear elasticity have been followed.

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/2D/bar.msh -v 0 -split 2
```

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

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 2 -bodyforceconditions 1 \
-dirichletconditions 1 -postprocess u -timelog -debug
```

Then to run the problem we need additional -wg flag

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

Advance Exercise 3

One interesting way of solving a linear Elasticity problem is to solve it via a pseudo nonlinear model. There is a preprocess level flag -model pseudo_nonlinear, which introduces pseudo nonlinearity into the finite element variational formulation of linear elasticity. You are encouraged to use this flag and see how the solver performs. Indeed, now you should see some nonlinear iterations (1 or 2) are taken for convergence.

```
PSD_PreProcess -problem linear_elasticity -dimension 2 -bodyforceconditions 1 \
-dirichletconditions 1 -postprocess u -timelog -model pseudo_nonlinear
```

Then to run the problem

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

[1] Mesh refinement is performed after partitioning.

To understand what the flag does, try to find out the difference between the files created by PSD_PreProcess when used with and without -model pseudo_nonlinear flag. Especially, compare LinearFormBuilderAndSolver.edp and VariationalFormulations.edp files produced by PSD_PreProcess step. You will see Newton–Raphsons iterations are performed for solving the linear problem. However, the nonlinear iterations loop converges very rapidly (in 1 iteration) due to linear nature of the problem. **Note:** This flag is exclusive for parallel solver.

Advance exercise 4

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 solver performs.

```
PSD_PreProcess -problem linear_elasticity -dimension 2 -bodyforceconditions 1 \
-dirichletconditions 1 -postprocess u -timelog -withmaterialtensor
```

Then to run the problem

```
PSD_Solve -np 4 Main.edp -mesh ./../Meshes/2D/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 FemParameters.edp, MeshAndFeSpace and VariationalFormulations.edp files produced by PSD_PreProcess step.