A practical point

All this is much simpler if you write code with the expectation that *someone else will read it*

(We collaborate on papers; why not so much on code?)

Coding with readers in mind makes some things easier:

- Easier to write comments you'll understand later
- Easier to write testable code, and to do unit tests for it
- Easier to do code reviews
- Easier to contemplate publishing it!

Unit testing questions

"How do I write tests when I don't know what results to expect?"

- Break it down into functions whose behaviour you can predict
- Test individual components, not the whole thing
- Testable code is also more readable code (and so more reviewable code, and...)

Unit testing is about trying to ensure that *the code implements the method*—not that *the method is the right one*

Unit testing questions

"What sort of test data and test cases should I write?"

- The simplest possible ones!
- Include a "null" input test (e.g. silent signal)

"But I have big data sets and complex results!"

- Don't use real-world data: that's a different kind of test
- Look for the smallest possible input to test a given behaviour

Version control systems

Git, Mercurial: Distributed

- every working copy has complete repository in it
- commits are local, then push/pull between computers
- e.g. work locally, then push to a remote hosting site

Subversion: Centralised

- one server, one repository database
- every commit goes straight to the shared repository

Version control hosting sites

General-purpose :

- GitHub: very popular for sharing Git repositories
- BitBucket: Git and Mercurial, good private repo support
- SourceForge: the old-school option for open source projects

Thematic:

- code.soundsoftware.ac.uk: for the UK audio and music research community

Very specific:

- Does your department provide hosting?

Day-to-day version control

- When should I start using version control for my project?
- Which files should I track in the repository?
- How often should I commit?
- How often should I push changes to a shared repo?

Publishing code

Make sure it has a licence!

We suggest one of the common open-source licences:

- BSD/MIT-style for most research code
- Consider GPL for complete applications or code with possible commercial value

Ensure licence is at least described in a README file

See our site for more about licences...

Give it a stable home!

- e.g. a recognised code-hosting facility (like ours ?!)
- Tell users what they should cite if they use it