Lab Notebooks for Computational Mathematics, Sciences, & Engineering

David E. Bernholdt (he/him)
Oak Ridge National Laboratory

Software Practices for Better Science: Testing, Reproducibility, and Documentation tutorial @ Exascale Computing Project Tutorial Days

Contributors: David E. Bernholdt (ORNL), Jared O'Neal (ANL)

Thanks to: Juan Pablo Haddad, Akash Dhruv, Anshu Dubey, Steve Fickas, Carlo Graziani, Boyana Norris

U.S. DEPARTMENT OF ENERGY | Office of Science

NNSA National Nuclear Security Administration

License, Citation and Acknowledgements

License and Citation

- **The requested citation the overall tutorial is:** David E. Bernholdt, David M. Rogers, and Gregory R. Watson, Software Practices for Better Science: Testing, Reproducibility, and Documentation tutorial, in Exascale Computing Project Tutorial Days, online, 2023. DOI: 10.6084/m9.figshare.21989507.
- Individual modules may be cited as *Speaker, Module Title*, in Software Practices for Better Science: Testing, Reproducibility, and Documentation tutorial, …

Acknowledgements

A minimal definition of a lab notebook

A goal of keeping a lab notebook is "...to write with enough detail and clarity that another scientist could pick up the notebook at some time in the future, repeat the work based on the written descriptions, and make the same observations that were originally recorded. If this guideline is followed, even the original author will be able to understand the notes when looking back on them after considerable time has passed!"

- Howard Kanare, *Writing the Laboratory Notebook*

Aren't lab notebooks only for experimental scientists?

Scientific HPC is several young fields that "on close examination, have not really stabilized or optimized their collaborative processes in a manner analogous to that of more mature, 'classical' sciences. As a consequence, valuable science is often needlessly lost, or left uncollected."

- Carlo Graziani, Computational Scientist at ANL, *HPC and the Lab Manager*

"… practicing the scientific method properly requires good software practices. This is understood in the experimental community…The computational science side has had a historic problem with it. As we can see, it's getting better."

- Katherine Riley, Director of Science at ALCF, ATPESC 2019

DIKUW

Data, Information, Knowledge, Understanding, Wisdom

A classification scheme that overloads everyday words so that we can use the same language and understand each other.

We will build on this to understand & appreciate documentation & lab notebooks.

DIKW pyramid (unaltered) – Wikipedia/Longlivetheux

Data & Information

- Data
  - Collection of numbers, symbols, text, *etc.*
  - It has value only because it was recorded and exists.
  - **Example**: Timeseries representation of temperature, relative humidity, and precipitation.

- Information
  - Facts gleaned from data.
  - Answers questions such as who, what, when, how much, how long, *etc.*
  - **Example**: Starting at 2pm the temperature dropped by 5ºF over 15 minutes. At 2:05 pm it started to rain and 0.25" of rain was accumulated over the next 30 minutes.

Knowledge & Understanding

- Knowledge
  - Derived from information, experience, and **understanding**.
  - **Example**: When relative humidity levels are high and the temperature drops substantially, there is an increased probability of precipitation.

- Understanding
  - A deep theoretical background in and practical experience with the system whose data was acquired and studied?
  - The ability to explain why?
  - "Understanding is a kind of ecstasy" – Carl Sagan
  - **Example**: A meteorologist could explain at different levels of detail how the atmosphere works to substantiate the knowledge.

Obligatory Einstein quote

"Any fool can know. The point is to understand."

- Albert Einstein

- Is understanding always the ultimate goal?
- Are there times when mere knowledge is sufficient?
  - I don't need to understand atmospheric science or weather prediction. I just need basic knowledge to determine if I should take an umbrella with me.
- Data, information, and knowledge are key ingredients in developing understanding
  - Need to manage all, appropriately

IDE▲S
productivity

ECP EXASCALE COMPUTING PROJECT

Knowledge Management

- DIKUW is related to knowledge management

- We want to
  - **Recognize** when we generate knowledge
  - **Capture** and **preserve** that knowledge
  - **Communicate** that knowledge

- Scientific work can benefit from knowledge management

Is knowledge communication only about communicating to others?

