Lab Notebooks for Computational Mathematics, Sciences & Engineering

Date: December 14, 2022

Presented by: Jared O'Neal (Argonne National Laboratory)

(The slides are available under "Materials from the Webinar" in the above link.)

Q. How do you manage your digital lab notebooks so that they can satisfy the legal requirements you mentioned? Blockchain would be one option, but is probably overkill... Do publicly hosted DVSs function appropriately? (Search caches etc. would provide some insurance.)

A. Unfortunately I do not have any experience related to using lab notebooks in a legal sense. The Nature article that I cited in the talk touches briefly on this topic. I do know that some institutions have policies set forth to govern the use, maintenance, and storage of lab notebooks. Perhaps your institution might be able to help determine the best means for using lab notes in this way.

Q. In an experiment, one often has many false starts and makes many mistakes. He may include all those in the lab notebook. Later he may find the right answer and wish to correct the previous mistake in his lab book. What is the best way to make corrections? Obviously we do not tear pages off the lab book. Should we cross out the mistakes in the lab book? or to annotate?

A. Lab notes are intentionally comprehensive and frozen at creation**.** One motivation for this is to limit the possibility of scientific fraud. Another reason is to avoid retroactive filtering of the content, which could itself be based on a mistake or short sightedness. Also, including the full history of exploration and mistakes is part of the scientific record and there is potential benefit in being able to trace and learn from this later. For paper notes, the technique that I learned was to cross out text with a single line to communicate clearly that there was a mistake, but to allow others to still read what was written. This allows for full transparency. If I determined that some early content was later proven wrong and could potentially be misleading, I would oftentimes note this in the margin and refer to a specific page that discusses why. Note that this is an informal way of using the notes that might invalidate some of their other uses (e.g., as a legal device). Such corrections and annotations are something that I struggle to do well with digital tools. It raises other questions. If notes are supposed to be frozen at creation, is it OK to manage them and update them within a version control system? All changes are logged, but the fact that notes have been altered is not immediately obvious.

Q. I feel like some of this filtering or summary of noisy notes will become the remit of AI tools like ChatGPT. Thoughts?

A. While knowledge can be embedded in lab notes, it is not necessarily embedded there in an obvious or explicit way. It takes some work when rereading the notes to identify it and piece it together. When this knowledge is then recorded in another document, we do not necessarily record it verbatim. For instance, more recent experiences might first allow us to improve, polish, or discard that knowledge. In addition, as we aggregate knowledge into a new document, we might generate more knowledge or, more importantly, generate understanding. Therefore, the act of filtering a lower level of documentation can be a hugely important activity for the human being and I appreciate the importance of that human capturing the experience in a document. I would be happy to use a tool if it can genuinely and correctly help me with that process. However, I would personally like to be heavily involved in this important and rich activity.

Q. What do you recommend to make it easy for someone to search through your notebooks? Do you have a living contents page?

A. I use high-level READMEs as a sort of contents or roadmap to indicate the role played by each notebook at the scale of an experimental phase or the whole study. Part of this is trying to figure out exactly what each notebook should do or what role it plays in the scientific analysis. If the READMEs are compact, hopefully finding a particular notebook is not too hard. If I have structured well all the READMEs involved in the study, hopefully I can follow a chain of references without too much difficulty. I find that managing such chains and cross-references is a particularly important but difficult aspect of digital lab notes. Within a notebook I try to use thoughtful formatting for a manual visual search or using the browser's Find command. Also, many of my notebooks use a similar structure. Certainly a living contents section at the top could be useful if your notebook is large.

Q. I noticed that you dumped your environment and package versions.. Do you have experience with conda and environment management? Is it useful or a rabbit hole? Any caveats to sharing with others and getting up and running?

A. I use python for most everything except for heavy computational work, which is done by compiled binaries. My limited experience with Anaconda is that it creates and provides a full software environment. I would be happy to use this if my work is restricted only to python. However, in my work I need to be careful about managing how a binary's build and execution environments mix with the environment setup by Anaconda. Please see slide 6 in 08-computational-expts.pdf for an example of a possible difficulty when combining the environments. I tend to be conservative and prefer simplicity. Therefore, I stick with manually-constructed, minimal python installations as virtual environments managed solely with pip. This is possible since I don't suffer much from suboptimal performance. Ideally I will include within my virtual computational laboratory environment a means for me and others to setup the exact python environment. I will certainly study the following comment to see if that can help me manage environment difficulties.

C. A comment about conda: we have been using Anaconda Project to manage our environments, basically using conda to provision the underlying python environment, (and any binary libraries like CUDA, FFTW etc.), but then including a minimal pip section to provision the virtual environment. This allows one to create a localized virtual environment for the project, but extends the regular conda environment specification with the ability to specify different architectures, etc. We combine this with a Makefile to reproducibly provision the environment on a variety of platforms. This is all still in development, but we have a set of cookiecutter templates one can use to quickly provision a project with these tools, including integration with continuous integration tools on GitLab and GitHub to run tests, automatic generation of documentation hosted on Read The Docs (see e.g. notes from a course Physics 581: Physics Inspired Computational Techniques generated from Jupyter Notebooks using Jupyter Book) and deployment on the virtual CoCalc platform where I interact with my students on shared coding projects in real time. The system is still rough and a little over-complicated, but I am quite happy with where it is going. If anyone is interested in these techniques, please feel free to contact me and we can discuss further: m.forbes+hpc@wsu.edu.

Q. Do you have any advice for combining Anaconda and pip/virtual environments? I currently have a project which is in pip/virtual environments but it conflicts with a user's Anaconda environment.

A. Yes: see above:-) I use anaconda-project. The person using anaconda can create an anaconda-project.yaml file that has any conda-specific dependencies they need (i.e. to get python on their system, with libraries etc.) and then just have a pip section that pulls in all of your pip requirements. This would create an isolated project-specific environment with the virtual environment installed.
One way we do this is to have all of the pip dependencies separate (i.e. in a pyproject.toml file). We then provision the pure python dependencies from this so that people who install python without conda can also install the package, but generally, we have everyone use anaconda-project.

The simple solution here is to create a localized conda environment with nothing but python, then using pip to install into that. For example:

```
conda create -p path/to/myenv python=3.9
conda activate path/to/myenv
pip install …
```

This will create an isolated conda environment located in `path/to/myenv` into which pip will install everything. Thus, it acts like a virtual environment, but allows the user to install other things with conda that might be difficult otherwise (CUDA toolkits are one example).