GMSH Workshop: Post-processing

C. Geuzaine and A. Modave

September 19, 2011

In this workshop we will learn how to use Gmsh plugins to perform increasingly complex post-processing operations on datasets.

1 Compute and display streamlines around the Star Trek Enterprise

We will first learn how to use a single post-processing plugin to compute streamlines:

- Open the enterprise data set: `gmsh enterprise.pos`.
- You should see the mesh and a vector field.
- In Tools>Plugins select the `Streamlines` plugin.
- Define a grid of 10 by 10 particles to be traced, placed in front of the Enterprise (e.g. $(X_0, Y_0, Z_0) = (-100, -500, -300)$, $(X_1, Y_1, Z_1) = (100, -500, -300)$, $(X_2, Y_2, Z_2) = (-100, -500, 50)$).
- Adjust the time step ($DT = 100$) and run the plugin over 1000 iterations (the plugin uses a RK44 scheme).
- Use Tools>Visibility to display the boundary of the Enterprise, and Tools>Options to change the display of options. For example:
  - Hide the volume elements;
  - Select the last time step of the trajectory view;
  - Draw the trajectories as 3D cylinders.
- Use Tools>Clipping to hide half of the Enterprise.

Save the display options for the current file with File>Save Options>For Current File: this will create `enterprise.pos.opt`.

Plugins can also be used in scripts. A good way to start is to click "record" in the plugin window, and run the plugin again: this will save the plugin commands in the same option file.

Now quit Gmsh and edit `enterprise.pos.opt`. See how plugin options can be set with the following syntax: `Plugin(pluginName).optionName = value;` and how Plugins can be run with `Plugin(pluginName).Run;`.

Adjust the option file and run `gmsh enterprise.pos`.

Save a MPEG movie of the moving particles with File>Save As (simply choose a file with ending with a `.mpeg` extension).

2 Assemble scalar datasets into a vector view

The following scalar views created by the DG code contain the time evolution of the x and y components of a velocity field: `SubmarinePml_*_COMP_1.msh` and `SubmarinePml_*_COMP_2.msh`.

Write a script to combine these scalar views into a single vector-valued view using `Plugin(MathEval)`. Hint: look up `w0` and `OtherView` in the plugin help tab.

Display the resulting vector view using lines segments 10 pixels long, colored by the magnitude of the velocity on a logarithmic scale.

3 Compute global quantities of interest from the output of the DG code

The computation of sound scattering by a submarine created the following datasets:

- `SubmarinePml_*_COMP_0.msh` and `SubmarineRef_*_COMP_0.msh` contain the pressure $p(t)$ for the reference and the perfectly matched layers (PML) solutions, respectively.

- `SubmarinePml_*_COMP_{1,2}.msh` and `SubmarineRef_*_COMP_{1,2}.msh` contain the x and y components of the velocity field $\vec{u}(t)$.

Write a script to compute the relative error on $p(t)$ and $\vec{u}(t)$, in energy norm, between the reference and the PML solution (Hint: use `Plugin(Integrate)`). The energy norm is defined as:

$$\int_\Omega \left(\frac{1}{2\rho} p^2 + \frac{\rho c^2}{2} |\vec{u}|^2 \right) d\Omega,$$

with $\rho = 1000 kg/m^3$ and $c = 1500 m/s$. This error should be computed only in the area inside the PML, i.e., inside the inner ellipse. You can achieve this by

- (easy) setting the inner region as visible in Tools>Visibility and using `Plugin(ExtractElements)` with the `Visible` option set to 1.
- (medium) by making creative use of the `SubmarineCache_*_COMP_0.msh` field
- (harder: C++ coding required) modifying `Plugin(CutSphere)` to create ellipsoids instead of spheres. Use `Plugins/CutSphere.cpp` as a template!

Display the evolution of the relative error vs. time as a 2D plot (Tools>Options>View>General>Plot type).