The Journal of Open Source Education

A graduate student-led participatory live-coding quantitative methods course in R: Experiences on initiating, developing, and teaching

Luke W. Johnston[2,3], **Madeleine Bonsma-Fisher**[1], **Joel Ostblom**[4], **Ahmed R. Hasan**[6], **James S. Santangelo**[7], **Lindsay Coome**[5], **Lina Tran**[8], **Elliott Sales de Andrade**[1], **and Sara Mahallati**[4]

1 Department of Physics, University of Toronto **2** Department of Nutritional Sciences, University of Toronto **3** Department of Public Health, Aarhus University **4** Institute of Biomaterials and Biomedical Engineering, University of Toronto **5** Department of Psychology, University of Toronto **6** Department of Cell and Systems Biology, University of Toronto **7** Department of Ecology and Evolutionary Biology, University of Toronto **8** Department of Physiology, University of Toronto

DOI: 10.21105/jose.00049

Software

- Review
- Repository
- Archive

Submitted: 15 January 2019
Published: 06 June 2019

Introduction

We present an open source learning module suitable for a semester long course and designed to leverage participatory live-coding techniques to teach both statistical and programming skills to primarily upper-year undergraduate biology students. Our learning module has three self-contained submodules spanning sixteen lessons: 1) Programming in R, basic data wrangling, and visualizations; 2) Exploratory data analysis, statistics, and modelling; and 3) Collaborative and reproducible science. Our learning module includes eight assignments distributed throughout the term to assess students' learning and understanding. The material is made available as R Markdown documents and designed to be taught using R Notebooks. Students are not expected to have any prior knowledge of the R language. Our material is licensed under CC-BY 4.0 while the code is under the MIT License. Our course is a response to the growing need for programmatic training emphasizing sound data analysis practices among researchers. We believe the included lesson topics, open accessibility, and modularity of our course makes it an ideal resource for instructors.

Statement of Need

In traditional undergraduate biology education, students learn statistical skills and biological concepts separately, without any practical application through coding. Designed primarily for upper-year undergraduate students, this learning module emphasizes gaining skills in R coding in the context of learning statistics and ecology. Notably, the material covers statistical concepts that are broadly useful in biological sciences, including mixed effects models, randomization tests, model selection, and differential equations. While we delivered the material and concepts as a four-month long course, these concepts are structured into primarily independent submodules focused around several lessons, which could easily be mixed and matched to suit any desired learning outcome. Lessons were designed to be interactive and delivered in a participatory live-coding format so students learn experientially. The teaching material includes assignments to hone and reinforce students' understanding and allow them to critically apply their skills to new problems.

Reproducible quantitative research skills are emphasized throughout, culminating in an open-ended self-directed project that requires students to apply their skills to a real ecological dataset and problem. The teaching material is hosted in a public GitHub repository which automatically generates a website that presents the text, code, and code output together on the same page. The material is openly available and licensed; anyone can easily copy and modify for their own purposes.

Learning Objectives and Content

The overarching objective of the course is to teach reproducible and collaborative quantitative research skills. The lessons are described in more detail in Table 1 and are organized into three submodules:

1. Programming in R (R Core Team, 2018), basic data wrangling, and visualization (lessons 1-5).
2. Exploratory and statistical data analysis (lessons 6-13).
3. Collaborative and reproducible science (lessons 14-15).

Table 1: Overview of submodules, lessons, and packages used in the learning module.

Submodule	Lesson	Description	Packages used
Programming in R, data wrangling, visualization	1	Introducing R, RStudio, and R Markdown	
	2	Vectors, data frames, basic operations, and functions	`tidyverse` (Wickham, 2017)
	3	Introduction to exploratory data analysis	`tidyverse`
	4	Introduction to statistics and visualization	`tidyverse`
	5	Data transformation and visualization	`tidyverse`
Exploratory and statistical data analysis	6	Cleaning and preprocessing raw data	`tidyverse`; `mice` (van Buuren & Groothuis-Oudshoorn, 2011)
	7	Descriptive and inferential statistics	`tidyverse`; `car` (Fox & Weisberg, 2011); `psych` (Revelle, 2018); `multcomp` (Hothorn, Bretz, & Westfall, 2008)

Submodule	Lesson	Description	Packages used
	8	Linear mixed-effects models	`tidyverse`; `plyr` (Wickham, 2011); `lme4` (Bates, Mächler, Bolker, & Walker, 2015); `lmerTest` (Kuznetsova, Brockhoff, & Christensen, 2017)
	9	Randomization tests and data simulation	`tidyverse`; `reshape2` (Wickham, 2007); `EcoSimR` (Gotelli, Hart, & Ellison, 2015)
	10	Multivariate statistics (e.g. PCA)	`tidyverse`; `car`; `psych`; `multcomp`
	11	Model selection and averaging	`tidyverse`; `lme4`; `lmerTest`; `MuMIn` (Bartoń, 2018)
Numerical models	12	Population modelling with differential equations	`tidyverse`; `deSolve` (Soetaert, Petzoldt, & Setzer, 2010)
	13	Time-series data and numerical models	`tidyverse`; `deSolve`
Collaborative and reproducible science	14	Scientific methods	
	15	Collaborating through Git and GitHub	
	16	Manuscript preparation in R Markdown	`knitr` (Xie, 2018); `rmarkdown` (Allaire et al., 2018)