Example: Lessons learned

- After we live through an experience, we want to derive lessons learned
  - Experience is valuable, more so if we reflect and are thoughtful
  - Generate/capture knowledge to grow, improve, and avoid difficulties/mistakes
  - Hope to improve understanding

- We get more if we derive lessons learned together
  - Create more or higher quality knowledge
  - Communicate the knowledge implicitly

Should we communicate lessons learned to others?

Which leads us to documentation

Documentation is knowledge communication & can build understanding
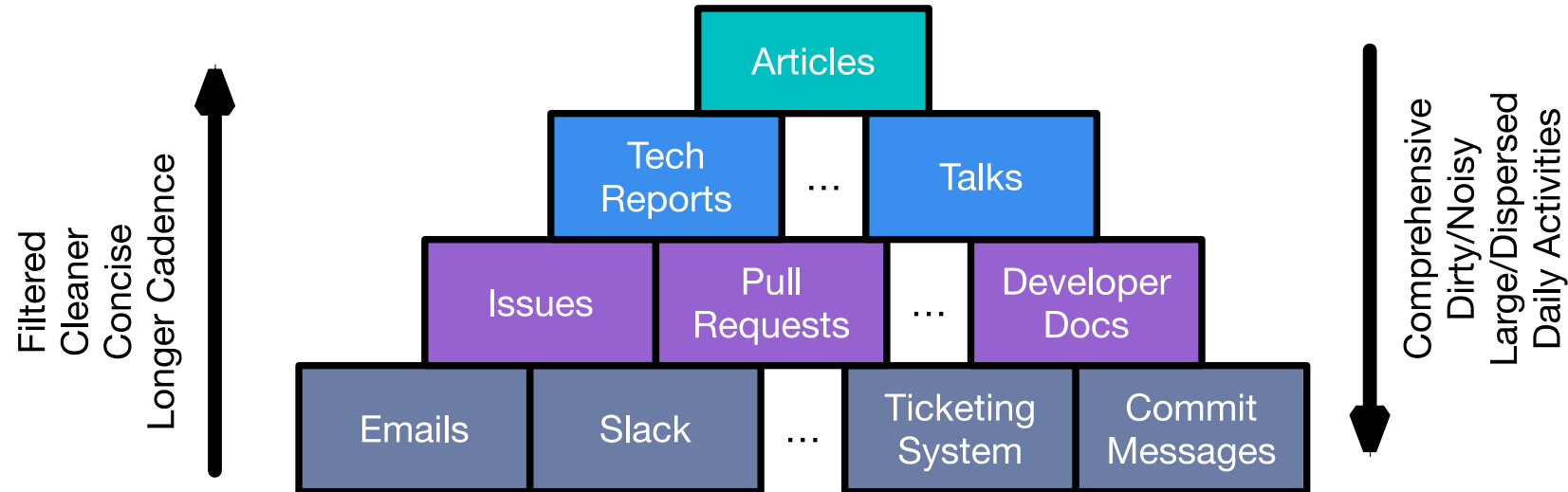
- Data at the bottom. Knowledge as we move up?

- Some documents frozen at creation. Others are living.

- Does this capture how hard it is to do documentation in a distributed, digital world?

And finally, we reach our destination

Lab notebooks are
- A fundamental part of communication as well as rigorous, reproducible science in a lab,
- A common-place or required part of an experimental laboratory,
- Populating a scientific "lab notebook" is increasingly an "automated" process in many modern experimental and observational research environments,
- A tool for preventing scientific fraud, and
- Record of invention and defending against allegations of fraud.

Lab notebooks
- Should be used regularly,
- Should be comprehensive and never filtered,
- Don't need perfect grammar and full sentences,
- Content is frozen at creation, and
- Hopefully contains more than just data (*e.g.*, motivation, reasoning, conclusions).

They aren't good at communicating knowledge, **but**
people interact, evolve, and grow by collaborating through the notebook.

