BCB 503 | WORKSHOP: VERSION CONTROL

BCB 503-01 | CRN 44933

WORKSHOP: VERSION CONTROL AND TEST-DRIVEN DESIGN FOR BIOLOGICAL SOFTWARE DEVELOPMENT

Workshop description

In this workshop, students will learn how to use version control and test-driven design to create and maintain biological software. Students will learn how to break down problems associated with processing and analyzing large, cumbersome datasets into a set of manageable requirements; how to implement software process tools such as version control and test-driven design in `R` to confidently write robust code that meets these requirements, and how to efficiently and effectively teach these skills to others.

Instructors: Breanna Sipley, Luke Harmon, JT
Helpers: Perhaps Kristen and Clint

Workshop design

This workshop aims to develop scientific inquiry, software process, and pedagogy skills evenly in learners. In particular, learners will learn how to develop software using test-driven development in `R`, collaborate using version control with Git towards a shared scientific goal (fixing bugs in a popular biological `R` package), and communicate those findings with classmates using inclusive pedagogical principles. See *Figure 1* for a graphical illustration of the conceptual framework of this workshop.

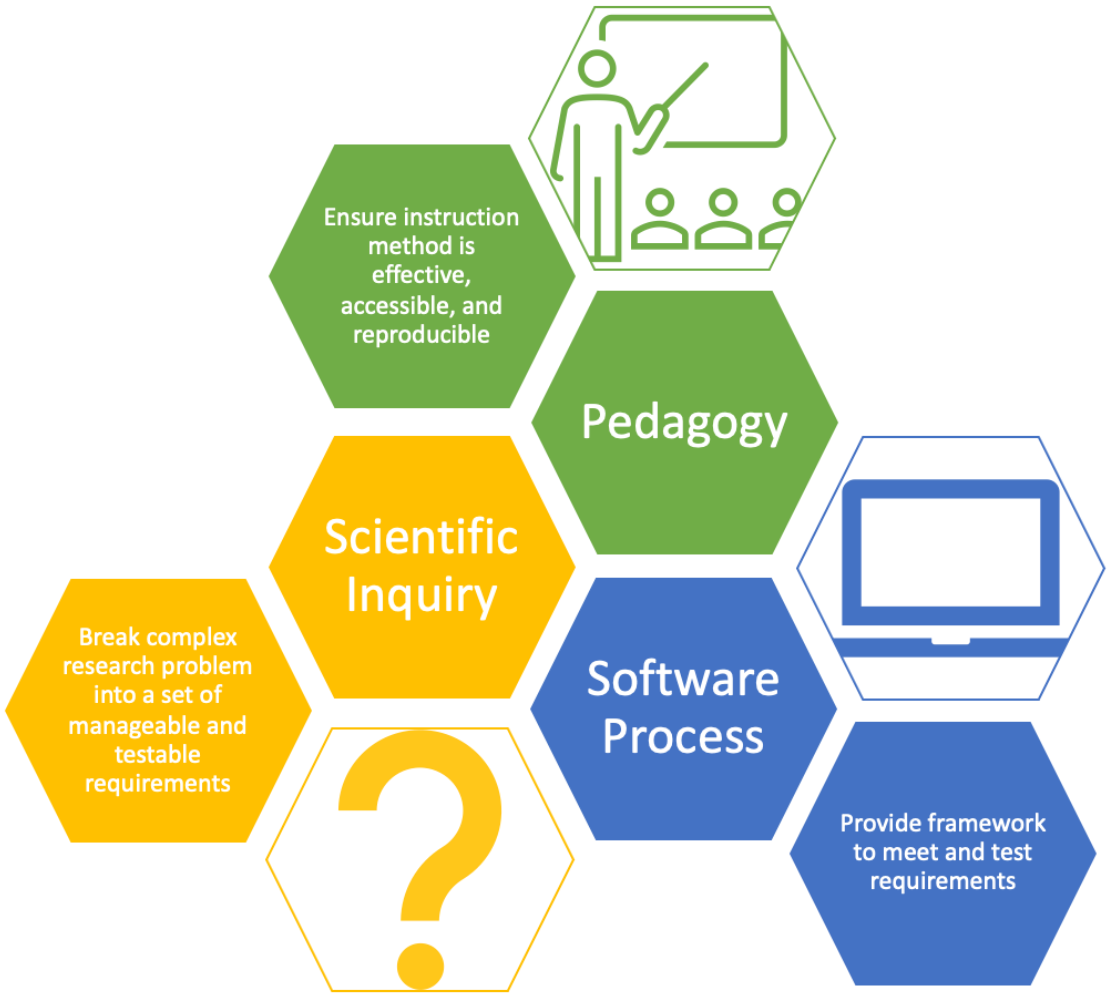

Figure 1. A graphical representation of the conceptual framework of this workshop.

In addition, this workshop has the potential for much broader impacts. This workshop was designed as an ***Idaho EPSCoR Vertically Integrated Project***, which aims to "provide the scaffolding to support transdisciplinary science and grow the next generation of conservation science leaders and workers" (for more information, see https://www.idahogem3.org/vip). This workshop is also part of the Fall 2022 Carpentries Workshops series sponsored by the ***Institute for Modeling Collaboration and Innovation*** (IMCI). ***The Carpentries*** is a registered non-profit that teaches "foundational coding and data science skills to researchers worldwide", training and certifying volunteer instructors and providing curriculum in a variety of topics designed to be presented as workshops (see https://carpentries.org/). This workshop is founded in the Carpentries Pedagogical Model, and most lessons, including "Version Control with Git", have been created by The Carpentries. This workshop is not an official Carpentries workshop, however. Instead, this workshop aims to test a new Carpentries lesson proposal entitled "Introduction to Test Driven Development" for submission to the Carpentries Incubator (https://carpentries-incubator.org/), where it may benefit from collaborative development, and

hopefully eventually be accepted into The Carpentries Lab for improvement by participating in open global peer review. Finally, this workshop serves as a critical component of a dissertation in Bioinformatics and Computational Biology for a multiply disadvantaged PhD candidate.