Instructional Design

Drawing on the instructors' previous experiences teaching introductory programming workshops, we designed our lessons to have the following components:

1. *Lesson Outline*: Each lesson has a clearly defined outline of the lesson objectives, including expected time spent on each objective. This gives students a clear expectation of what they should learn and gain from the lesson. It also provides a structured template for instructors to prioritize content and gauge how much time each objective should take.
2. *Participatory Live-Coding*: Coding in real-time with the students actively coding along, forms the primary focus of each lesson. This hands-on approach to teaching is frequently used by teaching organizations such as Software Carpentry (Haaranen, 2017; Michonneau & Fournier, 2018; Rubin, 2013; Wilson, 2018). While many learning outcomes focus on developing programming proficiency, some lessons are

centred around concepts (such as "Statistical Modelling" or "Differential Equations"), during which we still use the live-coding approach. This approach not only demonstrates the concepts in a step-by-step fashion but also helps students practice writing code.
3. *Interwoven Exercises*: Coding exercises or discussion points are interspersed throughout each lesson to assess and reinforce the concepts and skills being taught. These exercises challenge the students and help build confidence in the material and in their coding skills. They also help instructors identify problem areas that should be further reinforced later in the lesson or submodule.
4. *Summative Assignments*: Lesson specific assignments are used every two lessons to test the competency of students to the lesson material and expected skills to be gained, while a comprehensive final assignment is used to test the students' ability to bring together all concepts learned throughout the learning module.

Each of our submodules and individual lessons built on skills and concepts that would ultimately allow students to complete a final open-ended analysis of real open ecological data. We deliberately chose large and messy (e.g. missing values) datasets for the students, reflecting the types of data that are being increasingly generated across various disciplines. With this goal in mind, we designed lessons to provide the building blocks to clean, manipulate, visualize, and analyze any dataset the students may come across, both for the final project and in their future research.

Teaching Experience

For the first iteration of the course, our teaching team consisted of six graduate students from diverse fields of research; we divided course topics among each instructor to develop and deliver individual lessons and assignments to the eight students. We reduced the number of instructors to four graduate students for the second iteration and the number of students increased to 26. We estimate four instructors could effectively teach the current iteration of the course to around 40 students. We consider having instructors come from multiple fields as a major strength and strongly recommend this practice for teaching quantitative research methods and skills.

To maximize the learning experience, we prioritized in-class participation, engagement, and hands-on experience. The main teaching techniques we used to achieve this goal were participatory live-coding, exercises interwoven with teaching, and project-based learning (Markham, 2011; Sawyer, 2006; Strobel & Barneveld, 2009) where students collaborated in teams on data analysis problems to mimic a real world scenario.

To ensure proper teaching assistance was available at all times, we adopted a technique used successfully in workshops developed by The Carpentries (Wilson, 2006). This technique involved having at least two instructors present for each lesson, where one instructed and another acted as a "helper". Students would signal for assistance by attaching colored sticky notes to the back of their laptop monitor. This method avoided interrupting the lesson flow when individual students needed assistance.

Story of the project

While there are many excellent open source software packages available for quantitative data analysis, the use of less capable tools (such as spreadsheet software) is still prevalent among researchers, even though these drastically reduce analytical reproducibility, power, and efficiency. This happens partly due to lack of awareness, and partly because graduate

students, many of whom will be future researchers, often are not incentivized to learn new and better tools, as they usually must use what their supervisor or colleagues use. Those who do try to learn these modern tools often do so in isolation and without much formal training available. These are major barriers to learning. To help break down these barriers, we launched the graduate student group University of Toronto Coders where we run peer-led learning sessions on using code for research through skill sharing, co-working, and community building in a friendly and supportive environment.

After running many sessions and consistently receiving overwhelmingly positive feedback on our content and teaching style, we sought to formally share our experiences through the university curriculum. We designed a course on open, reproducible data analysis, and contacted multiple departments that could be interested in hosting this course. The Department of Ecology and Evolutionary Biology at the University of Toronto agreed, and we ran a pilot of the course with the title "Theoretical Ecology and Reproducible Quantitative Methods in R" to fourth-year undergraduate students. We modelled the structure and portions of the course content after the course "Reproducible Quantitative Methods", which was created by Dr. Christie Bahlai. We extensively modified the lesson content to include expanded material on data wrangling, visualization, reproducibility, collaborative science, and additional theoretical ecology topics.

Following a successful pilot term, we modified our lesson material further again to include more generally applicable statistical concepts and far fewer theoretical ecological concepts. We also renamed the course to "Quantitative Methods in R for Biology" to reflect this change. On both occasions, the course received excellent feedback from the students and the supervising professors and has been incorporated into the long-term curriculum as a third year level course.

