An open model for training in reproducible data science

Keywords: reproducibility; open-source; collaboration.

Training type: workshop; summer school; blended training.

Data science themes: reproducible research; data stewardship, management & sharing; data modelling skills; data exploration / interpretation; improving software, computing, infrastructure, architecture & data engineering knowledge; statistics skills; machine learning.

Health & bioscience themes: neuroscience; computational social science; psychology; imaging; medicine.

Scientific Value and Quality of Training

We ask for support to use open-source methods to form a sustainable community, materials and process for collaboration on teaching reproducible data science.

Background: Data science is an approach to data analysis with a foundation in code and algorithms.[1] It is a recognition of the central role that code has played in data analysis since computing became widely available [1]. The insight of data science is that code is comparable to bench skills and lab notebooks in wet-lab research; every researcher doing quantitative data analysis must learn code for data analysis, and they must learn organized process for using code and data. The consequences of not doing so are the same as those for the poorly trained wet-lab researcher; confusion, inefficiency and error. This is the central message of computational reproducibility; if we do not take code seriously, our work will be full of error, and poor organization means that analysis is hard to share and difficult to extend. This leads to great waste in wrong results and repeated work.

Another insight of data science testing is that code is the most fruitful foundation for teaching the central ideas of statistical inference and prediction. The mathematical formulation that generations of students have suffered was necessary because we needed maths to simplify calculations for lack of computing power; “mathematics as computational engine” [2]. With an abundance of computing power, we can use much simpler resampling algorithms that are easy to express in code, and give deeper understanding: “computation as computational engine.”

Data science teaching therefore contains three fundamental ideas:

1. **code-as-analysis**; code is the correct and essential tool with which to express algorithms of practical data analysis. This kind of teaching approaches code from the point of view of data; students start early with fundamental objects for data analysis such as arrays and data tables, and proceed to data selection and visualization.
2. **code-to-learn**; the traditional approach to code is to teach “programming” in a separate advanced course (“learning-to-code”). Data science uses code as the basis to explain and implement the algorithms of inference and prediction, including machine learning (“code-to-learn”).
3. **code-for-reproducibility**; taking code seriously involves careful process in the organization and sharing of code and data — organized coding practice (OCP). OCP leads to reproducibility as a natural product of organized working process — **reproducibility by design**. OCP has great impact in increasing efficiency, reducing error and supporting collaboration, and for these reasons, is all but universal in industry and open-source.

[1] https://matthew-brett.github.io/cfd2020/intro/what-is-data-science.html

OCP includes version control; standardized data organization and verification; code review; documentation; contribution and issue tracking; and continuous integration testing. Although typically considered advanced, we have shown that we can teach these methods successfully to undergraduates in a single-semester course [3].

Need and challenge: Data science teaching promises great increase in research productivity and reproducibility — but the field is very new, and there are very few teachers with enough experience and skills to teach data science, or to build new courses. It seems obvious that we will get further, more quickly, if this small community learns how to share work, but this is more difficult than it may seem. It is common for teachers to make their materials available for free online, but uncommon for other teachers to make substantial use of these materials. It is not yet completely clear why this is, but one factor may be that university teaching is so personal to the teacher, and particular to the class. Even more so than for code, we nearly always want to modify materials, to improve them or take a different path through the subject. We do not yet have an agreed framework for submitting edits to materials, or putting materials into a context in which they can be easily be re-used outside the course for which they were written.

Approach: Open-source *software* methods have been extraordinarily effective in allowing small groups of committed programmers to build software that continues to displace closed, proprietary software. This is so despite the apparently overwhelming advantages of proprietary software in terms of paid programmer hours and market share. If we can apply this model successfully to data science teaching, we expect very great improvements in quality and range, compared to traditional models of closed or effectively closed content in competing proprietary courses.

We need to address the *challenge of sharing materials* by using these elements of open-source software development: A **committed community** of user-contributors; **Clear process** of contribution; **Common ownership** of materials; Complete **vendor independence**; **Rapid iteration** of content and process.

A user-contributor is a teacher who is editing and adding to the teaching material, and using these materials to teach. This is a fundamental rule in open source; code improves by its use in solving real problems, in the same way that teaching materials improve by their use in teaching.

A clear process of contribution reduces the time a contributor has to spend thinking about *process*, so they can spend time on *content*. We reflect and iterate on this process so the common process is well-adapted to our problem. Defining the process is part of our shared work.

Common ownership is essential, and this depends on licensing. A contributor must be confident that they have the rights they need to use, share and distribute their work. This means that we cannot use common license clauses for teaching materials, such as Non-Commercial or No-Derivative clauses. Many contributors want to allow any group to benefit from their work, including companies. As we discuss above, it is rare we can use materials without modification. Modification is the engine of experiment, and therefore, improvement. For example, Berkeley allows anyone to use their textbook, but does not allow anyone to modify and redistribute their textbook materials without the explicit permission of the authors. Even if Berkeley grants this permission, the condition makes it possible for Berkeley to withdraw a contributor's ability to use their own work, if it modifies any of Berkeley's materials — a risk that most contributors will not accept.

