Elastodynamics Tutorials - Timelogging elastodynamic simulations

Mohd Afeef Badri

Abstract

This document details some tutorials of elastodynamics module of PSD. These tutorials are not verbose, but does instead give a kick start to users/developers for using PSD's elastodynamics module.

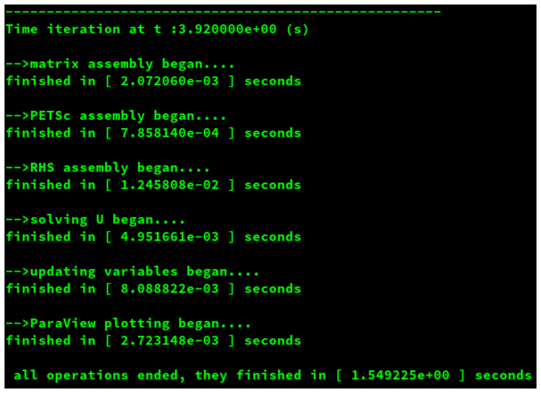
PSD provides mean to time log your solver via -timelog flag. What this will do when you run your solver, on the terminal you will have information printed on what is the amount of time taken by each step of your solver. Warning, this will make your solver slower, as this action involves MPIbarrier routines for correctly timing operation.

An example work flow of 2D solver with timelogging:

```
PSD_PreProcess -dimension 2 -problem elastodynamics -dirichletconditions 1 -tractionconditions 1 \
-timediscretization newmark_beta -postprocess uav -timelog
```

Once the step above has been performed, we solve the problem using two MPI processes, with the given mesh file bar-dynamic.msh.

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

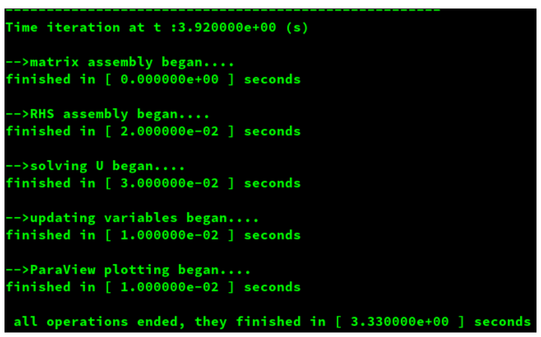


```
----------------------------------------------------
Time iteration at t :3.920000e+00 (s)

-->matrix assembly began....
finished in [ 2.072060e-03 ] seconds

-->PETSc assembly began....
finished in [ 7.858140e-04 ] seconds

-->RHS assembly began....
finished in [ 1.245808e-02 ] seconds

-->solving U began....
finished in [ 4.951661e-03 ] seconds

-->updating variables began....
finished in [ 8.088822e-03 ] seconds

-->ParaView plotting began....
finished in [ 2.723148e-03 ] seconds

 all operations ended, they finished in [ 1.549225e+00 ] seconds
```

Figure 1: Time logging output produced for parallel run on 2 processes.

The figure~1 shows the time logging output produced for parallel run on 2 processes using -timelog flag. Similar output is produced for sequential solver of the same problem shown in figure~2. Take note of the speed up, which should be two folds - parallel solver solves the full problem in half the time (1.5 sec) than that of sequential solver (3.3 sec). This is due to the fact we used 2 MPI processes.

Also take note of timings produced for different operations of the solver. Note that in figures~1, 2, we only see the final time step of the solved problem.



```
------------------------------------------------------
Time iteration at t :3.920000e+00 (s)

-->matrix assembly began....
finished in [ 0.000000e+00 ] seconds

-->RHS assembly began....
finished in [ 2.000000e-02 ] seconds

-->solving U began....
finished in [ 3.000000e-02 ] seconds

-->updating variables began....
finished in [ 1.000000e-02 ] seconds

-->ParaView plotting began....
finished in [ 1.000000e-02 ] seconds

 all operations ended, they finished in [ 3.330000e+00 ] seconds
```

Figure 2: Time logging output produced for parallel run on 2 processes.