Contributions

LWJ, MB-F, LT, and LC conceptualized the course. JO lead course development. JO, MB-F, LWJ, LC, ES, and LT designed and taught the first iteration of the course. JSS, LC, MB-F, and ARH taught the second iteration of the course, with guest lectures from SM and LT. Lesson development for second iteration: JO and ARH (1-5), JSS (8, 9, 11), LC (6, 7, 10), MB-F (12, 13), LWJ (14), ARH and SM (15), LT (16). LWJ, MB-F, JO, SM, LT, ARH, and JSS wrote the paper. LWJ, MB-F, ES, JO, LT, JSS, and AH proofread and edited the final draft.

References

Allaire, J., Xie, Y., McPherson, J., Luraschi, J., Ushey, K., Atkins, A., Wickham, H., et al. (2018). *Rmarkdown: Dynamic documents for R*. Retrieved from https://rmarkdown.rstudio.com

Bartoń, K. (2018). *MuMIn: Multi-model inference*. Retrieved from https://CRAN.R-project.org/package=MuMIn

Bates, D., Mächler, M., Bolker, B., & Walker, S. (2015). Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, *67*(1), 1–48. doi:10.18637/jss.v067.i01

Fox, J., & Weisberg, S. (2011). *An R companion to applied regression* (2nd ed.). Thousand Oaks, CA, USA: Sage. Retrieved from https://socialsciences.mcmaster.ca/jfox/Books/Companion-2E

Gotelli, N. J., Hart, E. M., & Ellison, A. M. (2015). *EcoSimR: Null model analysis for ecological data*. doi:10.5281/zenodo.16522

Haaranen, L. (2017). Programming as a performance: Live-streaming and its implications for Computer Science education. In *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE '17 (pp. 353–358). New York, NY, USA: ACM. doi:10.1145/3059009.3059035

Hothorn, T., Bretz, F., & Westfall, P. (2008). Simultaneous inference in general parametric models. *Biometrical Journal*, *50*(3), 346–363. doi:10.1002/bimj.200810425

Kuznetsova, A., Brockhoff, P. B., & Christensen, R. H. B. (2017). lmerTest package: Tests in linear mixed effects models. *Journal of Statistical Software*, *82*(13), 1–26. doi:10.18637/jss.v082.i13

Markham, T. (2011). Project-based learning: A bridge just far enough. *Teacher Librarian; Bowie*, *39*(2), 38–42. Retrieved from https://search.proquest.com/docview/915254354/abstract/707DEDB5F1E145E5PQ/1

Michonneau, F., & Fournier, A. (Eds.). (2018, November). Data Carpentry: R for data analysis and visualization of ecological data. doi:10.5281/zenodo.569338

R Core Team. (2018). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing. Retrieved from https://www.R-project.org/

Revelle, W. (2018). *psych: Procedures for psychological, psychometric, and personality research*. Evanston, IL, USA: Northwestern University. Retrieved from https://CRAN.R-project.org/package=psych

Rubin, M. J. (2013). The effectiveness of live-coding to teach introductory programming. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, SIGCSE '13 (pp. 651–656). New York, NY, USA: ACM. doi:10.1145/2445196.2445388

Sawyer, R. K. (Ed.). (2006). *The Cambridge handbook of the learning sciences*. Cambridge, NY, USA: Cambridge University Press. doi:10.1192/bjp.bp.106.029678

Soetaert, K., Petzoldt, T., & Setzer, R. W. (2010). Solving differential equations in R: Package deSolve. *Journal of Statistical Software*, *33*(9), 1–25. doi:10.18637/jss.v033.i09

Strobel, J., & Barneveld, A. van. (2009). When is PBL more effective? A meta-synthesis of meta-analyses comparing PBL to conventional classrooms. *Interdisciplinary Journal of Problem-Based Learning*, *3*(1). doi:10.7771/1541-5015.1046

van Buuren, S., & Groothuis-Oudshoorn, K. (2011). mice: Multivariate imputation by chained equations in R. *Journal of Statistical Software*, *45*(3), 1–67. doi:10.18637/jss.v045.i03

Wickham, H. (2007). Reshaping data with the reshape package. *Journal of Statistical Software*, *21*(12), 1–20. doi:10.18637/jss.v021.i12

Wickham, H. (2011). The split-apply-combine strategy for data analysis. *Journal of Statistical Software*, *40*(1), 1–29. doi:10.18637/jss.v040.i01

Wickham, H. (2017). *Tidyverse: Easily install and load the 'tidyverse'*. Retrieved from https://CRAN.R-project.org/package=tidyverse

Wilson, G. (2006). Software Carpentry: Getting scientists to write better code by making them more productive. *Computing in Science & Engineering*. doi:10.1109/MCSE.2006.122

Wilson, G. (2018). *Teaching tech together: How to design and deliver lessons that work and build a teaching community around them*. Leipzig: Amazon Distribution GmbH. Retrieved from http://teachtogether.tech/

Xie, Y. (2018). *knitr: A general-purpose package for dynamic report generation in R*. Retrieved from https://yihui.name/knitr/