Common ownership implies vendor independence. It must be demonstrably impossible for any company or group of companies to withhold a contributor's right to effectively publish their work. Therefore, the machinery to deploy these materials must work on any system, including the contributor's own desktop computer.

Rapid iteration is an essential engine for continuous improvement. It must be simple to build and deploy these teaching materials in a form from which we can teach. This should be possible using automation with scripts; in due course deployment should be part of the testing process of every contribution (continuous integration).

Tools: The current success of data science has much to do with the great increase in the quality of its tools. The two primary languages of data science are Python and R, along with their associated data science libraries. Our team has experience in teaching both. We will adapt our materials and process to support both languages in the same framework. Notebooks have many advantages for teaching both languages; we will use browser-based interactive notebooks for most of our introductory teaching (Jupyter / JupyterHub for Python, R / RStudio server in JupyterHub for R). Experience shows teaching in notebooks works much better if students have regular in-line tests to check they are on the right track (Otter-grader for Python; Ottr or Testwhat for R).

For each course type (see below), we build an online, interactive *textbook* (Jupyter-Book for Python; Bookdown / notebook customizations for R). We will also deploy the course into Open edX, an open-source virtual learning environment (VLE), that supports blended learning with a combination of video and live instruction. Open edX has extensive mechanisms for customization.

Objectives: We will seed and grow the community of contribution by applying open-source methods and process. Community and process develop by shared work, and reflection on that work. We will use rapid iteration to support that process.

We plan three *course types*, listed below, and three course *versions*. For quick iteration, and maximum flexibility, we will deliver the core course types in each version. The versions are: two-day introductory workshop (2DWS); two-week summer school (2WSS); 10 week full semester course (10WFSC). The last will be a full video and live online blended course, within a VLE We will start with the 2DWS version of the core courses, then build the 2WSS, and finally the 10WFSC. See the Gantt chart for timelines. We omit the 2DWS versions for the connector courses (see below).

Our course *types* are:

1. **Foundations of data science** (FoDS) teaches code-as-analysis and code-to-learn. The course has no requirements for prior experience of coding or maths beyond GCSE. We introduce basic code for data analysis and sampling, such as variables, for-loops, arrays, data tables, and plots. There are frequent guided exercises in the form of computational notebooks running in the browser and the VLE. We teach statistical inference with simulation, resampling, and permutation. If time allows, we teach the bootstrap for confidence intervals, line fitting for regression and correlation using numerical optimization, extension of line fitting to multiple regression, an introduction to Bayesian analysis; models for prediction and basic machine learning. Seed materials start with our existing data science courses,[2] inspired in turn by Berkeley's foundation course.[3] We use a wide range of data for the analysis problems, to show the generality

[2] https://matthew-brett.github.io/cfd2020
[3] https://www.inferentialthinking.com

of the methods, and engage learners from different fields of health and life-sciences.

2. **Reproducible data science** (RDS) teaching code-for-reproducibility. Students for this course should have some prior experience of programming — for example, the FoDS course. The RDS course shows the standard methods behind OCP, including: version control with Git; code and data organization; data verification; code review with contribution and issue tracking using platforms like Github / Gitlab, and continuous integration testing. We teach tools to increase analysis and coding efficiency, such as the command line, and text editors, as well as automated tools for testing and analysis such as scripting and dependency tracking. We show the power and need for these tools with a final project involving an open-ended collaborative data analysis. Pedagogy and materials will start from our initial Berkeley course [3].
3. **Connector courses**, where the FoDS course is a prerequisite. These courses use the FoDS pedagogy to expand content into specific fields or applications. We plan three of these courses. The first is an intermediate course on probability, starting with simulation, and developing more mathematical understanding. The second is an extended treatment of statistical analysis, with a basis in resampling. It will cover multiple hypothesis testing, multi-factor ANOVA designs, and extensions of ANOVA models. The third will be an application of foundational methods to computational social science.

See the Gantt chart for timing of each course version and type.

We will target the 2DWS versions of the courses at faculty leading research groups, teachers of researchers and advanced post-docs. The aim of these courses is to show potential leaders and teachers how these teaching approaches work, and how they can be effective in teaching apparently difficult ideas like code-to-learn, and code-for-reproducibility. These workshops will finish with a brief introduction to our framework for using and contributing to the open material, and an brief introduction to deploying modified versions of the textbook and course machinery, via the Open edX VLE.

The 2WSS versions of the courses are to allow to us iterate more quickly, by starting with shorter versions of the courses, but they also allow more flexibility in the course form, so making it more likely there will be some version of the course that others can use and deploy.

As we iterate, we will improve the automation and range of the course deployment. We will likely start with manual deployment of the course textbook, interactive notebook hosting and, later, the VLE, using existing procedures. Over the grant period we will streamline and automate deployment. We intend to finish by making it simple for new teachers to use our deployment methods with minimal support.

As part of our work building teaching materials for these courses, we will curate and extend two additional resources that we have found to be enormously useful in our own teaching:

- A library of curated, processed and clearly licensed **datasets** for teaching. Each dataset will give detailed descriptions and / or copies of original sources, along with a data license, any code to clean or refine the data, and cleaned / refined versions suitable for teaching. We have started this work at https://github.com/odsti/datasets. Much of our work in data science teaching has been to find and refine these datasets.
- A private library of exercises with solutions, available to teachers on request. We have a seed repository for this available privately on Github.

