MATH 762: Individual Project for Bryn Barker

February 18, 2021

Your project consists of writing a Maxwell solver, writing a Navier-Stokes solver, and then getting them to talk to each-other. This is pretty ambitious. I admire that you want to try and get things working with general conservation laws but writing an MHD solver is already going to be pretty hard, so while you should keep that in mind in planning phases, I think just getting a multiphysics solver working is already more than enough for a semester-long project. The major things you need to implement are:

- A time-dependent Maxwell solver.
- A time-dependent Navier-Stokes solver.
- Some code that gets them to talk to each-other.

I am moderately familiar with how the first two should go - I am less familiar with how the coupling should occur. I suspect that you will ultimately have to choose between a monolithic scheme (evolving both systems together by solving a single linear or nonlinear system at each time step) or with a partitioned (or staggered) scheme. It's easier to write down a stable monolithic scheme than a stable partitioned scheme, but its a lot harder to implement a monolithic scheme. Even if you are writing the two solvers independently you should keep this in mind and it should effect how you design the interfaces.

There is more than one way to do it but the most common method for solving Maxwell's equation is in their curl-curl form. Fortunately, deal.II has a very good implementation of a modified Nedelec element, `FE_NedelecSZ`, that works pretty well. Be aware that we also have `FE_Nedelec`, which doesn't work with complex geometries and has some serious performance problems - I recommend ignoring that implementation of the Nedelec element. I believe Ross Kynch and a few other deal.II users have used `FE_NedelecSZ` in publications - someone has some notes on the topic in the deal.II wiki under "Electromagnetic problem". You will want to use functions like `project_boundary_values_curl_conforming` to get the right answer with these elements. `step-62` models a wave equation but implements a PML boundary condition - you might want to look there if you are interested in implementing something like that.

Since you are interested in doing long time integrations you should keep an eye out for schemes with good conservation and dispersion properties. This is also a place where using AMR could help a lot (so you can resolve waves to high accuracy). I am not positive that the Nedelec elements work with AMR, though.

I'm not aware of any more code that's publically available and actually solves Maxwell's equations with deal.II. I think Matthias Maier might have some unpublished code - we can ask him if you get stuck.

For the Navier-Stokes solver: there are more people doing fluid mechanics so there are a variety of examples available. It is also not that hard to pick a stable finite element pair. Try starting with `step-35`. Martin Kronbichler wrote a massively parallel Navier-Stokes solver for his PhD thesis - try looking up `adaflo` to see that implementation. I recommend using a projection method or something else that is simple and second-order accurate, but make the code general enough that you can upgrade it later. It would be great if you could use geometric multigrid as a preconditioner for the Laplace solve, but our geometric multigrid framework is very hard to understand unless you are already an expert. There are two ways around this:

- Use PETSc and through our PETSc bindings use HYPRE for AMG.
- Look at deal.II PR number 11699 - another option would be to port this class once it is done into your code so that you can use geometric multigrid without exposing every single option available.

This project is subject to modification by mutual agreement between you and I. The goal is to have a project that fits your research interests. If you would like to deviate from the path outlined here, please discuss any changes first with me so we can update our plan.

Project tasks: As part of your project, you will need to meet the following milestones:

- *Milestone 1: March 25, 2021*
  - Review the literature and decide what algorithms you want to implement for solving Maxwell and Navier-Stokes independently. Concurrently, you need to also figure out what a stable coupling scheme would look like. The coupling scheme should influence your choice of solvers for the individual problems. You should have a clear description of time integration, boundary conditions, stability conditions, etc.
  - Implement the proposed Maxwell solver. Your design should be flexible enough that you can make it talk to the yet-unwritten Navier-Stokes solver.
- *Milestone 2: April 22, 2021*
  - Write the Navier-Stokes solver.
  - Get the two solvers working together.

Deliverables (end of the semester):

- Your final report, which will build on the description you wrote for Milestone 1. This should describe what you can simulate with your code and why.
- Your finished, documented code and all input files necessary to run it, checked in to a GitHub repository to which I have access.

Grading: I will determine your grade (out of 50 points) based on the following criteria:

- (10 pts) Clear documentation and code formatting. Variables should be named relevantly and consistently. Code should be neatly indented (e.g., with `astyle` or `clang-format`). Classes and functions should have at least a sentence or two explaining both *what* they do and *why* they do it.

  This documentation doesn't have to be perfect but it should be enough to guide someone through the program. I will help you clarify the design and documentation over the course of the semester. You will have a good idea of what I expect by Milestone 2.
- (15 pts) Documentation around the program, describing the problem that it solves and the motivation for doing so, as well as a general description of the code's structure and motivation behind its organization. Like the first point, this will evolve over the course of the semester and you will have a clear idea of what I expect by Milestone 2.
- (15 pts) The extent to which you finished your Milestone 1 goals.
- (10 pts) The extent to which you finished your Milestone 2 goals.