This workshop has intellectual merit. Participants in this workshop will help improve the popular biological software package `GEIGER` and will contribute, if they'd like, to a pedagogical research study (see Disclaimer). This workshop benefited from consultation with the Center for Excellence in Teaching and Learning.

Is this workshop for you?

In general, this workshop will benefit those with interest in scientific inquiry (especially early career biologists), inclusive pedagogy, and/or software process. To that end, this course will especially appeal to graduate students and/or postdocs interested in learning tools to help debug code, skills necessary to develop a successful research programme and/or design workshops founded in inclusivity, and/or are interested in transitioning to industry.

To help decide whether this workshop may be useful to you, please reflect on whether any of the below questions apply to you:

- Do you feel overwhelmed with cumbersome biological datasets?
- Do you feel intimidated by the evolving landscape of computational skills necessary to analyze biological data?
- Do you write code AND have never heard of "software process"?
- Do you write code AND use version control with Git either not at all or only intermittently?
- Do you know that version control would be useful to you but have difficulty imagining how to implement it in your daily workflow?
- Would you like to practice collaborating via a remote server such as Github?
- Do you feel overwhelmed towards the end of projects and/or find that you waste a lot of time debugging code that you thought worked?
- Are you interested in learning how to debug code and/or practicing debugging code more efficiently?
- Are you interested in connecting with a practical computing community?
- Are you interested in learning tools to help improve inclusivity in classrooms?
- Are you a life-long learner curious about the concept of "Test-Driven Development"?
- Are you an evolutionary biologist?
- Do you use the software package `GEIGER`?
- Are you interested in contributing to broader impacts? (Please note: this workshop serves as the foundation of part of a dissertation chapter for a first-gen non-binary autistic person from a challenging socioeconomic background)
- Are you interested in supporting Idaho EPSCoR, especially VIP project development?

Prerequisites

No prerequisites are required for this workshop. Experience in `R` will be beneficial but not necessary. Learners will acquire all the tools they need to be successful in this workshop – and experienced learners can rest assured that minimal time will be spent reviewing basics. Only the skills necessary to complete the workshop project will be taught; there will be no superfluous lessons.