Strategy

See *Team and Management* below for our process for learning from course iteration.

The audience for our work is researchers, both senior and junior, but, in order to reach a wide range of researchers, and build a community of practice, we also need to recruit teachers. Recruiting teachers allows us to take full advantage of our open-source model. Teachers will test our material, documentation, and deployment tools; some will adapt, improve and extend our materials.

To engage the broadest range of teachers, we will run local, but initially virtual, versions of each version of our 2DWS courses that rotate between the four home centres of our investigators: Birmingham, King's College, Cape Town, Essex. The 2DWS courses include an introduction to using our open materials, process and deployment methods. The 2DWSs will be open to any researcher, but we will advertise locally first, to build personal connections within the home university and aim for critical mass committed to data science teaching. In the second year of the grant we will run local versions of these workshops in other interested universities, identified with a combination of personal contacts, outside students from our previous courses, and advertising on various channels including statistics / data science teaching mailing lists, Slack groups, and Turing Institute newsletters.

A key tool for recruiting teachers is lowering the barrier to using our materials as far as we can, by iterating on course materials and deployment procedures, and making them fully open for modification and re-use.

Our end goal is to teach reproducible data science to researchers. Outreach to research group leaders is important for guiding research culture, but also in guiding lab members to the 2WSS, and 10WFSC versions of our courses. All our courses will be open to any interested researcher. As we expand our range of 2DWS, we expect our courses will be better known, and we will recruit a wider range of students, from a larger number of universities. Our online textbooks and blended learning videos will be freely available, and we will use these to attract more students.

We believe that reproducible data science is the foundation for all sound data analysis, so we want to teach it to all researchers, not just those with a prior interest in code. We will use our 2DWS and other outreach to show that our courses work for researcher with no prior experience or interest in programming. We will monitor our recruitment, to make sure we are getting a representative cross-section of researchers, and adjust recruitment and in-course support if we find we are getting a lower proportion of applicants from under-represented groups than expected for the field. It is an intended effect of our 10WFSC that any student, anywhere, can take our course online.

Our proposal includes funding for a post-doctoral fellow. Our primary intention here is to train the post-doc in building and using these teaching methods. They will also get experience of teaching other teachers. We hope that this post will evolve into a permanent post that may involve both teaching and data science research, in either of our two main applicant groups, or elsewhere. We expect the post-holder will continue to support and expand the community.

Value and Impact

Our resources fall into two broad categories: instructor time and development / consulting resources. We have chosen our instructor resources to provide substantial time from the very small number of teachers who have extensive experience of teaching data science to

life-scientists. By doing this, we hope to seed a community that can build and expand on this teaching experience, that is currently very rare. We have also selected teachers who have great experience of running distance learning courses in life-sciences, to guide our use of learning environments, and improve our interactions with students. We designed our team to give the best possible combination of experience in order to build balanced, well-designed courses in this new field.

Our development and consulting resources are to support the community of contribution. To attract our fellow teachers, we have to make it as easy as possible to deploy courses, and to use and edit our materials. We have to support common ownership, and to do this, we need to make sure we have vendor independence, and that the expertise needed to deploy courses is within the range of the largest number of teaching groups.

We will measure our success by: the number of teachers that we recruit to use our materials; the number of teachers who interact with the community to raise issues or improve documentation; the number of teachers who contribute new or improved materials; the number of students applying for our summer and semester courses; the diversity of our student body; and the performance of students in assessments and final projects.

Our end ambition is that reproducible data science should become the foundation for data analysis teaching for all researchers, whether practicing or in training. We plan to make these materials so compelling that they can form a foundation for all first and second undergraduate year training in research methods and statistics.

Team & Management

Matthew Brett (MB) will lead the project. All the co-PIs will form a board, to which MB will report every six months, in an all-hands day-long review meeting. During the project we will form working groups for each iteration of the courses we deploy. These will typically be two teachers and two reviewers, where at least one team member will have development expertise in order to refine working process for building and deploying materials, and feeding back to the development consultant team. For four weeks prior to the course to one week following we will have a weekly short review meeting, and thrice weekly 30 minute stand-up meetings to feed back on progress and obstacles. We will collect student feedback after each course, and follow up with a team meeting at which the team leader presents and discusses a ~2 page report discussing success and failure, and proposals for improving our process, and the next course.

We will record bugs, errors in documentation or desirable features using the usual Github / version control features.

We will set up a mailing list for the team members, and a Discourse forum for the project, to encourage questions and feedback from the widest possible range of teachers and students.

1. Donoho D (2015) 50 years of data science. In: Princeton NJ, Tukey Centennial Workshop
2. Cobb GW (2007) The introductory statistics course: A ptolemaic curriculum? Technology Innovations in Statistics Education 1:
3. Millman KJ, Brett M, Barnowski R, Poline J-B (2018) Teaching computational reproducibility for neuroimaging. Frontiers in Neuroscience