SER
Software Engineering for Research

Intros

- Tom
  - Ran the data team at a SaaS company
  - Now working on a startup - making it easy for anyone to search and work with data
  - Have written a lot of software
- Claudia
  - HDS CDT Cohort 1!
  - Now working on computer vision for histology analysis (HAP.PY project)
  - Try to apply engineering practices to research code

What is software engineering?

Wikipedia:

Software engineering is the systematic application of engineering approaches to the development of software.

John Carmack:

Engineering is the disciplined production of technology

What is the goal?

- To build software that accomplishes all of its objectives.
- The interesting question is, what are the objectives?
- Is it enough just to get the answer right?

Objectives for a software project

- Flexibility
- Accuracy
- Performance
- Reliable
- Collaboration
- Reproducibility

What does it involve?

- Techniques - specific tools to accomplish our goals. E.g.
  - Testing
  - Version control
  - Abstraction
- Practices - the human side
  - Convention - e.g. naming and project structure
  - Empathy for other users of the code
  - Review - 2 sets of eyes catch more errors than 1
  - Discipline - practices and techniques are useless if you don’t do them

Why is any of this relevant to research?

- We need confidence in our results. Do we know they are right?
- There is a crisis of reproducibility in research, this is in large part due to poor software engineering practice.
- Feedback loops are slow when developing models - they don't have to be.
- It is hard to collaborate on research code. Whether it's reusing bits of code, extending someone's work or reproducing it.

Learning objectives for the week

- Develop awareness of the tools, techniques and practices that are available to solve problems in your work
- Learn how to use version control to collaborate on software
- Learn how to structure code for reuse and collaboration
- Learn how abstractions can make code reusable and testable
- Learn how testing can improve code quality and make code faster to iterate on.

Even though the practicals will feature ML code, the course is not directly about ML.

After the week

Learning never stops. If you keep writing code you will constantly be learning new stuff, forever.

As such, another goal of this course is for you to come away with a ‘lay of the land’ and a load of resources that you can refer to.

Everything we’ll talk about is a very deep topic in its own right, we can’t cover everything. Knowing what’s available and where to look for more info is what you should strive for.

Some resources will be fairly lightweight and specific, but others are quite heavy - e.g. official documentation. Learning to use resources like that is absolutely central to becoming an effective software engineer - don’t be put off!