Conversations with Carlo

Carlo Graziani is a Computational Scientist at ANL
BSSw blog article *HPC and the Lab Manager*

- As researchers' careers progress
  - The problems become more complex and larger
  - Previous informal techniques for executing a study start to fail
  - The researchers' sense that something is missing
    - *"Doing in-flight airplane repair"*
  - They invent processes and tools to compensate

  This happened to Carlo and at some point, he realized that
  "I had re-invented the lab notebook!"

No one likes writing lab notes

"Lab notes are a waste of time. I write notes, but never use them."
  - Almost all newcomers to lab notebooks

- This attitude comes from lack of experience & awareness
  - Good notes are implicit communication & sharing
  - Good notes can be turned into procedures

- Lab notes become more useful as time passes
  - Our memory fades
  - It can take years before we see the benefit

- Writing but not reading lab notes is a good thing
  - Lab notes are most useful when something has gone wrong

Example notebook entries

A bad example

Monday July 25, 2022
9:05 am - Do study ABC
8:47 pm - Lot's of interesting data!
          - Results are in GCE

A better example

Monday July 25, 2022 (Jared)
9:05 am - Continuing work for study ABC. (See July 7)
- I presently believe that if A happens, then B must also happen.
- To verify this, I intend to
   - …
   - ***

9:30 am - Started executing this experiment on Bebop.
   - Built debug version of binary with Intel 20.4
   - Based on <u>clean</u> commit 5a43b21c
   - Build log saved to `my_test_2022.log`
   - No errors or warnings emitted
   - Used job script `run_my_test` with configuration 24 (Job ID 123456)
   - Stdout/err & results saved in folder ABC

10:07 am - Analysis run with script `analyze_my_test.py` and results saved in same folder.
- Since no peak seen around 1.5 MeV, I was wrong. But based on this, I now *believe* that if A happens, then C must also happen.

IDEAS productivity   ECP EXASCALE COMPUTING PROJECT

Nothing beats good ol' pen and paper

"Since at least the 1990s, articles on technology have predicted the imminent, widespread adoption of electronic laboratory notebooks (ELNs) by researchers. It has yet to happen — but more and more scientists are taking the plunge."

- Roberta Kwok, *How to pick an electronic laboratory notebook*, Nature

Pen & Paper Pros	Electronic Woes
• Most can use paper and pen in any situation	• Tied to technology that could fail
• Open format can allow for creativity and easier annotation	• Overwhelming variety of possible solutions with different pros and cons
• Concentrate on the work rather than tooling	• Uncertainty about future of tool, increased costs, inability to export
• Good if notetaking slows down progress	• Does funding restrict where and how digital notes can be stored?
• Notebook is stored **publicly** next to where it is used	

Criteria for lab notebooks for computing?

- **Paper won't work.** We work anywhere and sometimes in distributed way.

- Should notebooks be public and how to do that?

- How many different types of notebooks do we need?

- Do we use a single ELN or distribute notes across a suite of tools?

- How can we use automation appropriately to overcome difficulties and increase productivity?

We likely need many streams of lab notes

Different streams of lab notes

- Lab notebook for changes to scientific instrument
  - Changes in code repo necessary for study
  - Changes to SW environments
  - Changes to build/job files and build systems

- Lab notebook for data analysis tools

- Lab notebook to detail how experiment was designed and executed

- Right tool for the job
  - We don't want a single 10,000-line README

Git lab notes stream

Keep lab notes for your software as close to the "instrument" as possible

```
Date:    Mon Jun 27 10:42:22 2022 -0500

    (Issue #215) Added in Milhoja Init unit test.  I have tried to structure this in
    accord with the unitTest architecture in the User Manual.  One main consequence
    of this is that it uses the generic Grid unitTest evolveAll routine, which
    writes a unitTest output file.  Since this test also uses the ut_testDriverMod
    framework, I had to simplify that so that it doesn't attempt to write the same
    file.  This change will likely affect the AMReX unit tests.

    I have run this successfully in 1D, 2D, and 3D on GCE/compute-12 with Intel.
    The test correctly creates a unitTest_XXXX output file and adds in a success
    line only if the test was 100% successful.  I temporarily dumped the ICs and
    final solution to AMReX-format files and manually confirmed correct content.
```

Details not obvious from commit diff:
Motivation, reasoning, consequences

Testing notes

*This message is missing a title as the first line.

