Ensuring continuing trust in our numerical ecosystem

Outline of BoF

1. 30 minutes to outline issues people have already encountered
    - I'll present two short examples I've encountered, then I'll open the floor for everyone to give their own experiences
    - If we don't get to you, please put your examples in chat
2. 20 minutes to discuss possible solutions
3. 5 minutes to summarise to wrap up

Scope

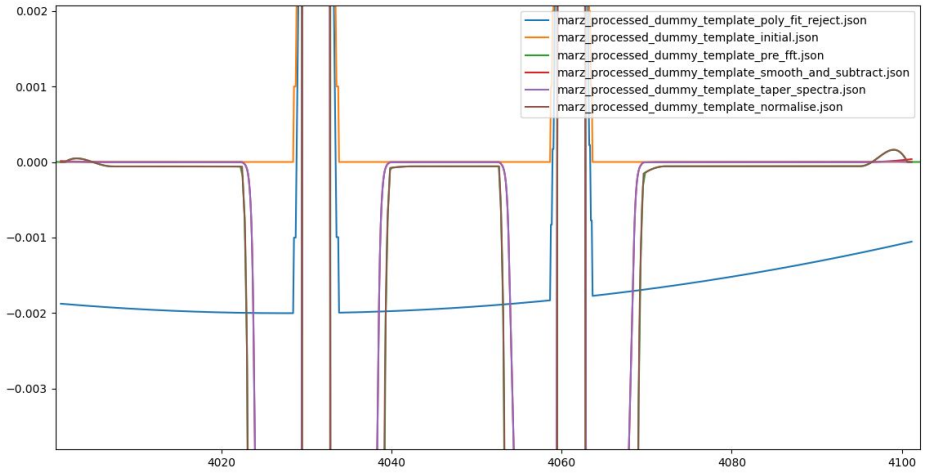
- Things that are “on topic”:
  - Where we find algorithms (e.g. papers, textbooks, wikipedia!)
  - Where we find implementations of these algorithms
  - How we check these implementations are correct
  - How we ensure the community uses these correct implementations, not incorrect ones
- Things that are “off topic” (to try to keep to time)
  - Licensing
  - Funding (that should have its own BoF)

Example 1: MARZ

- Rewrite of AutoZ from IDL to JavaScript to run in a browser/under nodejs
- Uses FFTs with templates to determine redshift of Galaxies
- Uses two libraries which are unmaintained and give invalid results:
  - DSP.js — FFT
  - Regression.JS — Linear Regressions

Example 1: MARZ

- Regression.JS
  - Fails to fits flat line with two peaks (introduces curved artefact)
- DSP.js
  - Sign issues — exactly which FFT algorithm implemented is unclear
- Testing situation in both libraries is bad
  - CI broken
  - Almost no coverage
- Can we trust JavaScript libraries? Probably not…



Example 2: Solving a cubic

- Solving a cubic should be easy:
  - Wikipedia, Numerical recipes gives formulae
- Standard formulas are fairly inaccurate:
  - Property-based testing can show this quite easily
- Kahan (of IEEE-758 fame) gives a much more accurate algorithm:
  - https://people.eecs.berkeley.edu/~wkahan/Math128/Cubic.pdf
- I implemented this in Python
  - Property-based testing shows accuracy of algorithm down to numerical precision
- Can we trust Wikipedia? Probably not.
- Can we trust Numerical Recipes? Maybe, we should test it though.

What issues have you seen?

Some ideas I've had

- Citation of specific algorithm used (see scipy docs for a good example)
  - If derived from a different codebase, cite that also (as well as following licensing)
- Proper regression tests (i.e. not just the simple cases), and using property-based testing to ensure correct behaviour (and these should be running in CI)
  - There's a BoF later this week on the topic of testing
- Be clear about any limitations of the code, and what the target use-cases are
  - Approximations or speedups which may affect precision could be acceptable in parts designed for visualisation/sonification/etc. but not for data reduction or analysis.

Ideas/Possible Solutions?

Wrap-Up