Lab 6: Software Projects, Workflows, and Object Oriented Design

Matthew Duschenes

Department of Applied Physics, University of Michigan

October 12, 2020

1 Summary of Work

During the lab period, the class worked on developing infrastructure for SpMV sparse matrix-vector multiplication library. In groups, UML diagrams were developed for the base SpMV class, and it was determined that there would be attributes of the data of the matrix and vector (user input), the size of the matrix and vector; there would be modules of the functions to get and set the data, and to call the matrix-vector multiplication routine. There may be dependent subclasses of a sparse matrix and vector, with the data type (single or double precision), data shape etc.

An important distinction for this class, is the specific sparse matrix format (COO, CSR, ELLPACK) used for storing the data, and the associated routine for performing the multiplication. This requires abstraction of the base SpMV class, to allow for different data formats to be input, and different multiplication routines to be called.

Finally, tasks for this initial base set of requirements were added to the github infrastructure, through the creation of issues, and initial templates for some files, such as cmake files for compilation of dependencies, containing the determined subclasses. Issues were assigned labels, depending on their importance, and relevance to various major or minor goals.

2 Deliverables

2.1 What was learned?

By participating in the activities, several approaches about splitting up project design, both in terms of infrastructure, as well as in term of class design were learned.

When setting up the project infrastructure and creating issues and defining the main objectives and requirements, being as simple and specific as possible seems to be really effective at efficiently and succinctly ensuring all project aspects are clear and understood. Breaking up tasks, such as defining the large objective of creating a multiplication routine for a particular storage format, into short, even one sentence issues, such as create header (or cmake) template, or write documentation for this routine, or write tests, really simplifies things. Accomplishing many small goals ensures sequential progress, that each task is manageable, and that it is easy to keep track of where bugs may occur.

When setting up the design of the classes, it was learned that it is important to similarly make class attributes, modules, and dependent classes, as simple and stripped down as possible. Making clear, explicit attributes, such as separate data, data type, data size etc., makes this initial setup easier, the programming more simple, and the end use easier to learn and use.

2.2 Approaches taken to accomplish objectives?

To accomplish the objectives of setting up the project infrastructure, and the base class structure, the group used a combination of shared documents, comments in github, and polls. This allowed decisions to be made on how to structure project aspects, and to make some initial decisions on important matters such as the implemented programming language, and how to split up the issues.

2.3 Things that worked well?

Some aspects that worked well included the use of the shared documents to draw or write out the structure of certain objectives. This allowed for easy remote collaboration. The use of the github comments also allowed the group to quickly create and edit issues, and to get a good sense of the project objectives and main goals and timeline going forward.

2.4 Things that did not work well?

Some things that did not work so well include the general objectives of the lab itself, and what kind of objectives the group was hoping to define for the initial project outline. It was unclear how in depth or abstract the objectives or plans should be, and how we should structure the brainstorming sessions.

Further, once it came down to diving the labour up for explicitly writing down the issues and initial tasks to be completed in github, it was difficult to do this remotely, and to known exactly who was in charge of, and even to gain a consensus for what how the issues were divided into subtasks. There was confusion on how time should be spent on defining, or inputting these issues into the github system as well.

2.5 What should be improved?

To improve the above issues, and take advantage of the new skills learned of subdividing and simplifying as much as possible, initially creating sub-teams, in charge or smaller or broader initial tasks, or even ahead of time defining the issues at hand, would greatly help in performing this initials infrastructure remotely. Everyone would be able to contribute to defining the specific sub-tasks to be completed, and would get input and consensus on other sub-tasks if the teams each convened at the end to give a summary of what has to be accomplished. This would also help the github issues creation, and avoid group members being unsure on what has to be accomplished.