Use Pull Request to capture final verification streams

Pull request as a "filtered" notebook entry

- A pull request is an aggregation of commits to a git repo
  - Individual commits are linked to the pull request

- The PR allows for additional content that's distinct from the individual commits

- Use PR description to record process to verify correctness of changes
  - 2-2.5 days effort carried out over a week
  - Copy/pasted from previous PR and adapted first (designed process)
  - Improved as carried out process – converging on a quasi-procedure
  - Filtered so that reviewers aren't overwhelmed
  - Helped organize effort & design good tests

- Additional benefits
  - Senior reviewers provide feedback & suggest improvements
  - Junior reviewers exposed to work habits of other people

- Example: Flash-X PR #247 on next slide

Example: Flash-X PR #247

jared321 commented on Aug 18 • edited ▾ `Author`

Working on GCE/compute-012 with

```
Currently Loaded Modules:
  1) intel/20.4   2) mpich/3.4.2-intel   3) hdf5/1.12.1-mpich-3.4.2-parallel-fortran
```

- All tests will be built in debug mode using the Intel/20.4 + MPICH + HDF5 stack loaded via GCE modules.
- The same pseudo-UG-only testing was carried out identically for the 2D/Sedov/simpleUnsplit and 3D/Sedov/Unsplit simulations with Paramesh, AMReX, and Milhoja. These two simulations were chosen as they will hopefully become official Milhoja Comparison tests in the GCE test suite.
- All 2D tests ran in 304 steps with 6 checkpoint files including the initial conditions; 3D, 61 steps and 2 checkpoint files.
- Ran the 2D tests with 4 MPI processes; the 3D, with 8 MPI processes. These are the number of processes presently used for Sedov tests in the GCE test suite.
- Confirmed with `sfocu` that the Paramesh and AMReX final checkpoints are identical and that the Paramesh and Milhoja final checkpoints are identical.
- Confirmed that the integrated quantities were conserved in each case up to a reasonable level and that the initial values were consistent across all associated runs to at least 12 decimal digits.
  - standard deviation of 2D conserved quantities less than 2.5e-14
  - standard deviation of 3D conserved quantities less than 3.25e-14
  - Confirmed that the initial values of the [xyz]-momenta were all exactly zero and that the z-momenta was exactly zero for each all time steps in the 2D simulations.
- Confirmed in Milhoja 2D case that we get, as expected, the identical final checkpoints if we run with 4 and 8 MPI processes.
- Where possible, the `milhoja.log` files were reviewed to confirm correct configuration and execution.
- Viewed the Milhoja final checkpoints in Visit to qualitatively confirm reasonable results for all variables and correct mesh overlays. 3D data was viewed with a slice at 50%.

README lab notes streams

- High-level road maps with motivation, documented decisions, and conclusions

- Concise living docs that function as executive summaries
    - Higher up in documentation hierarchy

- Low-level notes such as managing software stack
    - Traceability of SW environment & therefore verification
    - Can be turned into procedures

High-level README

My Study Executive Summary

This repository encapsulates the computational laboratory environment constructed and used for our 2022 My Study scientific research. Not only does it contain all tools needed to setup and use the environment, but it also contains the metadata and context needed to understand how data was acquired and how to use it for analysis and drawing conclusions.

Study Goals

Our motivation for this study is ...

We believe that ...

We intend to use our software to ...

To accomplish our goals we require that ...

We understand that our software is limited and therefore that our study is limited in the sense that ...

Some optional goals are ...

Ideally, we would like to publish our results in the Journal ZYX with a submission target date of ...

Organization

ABC will be the PI and will ...

XYZ is responsible for ...

AMReX Installation History

5 March 2020 - FlashFluxRegister b1dd083

- Using changes from FlashFluxRegister branch by Weiqun.
- Need for improved flux handling across all branches.
- Built 1D/2D/3D libraries for both gfortran and Intel at this commit.
- These were tested with the Staged/Staged-Intel/Master compute00x test suites with no other changes made. All tests passed.

