Ultrasound Simulation Package Update

By Adam Gleichman

Credit: Pixabay; https://www.pexels.com/photo/water-drop-photo-220213/

Abstract

The Wave Simulator code is meant to be a proving ground for new functions that my group can add to our ultrasound simulation package. Our lab created an ultrasound simulation package for other labs to use in their experiments called the Fast Object-Oriented C++ Ultrasound Simulation (FOCUS). To improve our package with a very modular and fast way to simulate shear waves in the 3D space, we plan to use a numeric approach which is similar to methods like the Finite Difference and the Finite Volume Methods called the Discontinuous Galerkin Method [1]. This method has the advantage as a better means to parallelize our code which can take advantage of super computers to evaluate behavior of waves in 3D space with much more accuracy and detail.

Methodology

This project was meant to be difference than the suggested layout of the project because the project is written entirely in MATLAB and that this project is not used for a specific experiment. The functions written are meant to become add-ons to an ultrasound simulation package my lab group works to update for general use. This library is written such that we do not tailor our code for a specific experiment instead trying to make the library as broad as we can within the field of ultrasound simulation to maximize the reach of our software. Our code is written in teams so writing in MATLAB is a given standard within our group for communication and because of MATLAB's simple plotting tools to make debugging easier on our end by using plot functions for us to check, for example, stability of our function. Sphinx, cookie-cutter, and unit test packages that were recommended by the course were not

used in the project because they are meant for Python. Despite that, I took time to try to use MATLAB's equivalents to try to replicate the functions. I used the published HTML report files from the MATLAB publish function in the IDE as a substitute for Sphinx auto documentation. I was happy with this equivalent and I think this works well. I used MATLAB's project setup system to try to format like a cookie cutter which worked fine without impeding the documentation. MATLAB's form of unit testing was difficult; in the end, I was not able to get a basic test function to correctly recognize which function to test.

My original project proposal was written under the assumption that my advisor and I would be working on a Finite Volume Method algorithm for numerically modeling ultrasound waves. After 3 weeks of trying to understand and complete a one-dimensional wave being modelled with Finite Volume Method; my advisor and I decided the best course of action was to switch from Finite Volume Method to Finite Difference Method. We were motivated to do so because of my advisor is more familiar with the Finite Difference and we thought that we could use the opportunity to use Finite Difference Method to learn more about modeling the ordinary differential equations. This decision was reached around the beginning of November and in that time, I was able to program a working one and two-dimensional numerical simulation with Finite Difference.

Concluding Remarks

I appreciate the topics of the class, but the topics on how to be a better programmer were more impactful to me. I knew what documentation was and that keeping yourself organized in programming was important, but I did not know, or I was not familiar, with tools to help with that part of the programming process. Before this class I was happy to have working code, quick comments, and maybe some handwritten notes on my code that I would keep in a notebook for a project. Understanding GitHub and auto documentation tools was a big turning point because I was forced to better the structure of my own code to correctly get them to work. I learned that I had bad habits such as writing tests built into the code, not creating a modular setup with functions, not creating a standard for naming my code files, and just doing the bare minimum with error handling. I do think I have solved all these problems yet, but I am happy to know where I can improve. I have also taken the time to teach my other coworkers about MATLAB documentation and testing.

This project and the class have exceeded my expectations on improving me as a programmer, even though I was a little underwhelmed with the progress of the work I did with my research. All the work I have achieved I know I still repackage in some way to add some more value to our library even though it is not the end goal. I have a much better foundation to work with and I will use that when working the rest of my Finite Difference Method simulations before I work to the overall goal of creating a Discontinuous Galerkin Method algorithm for 3D shear wave simulation to process on the HPCC.

References

[1] J. Kelly, X. Zhao, D. Murray, S. Marras, and R. McGough, "Nonlinear ultrasound simulations using a time-explicit discontinuous Galerkin (DG) method," in *IEEE International Ultrasonics Symposium, IUS*, 2017.