Workshop website

Link: https://sipley.github.io/2022-10-17-git-TDD/

Course details

Associated Term:Fall 2022
CRN:44933
Campus: Moscow
Room: LSS341
Schedule Type: Lecture
Instructional Method: Classroom Meeting
Section Number: 02
Subject: Bioinformatics/Computatnl Biol
Course Number: 503
Title: WS: Version Control
Credit Hours: 2
Grade Mode: Pass/Fail Course P/F/I

In the spirit of inclusivity, the instructor can accommodate hybrid participation during regular scheduled class meetings. Please reach out to Breanna directly at sipl0809@vandals.uidaho.edu if in-person attendance would limit your participation in this course. Zoom link: https://uidaho.zoom.us/j/89049753624

Workshop communication

We will communicate via the platform Slack. Link here: https://join.slack.com/t/uicarpentries/shared_invite/zt-1h8o28vvb-hQi6nZf~dB7ins82errkCg. Please search for and join the channel called “#2022-version-control” and please note this invitation link will expire in 30 days from 2022-10-06.

Disclaimer

This course is part of a research study determined by IRB to be Exempt under Category 1 at 45 CFR 46.104(d)(1) (Protocol: 22-101, Reference: 019087). This study consists of pre- and post-surveys designed to assess attitudes and competence in both test-driven design

and version control approaches for software design. Participation in the surveys is voluntary, anonymous, and unlinked to grades and instructor perception. This research is part of a student dissertation, and we plan to publish a paper on our approach, and will include anonymized survey results.

Learning goals

- Explain how software process is to software development as the scientific method is to researchers and therefore of value to researchers who code
- Build confidence in accessing and utilizing software process tools to debug `R` packages
- Improve inclusivity in STEM by learning how to create and maintain explicitly inclusive learning environments

Learning outcomes

Practicing pedagogy

- Understand how to creating and maintaining an explicitly inclusive learning environment
- Understand the limitations of knowledge in the absence of a functional mental model
- Understand the benefits of formative assessments to diagnose broken mental models
- Describe how expertise can help and hinder effective teaching
- Demonstrate strategies for avoiding dismissive language
- Understand the quantitative limit of human memory
- Evaluate cognitive load associated with learning a task
- Explain strategies to avoid demotivating learners
- Build confidence soliciting feedback
- Give thoughtful and useful feedback

Practicing scientific inquiry

- Describe the Scientific Method
- Clearly state an observation
- Break down research problems into manageable sets of requirements
- Propose solutions (develop hypotheses)
- Make predictions based on hypotheses
- Assess the extent to which requirements are met
- Write new requirements (develop new hypotheses) when necessary
- Assess what inferences can be made
- Identify limitations
- Build confidence understanding and discussing scientific papers

Practicing software process

- Explain the benefits of software process, especially Agile Software Development

- Provide overview of Test-Driven Development (TDD)
- Write unit tests for software requirements using pseudocode
- Understand the benefits of an automated version control system
- Describe the modify-add-commit cycle
- Distinguish between descriptive and non-descriptive commit messages
- Explain why ignoring files can be useful and configure Git to ignore specific files
- Track progress, revert changes, and collaborate via remote server
- Write tests using `testthat` in RStudio
- Write code to pass tests in `R`

Class Project

The class project is designed to demonstrate the above skills, namely managing integration of new feature requests, collaborating via remote server, practicing inclusive pedagogy, writing test functions, resolving issues in an `R` package via pull request on Github.

