Status of Software Preservation in Planetary Science

C. Million (Million Concepts),
A. Brazier, A. Hayes (Cornell),
T. King (UCLA)

Email: chase.million@gmail.com

For this talk, I'll take as given that:

- Research software should be preserved.
  - Because software == methodology.
  - Because software is expensive.
- Preserving source code and documentation is necessary but not sufficient.
  - Getting code to run without context is really hard.
  - Understanding functionality without poking at it is really hard.
  - Function is often emergent or context dependent.
- Not all software can be saved. Some will be lost.
  - But we should do what we can now.

If you disagree, please see me at any time to discuss.

Since the 2nd Data Workshop:

- Code sharing has become a requirement of some funding mechanisms (e.g. PDART) and is more generally encouraged.
- More recognition within field that code sharing is possible and desirable.
- PDS policies wrt code have been clarified.

Policy on Software Archiving: “Source code that is sufficiently documented may be submitted as documentation for or an example of some processing algorithm. It will be subject to the same review and standards requirements as any document submitted for archiving.”

ROSES16/17 guidance on software:

"The DMP should also cover any other data and software that would enable future research or the replication/reproduction of published results. Software, […] should be made publicly available when it is practical and feasible to do so and when there is scientific utility in doing so. […] NASA expects that the source code, with associated documentation sufficient to enable the code's use, will be made publicly available via GitHub (https://github.com/NASA-Planetary-Science), the PDS (for mission-specific code, when appropriate), or an appropriate community-recognized depository (for instance, the homepage of the code base for which a module was developed). Archiving software […] does not require the proposer to maintain the code."

PDART defines a “PDS Equivalent Archive” as one with:

- Independence – [Sure.]
- Sustainability (>25 years) – [Maybe >10 years.]
- Open Accessibility – [Maybe impossible.]
- Searchability – [Solved.]
- Citability – [Solved.]
- Preeminence – [Solvable.]
- Standardization – [Solvable.]
- Optional: Peer Review – [Solved / solvable.]
- Optional: Documentation – [Solved / solvable.]

No such archive exists for software!

Sustainability: >25 years. Ideally 50.

- Nobody can guarantee this with software.
- I think that >10 years is feasible w/ virtual machine nesting and existing market forces.
  - It's VMs all the way down!
  - See: ISIS2
- 50+ years might be feasible with a Universal Virtual Machine (which does not exist)

Open Accessibility

- Software licensing is a huge mess.
- Software can be ITAR or EAR
  - Whereas observational data gets a pass.
- Analogous to copyright problems with book digitization efforts, but >100X harder, and Google has not yet offered to pay for it.

At least in the foreseeable future, any software archive will have to have managed access.

Searchability:

- Plain old search is fine, but how do you know if the software you're looking at is what you need? How do you obtain and use it?
- Answer: Olive Executable Archive
  - It's kind of like YouTube for virtual machines.
  - Developed by / with archivists at Carnegie Mellon.
  - olivearchive.org

Citability:

- Software citation is now common.
- Several easy mechanisms for obtaining DOIs or equivalents.
  - I recommend the ASCL: ascl.net
- Most citation tracking services now include software (when trackable).

Cite software!!!

Peer Review & Documentation

- It's not clear what "peer review" of research software even *means*.
  - What qualities do you review for?
  - Who count as "peers?"
- There are more opinions on documentation than there are programmers. Every programmer agrees that nobody (else) does enough of it.
  - Adequacy of documentation should be peer reviewed.
- Trailblazing effort of Journal of Open Source Software (JOSS): joss.theoj.org/

And so and also…

- Independence
- Preeminence
- Standardization

It requires only the will to do so.

And money.

A pedantic aside…

Re: “archive in Github”

Github is not an archive!

It is a project management tool.

But I’m rethinking my argument.

Suggested language for DMPs:

“All source code, scripts, or attendant documentation created under this award will be retained in the PI's personal or institutional archive at the conclusion of work. When legal and feasible to do so, source code for software created under this award will be published to Github, per the requirements outlined in Section 3.5.1 of C.1 "Planetary Science Research Program Overview" of the ROSES 2016 solicitation. Where such software is critical to the results of research performed under this award, the source code will be accompanied by documentation describing the hardware / software operating environment in which the software was developed and used, including an index of external software dependencies and libraries, as well as descriptions of tests or scripts that, at minimum, reproduce the results upon which any published research relies. When legal and feasible to do so, virtual machine images that instantiate running examples of such software will be published to an appropriate public data repository (e.g. Zenodo). When licensing restricts the release of a virtual machine image, such images will be created and retained in the PI's personal or institutional archive.”

“Don’t let perfect be the enemy of good.”

- Momentum continues towards a robust software archiving solution.
- We can do *pretty well* with existing tools.
  - And should do what we can with what we have.
- If just people here committed to (or insisted on) long term preservation of their own work products, that would be A BIG DEAL.
- An “archive lite” for software needs to exist.
  - (for Planetary Science, for NASA, for the world)

Email: chase.million@gmail.com