Lab 01 Getting Started and step-1

Computational Methods for PDEs Summer School 2019

---

1. Virtual Machine setup
   - Download and install "VirtualBox" for your machine. See https://www.virtualbox.org/wiki/Downloads and choose "windows host" (if you are running windows), "OS X hosts" (if you are running MacOS), or install it under Linux (depends on the distribution).
   - Download the virtualbox image linked at http://www.math.clemson.edu/~heister/dealvm/. The .ova file is 3.6GB big so complete this step before traveling to Fort Collins.
   - Double-click the .ova file and "import" it into virtualbox. When prompted pick the amount of RAM and number of cores (if unsure pick 2 cores and 4GB of RAM unless you know if your machine has more). You will need in the order of 20GB of disk space.
2. The Virtual machine
   - The machine is a light-weight Ubuntu with various things installed. If needed, you can install additional software. You can use "synaptic" to do that.
   - To get the latest files for this workshop run `dealvm-install pdeschool2019` in a terminal window. This will create a directory `~/pdeschool2019`. Rerunning the command will update to the latest version.
   - The following directories are of interest:
     - `pdeschool2019/` – Here you can find the lecture notes, exercises, codes, etc.
     - `deal.II/installed/` – This is the installation directory of deal.II, you probably don't need to access it directly though.
     - `deal.II/dealii` – The deal.II source directory.
     - `deal.II/dealii/examples/` – all tutorial programs.
     - `libs/` – libraries deal.II depends on.
   - to make a copy of tutorial 1, configure, compile, and run it:

     ```
     cp -r ~/deal.II/dealii/examples/step-1 ~/
     cd ~/step-1
     cmake .
     make
     ./step-1
     ```

   - IDE: open `qtcreator .`
3. Tasks for tutorial step-1:
   1. See documentation at https://www.dealii.org/current/doxygen/deal.II/step_1.html
   2. `~/pdeschool2019/lab01/step-1/` already constains a clean version of step-1 for you to work with (we removed the extensive comments using `make strip-comments` in the file, see above to find the original version).
   3. Compile and run inside qtcreator and look at the output files. You can view `.eps` files with `gv <filename>`, for example `gv grid-1.eps`.

4. Looking at .eps files is somewhat clunky. Switch to outputting `.svg` files instead as they open in firefox (can you guess the function name?).
5. Add the line `triangulation.reset_manifold(0);` after the call to `::hyper_shell`. We already added the line commented out in the modified version of step-1. What happens now? Why are only the corners refined?
6. Create an image of an L-shape domain (add a function third_grid() to step-1) with one global refinement.
7. Refine the L-shaped mesh adaptively around the re-entrant corner several times but with a twist: refine all cells with the distance between the center of the cell and re-entrant corner is smaller than 1/3.
8. Bonus: Create a helper function that takes a reference to a Triangulation and prints the following information: number of levels, number of cells, number of active cells. Test this with all of your meshes.
9. Bonus: Create a mesh that represents the surface of a torus and refine it 2 times globally. Output to vtk format and check the output. Note that your Triangulation needs to be of type `Triangulation<2,3>`, which we probably do not discuss this week.
10. Bonus: Take a look at step-49 and read the included .msh file in your modified step-1 program.