Workshop on Best Practices and Tools for Computational and Data-Intensive Research

Klaus Bartschat (Drake University) and Barry I. Schneider (NIST)

1. Background

The two authors are senior members of a group of computational atomic and molecular physicists who realized about twelve years ago how the NSF advanced computational infrastructure program could enable research progress far beyond what could be done on local resources. Over the years we have had collaborators and postdocs move in and move out as their interests changed, or when they became independent researchers themselves.

Looking back at the past decade, aside from obvious allocation of resources that enabled some very high-quality science, we were lucky to be able to interact with knowledgeable and engaged people at the NSF centers and via the XSEDE ECSS program. This helped in solving problems as simple as not linking to correct libraries to making some of our codes significantly more efficient.

2. A Pervasive Problem

In common with many research groups, a large fraction of the codes we use regularly go back in time to the 1970s, where programming languages were far less powerful and flexible. While these codes remain very useful in studying interesting science, they are difficult to understand, often poorly documented, and nearly impossible to modify by anyone who was not thoroughly immersed in the original developments.

Much of the above situation derives from the fact that the codes were written by graduate students and postdocs who were (understandably) more interested in doing the science than the coding. The programmers are generally not rewarded for the coding when it comes to finding a permanent position in the scientific area for which they write the program. Realistically, it is highly non-trivial to rewrite these codes and (too) expensive even if there is enough knowledge to proceed.

Practically, all we can do is try to document what is already in place and make it easy to use with developments using current programming practices. In some cases, intensive computational tasks can be isolated and made more efficient using parallel programming techniques and/or libraries.

3. Improving Practices

Any group containing even a modest number of researchers working together has to develop a set of practices that allow the group to work smoothly and symbiotically. Although our particular group has not always been a shining example either, we can make some general points that hopefully will be useful overall.

- Code needs to be placed in a repository such as Github or Bitbucket, and users must be responsible. A repository allows a user or a group to return to earlier versions, and the use of branches can help prevent serious issues from developing.
- Use libraries that have been developed by others where possible. This leads to efficiencies in time to solution.
- Try to write code that is re-usable.
- **Documentation.** If a code is not well documented, it becomes useless as time goes on. If one gets into the habit of adding comments directly in the code, this may be very helpful to other developers.
- Object oriented programming(OOP) has been a C++ standard for many years. Other languages such as modern versions of Fortran have OOP capability and defined types as part of the language. Even though these features are useful, they have not been widely adopted in many communities.

4. An Explicit Example: Gateway for Atomic and Molecular Physics

One of us (BIS) was a co-organizer of a workshop entitled "Developing Flexible and Robust Software in Computational Atomic and Molecular (A&M) Physics". The intent of that workshop, which was held in May of 2018 at the Institute for Theoretical Atomic and Molecular Physics **(ITAMP)** of the Harvard-Smithsonian Center for Astrophysics, was to bring together a group of internationally known researchers to work collectively on making their codes available and easier to use by the partners as well as others. The discussion focused on:

- Identifying and prioritizing outstanding problems in A&M science, which would benefit from a concerted community effort in developing software tools, algorithms, and approaches that would lead to more rapid and productive scientific progress for the entire community.
- Approaches to achieving that goal.
- Producing and disseminating a report of the workshop to the community.

The success of the workshop led some groups [Barry I. Schneider, NIST (USA); Klaus Bartschat & Oleg Zatsarinny, Drake University (USA); Igor Bray, Curtin University (Australia); Armin Scrinzi, Ludwig-Maximilians Universität, (Germany); Fernando Martín & Markus Klinker, Universidad Autónoma de Madrid (Spain); Jonathan Tennyson, University College London(UK); Jimena D. Gorfinkiel, The Open University(UK)] to write an **XSEDE** proposal to build a science gateway to host the codes of these groups. That proposal was supported both in the form of supercomputer time and, critically, personnel [Sudhakar Pamidighantam, Indiana University(USA)]. Since May of 2018 the group members have been meeting approximately every other week via Zoom to ensure continued progress. A number of the codes are already ported and running on various XSEDE platforms. Some progress has been made in making them usable by others within the group but not yet the outside world. We are now taking steps towards achieving this last goal by planning a second workshop in December of 2019.

At present, the group has achieved the following:

- Created the **AMP** gateway, including a first attempt at a visually pleasing webpage.
- The codes are running in command line mode on a few XSEDE platforms (Comet and Stampede2).
- We started creating a GUI for the codes.
- An invited paper was submitted and accepted to the **PEARC** conference.
- **MOLSSI** and the Applied and Computational Mathematics Division of NIST have agreed to provide support, and we expect additional funds from the National Science Foundation.

The AMPGateway uses a multi-tenanted framework **Apache Airavata middleware** served by the **SciGaP** hosting services for sustained operation. In the first stage of our efforts the software suites were deployed as independent applications with specific input interfaces. Community building (there was enthusiastic response at DAMOP-2019 when our work and intentions were presented at the TAMOC meeting as well a throughout the conference by some of the group members) has already started and a few additional software suites have been identified for inclusion in Phase Two. The interoperability of the software suites is very important and will be addressed as a follow-up.

We expect the program of the second workshop to consist of a set of tutorial talks on each of the codes, followed by breakout sessions where interested attendees can get more detailed information and actually go to the gateway to perform some simple calculations. One of the tutorial talks will focus on the gateway itself. It is important that users do not have to deal with the details common to command-line Linux but instead can use the power of the gateway to make progress.