6 November 2019 – master – 93fb085d

- Austin encountered an AMRex based bug and reported it in Issue #643. He reports that AMRex fixed that bug at commit - 23f943f.
- New install location is /nfs/proj-flash5/ a shared group at MCS, we are no longer going to use the /sandbox/flash/ versions on compute00[123]
- Built 1D/2D/3D libraries for both gfortran and Intel at this commit.
- These were tested with the Staged/Staged-Intel/Master compute00x test suites with no other changes made. All tests passed.

29 March 2019 – development – 06c6c0e2

- Rebuilt the libraries from a commit that contains the multifab changes implemented manually in the previous libraries. These changes were merged in to AMReX at commit 66392f8d on Tue Mar 26. No code in AMReX needed modifying for this build.
- These libraries were also built with Particles and linear solvers enabled for Saurabh's tests. He showed me that I had to update the automagically updated makefile to get the Particles Fortran interface into the libraries.
- Built 1D/2D/3D libraries for both gfortran and Intel at this commit.
- These were tested with the Staged/Staged-Intel/Master compute00x test suites with no other changes made. All tests passed.

20 March 2019 – master – 884d3194 (modified)

- Updated the files AMReX_multifab_fi.cpp and AMReX_multifab_mod.F90 at commit to incorporate the local tile index in the Fortran interface. These changes had already been tested during development in local repositories.
- Built 1D/2D/3D libraries with both gfortran and intel on compute001. These were then copied over to compute002 and compute003.

Low-level README

Capturing data context & metadata

- Automate as much as possible
- Build dates, user, system name, git hashes, configuration data in file headers
  - Self-documenting files
- Build & job logs
  - Software environment info (e.g., modules, `ldd` output)
  - git diffs
  - Environment variables

Jupyter notebooks

The exception to the rule?

- Jupyter notebook can put context & metadata next to data
  – High-level design & motivation up top
  – Low-level lab notes for acquiring data
  – Load & use data
  – Generate visualizations in place
  – Comment on results

- Where do notebooks fit into the documentation hierarchy?

- Repetitive use of notebook?
  – Limit amount of code in notebook

How to organize your "virtual" (multi-stream) lab notebook?

- It should be
  - Easy to create and maintain lab notes
  - Easy to concentrate more on executing work & less on documenting it
  - Easy to find what you need

- Each stream should
  - Have a clear identity for what it records
  - Not contain notes contained in other streams
  - Be recorded by using the right tool for the job

This implies the need for a documentation scheme.

Consider: design your lab notebooks into a larger execution environment.

Summary

- Software best practices are foundational science & are mandatory

- Knowledge management can improve science & productivity

- Productivity can arise from selflessness

- Not all documents are alike

- Not all lab notebooks are alike

- Lab notebooks are mandatory

- Lab notebooks allow for learning

- Lab notebooks for CMSE are hard

- We want to construct an execution environment

Resources

A Sampling of Execution Environments

- ATPESC 2022 Laboratory Environment BSSw Tutorial – Example Repo

- Lab Notebooks for Computational Mathematics, Sciences & Engineering, Best Practices for HPC Software Developers webinar series, December 2022

- Popper – 2018 BSSw Fellow Ivo Jimenez

- FlashKit – Aaron Lentner (George Washington University)

- Code Ocean

- Weight & Biases (machine learning)

References from slides

- Howard M. Kanare, *Writing the Laboratory Notebook*. American Chemical Society, Washington, D.C., 1985.

- Carlo Graziani, *HPC and the Lab Manager*. **Better Scientific Software**. https://bssw.io/blog_posts/hpc-and-the-lab-manager. Nov 17, 2021.

- Katherine Riley, *What All Codes Should Do: Best Practices*. ATPESC 2019 presentation. Retrieved from YouTube. Nov 5, 2019.

- DIKW pyramid. 2022, August 4. In *Wikipedia*. *https://en.wikipedia.org/wiki/DIKW_pyramid*.

- Roberta Kwok, *How to pick an electronic laboratory notebook*. **Nature** 560, pp. 269-270, Aug 6, 2018.