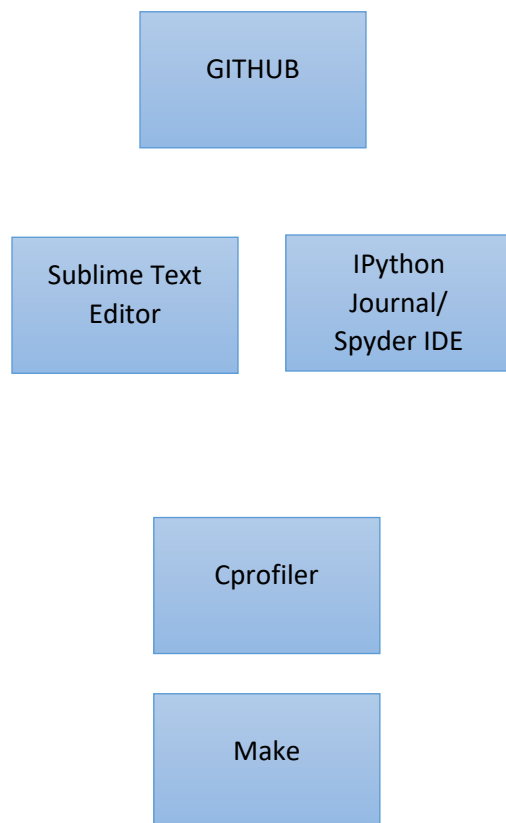


## IMPORTANCE OF A TOOL CHAIN

Having a proper tool chain helps a programmer ensure that they can conduct work in a professional manner, provides the tools necessary for collaboration with colleagues and team members, and ensures that the programmers time is spent as productively as possible. Professionalism in program involves using the right tools, at the right time, in the most productive manner. A properly developed tool chain, will ensure that the programmer has access to, and efficiently uses the correct tools. For example, properly managing GitHub, and the associated branches, allows a programmer to ensure version control, while simultaneously make changes to codes to test new experiments. As the code becomes more and more complex, this becomes a necessity. GitHub facilitates collaboration. GitHub is an amazing resource for programmers to manage projects with teams, as each team member could be given access to a repository and see the work, or comments of other programmers. Finally, the most important aspect of the tool chain, is how much easier it makes the programmers job. Having build tools, such as make, which automate the building of programs, as well as version control systems, IDE's, appropriate text editors, and auto documentation, are all designed to facilitate fast and productive work by the programmer. They automate simple task, and can give insight into writing better code, as is the case with a good profiler. Good debuggers can speed up the the discovery and fixing of any problems with the code.

## MY TOOL CHAIN




The beginning of my tool chain will be Github. As I write more code, and develop useful algorithms, I will reference previous work as I begin new projects.

From their most of my actual coding is done via Spyder, which has a useful debugging feature and allows code to be ran directly. Additionally, I use IPythons Journal to explore data. Finally, I just got familiar with Sublime, as it is the text editor in the journal. This text editor allows me to (1) view large data files, which is useful in checking my code, as well as write scripts.

I have not gotten to the point in my coding where I have used a profiler. I am open to recommendations, but as of now I am planning on using CProfiler, or MPI Profile.

Similarly, I have not gotten to the point where a build tool would make me more productive. However, I will use Make when it is time for me to build and test my final product.



Doxygen

I have done research on auto documentation programs, and it seems that Doxygen is in the lead, I have not used this product yet.

### **MY TOOL CHAIN IN THE FUTURE**

As I programmer I am rather undeveloped, consequently so is my tool-chain. I have done the research and can understand and appreciate how these tools increase productivity, and I am looking for opportunities to use them. Because of this, it is likely my tool-chain will go through many changes as I explore different products. I expect, especially if I begin programming frequently, my tool-chain may in fact gain complexity, as different products may be better for different applications. As of now, I have used the tools, which I feel have made me the most productive. My go to tool is Spyder, this is likely because that was the first product I used. I recently figured out how to use the Journal, and now that I know how to use it, I will be using it for my notes, and experimentations. The fall back on Spyder, was when I called for a plot, it came up as a tiny screen in my kernel. With the Journal, I am hoping to get better visibility on the data, which will allow me to make informed decisions.