The biological software package `GEIGER` (Harmon et al, 2008) will be used as a test case for the course. `GEIGER` is a popular `R` package used for investigating evolutionary radiations. There are some known issues in this software package, which we will attempt to resolve via this workshop:

- https://github.com/mwpennell/geiger-v2/issues/6
- https://github.com/mwpennell/geiger-v2/issues/4

In this workshop, we will begin debugging together – the instructor sharing screen and soliciting suggestions from learners. Learners will be introduced to lessons as needed, as required knowledge arises – and lessons will be revisited when helpful. For instance, as the software was written in `R`, learners will be introduced to only what is needed to be successful in meeting the workshop objectives (e.g., basic syntax, project management in R, writing functions). After a known issue in the software package is successfully resolved together – or when learners feel sufficiently confident to try resolving an issue without instructor lead – learners will be paired randomly, each pair tasked with collaboratively improving the resolution to the first issue and resolving an additional issue using test-driven development and version control. Learners will share their solutions, where applicable, as well as hiccups, using live coding, with the class. As a group, with help from instructors, learners will solicit and give feedback and refactor code to come to one team solution for each issue, which will be submitted as final pull requests to `GEIGER` on Github.

Workshop agenda

Below is a possible workshop agenda. Please note the instructor will be flexible and will alter this agenda to facilitate learners. In the event that material is changed, the workshop website will be updated to reflect changes.

Date	Specific outcomes	Linked content
10/17/2022	Demonstrate creating and maintaining an explicitly inclusive environment; Solicit feedback; Give thoughtful and useful feedback	[link to pre-survey] https://carpentries.github.io/instructor-training/01-welcome/index.html https://datacarpentry.org/R-ecology-lesson/00-before-we-start.html
	Describe the Scientific Method; State an observation	https://github.com/mwpennell/geiger-v2/issues
	Explain the benefits of software process, especially Agile Software Development; Provide overview of Test-Driven Development (TDD)	https://v4.software-carpentry.org/softeng/agile.html https://sipley.github.io/tdd-course/index.html
10/24/2022	Understand the limitations of knowledge in the absence of a functional mental model; Understand the benefits of formative assessments to diagnose broken mental models; Describe how expertise can help and hinder effective teaching	https://carpentries.github.io/instructor-training/02-practice-learning/index.html https://carpentries.github.io/instructor-training/04-expertise/index.html
	Break down research problem into manageable set of requirements; Use paper discussion template as aid in reading and discussing scientific papers	https://github.com/mwpennell/geiger-v2/issues https://academic.oup.com/bioinformatics/article/24/1/129/205524 [link to discussion template]
	Write unit tests for software requirements using pseudocode; Understand the benefits of an automated version control system Describe the modify-add-commit cycle; Distinguish between descriptive and non-descriptive commit messages	https://swcarpentry.github.io/git-novice/01-basics/index.html https://swcarpentry.github.io/git-novice/02-setup/index.html https://swcarpentry.github.io/git-novice/03-create/index.html https://swcarpentry.github.io/git-novice/04-changes/index.html https://swcarpentry.github.io/git-novice/05-history/index.html
10/31/2022	Understand the benefits of formative assessments to diagnose broken mental models; Demonstrate strategies for avoiding dismissive language; Understand the quantitative limit of human memory;	https://carpentries.github.io/instructor-training/05-memory/index.html https://carpentries.github.io/instructor-training/09-eia/index.html https://carpentries.github.io/instructor-training/08-motivation/index.html

	Evaluate cognitive load associated with learning a task	
	Break down research problem into manageable set of requirements Propose solutions (develop hypotheses); Make predictions	https://github.com/mwpennell/geiger-v2/issues https://academic.oup.com/bioinformatics/article/24/1/129/205524 https://lukejharmon.github.io/pcm/chapter8_fitdiscrete/
	Explain why ignoring files can be useful and configure Git to ignore specific files; Track progress, revert changes, and collaborate via remote server	https://swcarpentry.github.io/git-novice/06-ignore/index.html https://swcarpentry.github.io/git-novice/07-github/index.html https://swcarpentry.github.io/git-novice/08-collab/index.html https://swcarpentry.github.io/git-novice/09-conflict/index.html https://swcarpentry.github.io/git-novice/10-open/index.html https://swcarpentry.github.io/git-novice/11-licensing/index.html https://swcarpentry.github.io/git-novice/12-citation/index.html https://swcarpentry.github.io/git-novice/13-hosting/index.html https://swcarpentry.github.io/git-novice/14-supplemental-rstudio/index.html
11/07/2022	Explain strategies to avoid demotivating learners; Solicit feedback; Give thoughtful and useful feedback	https://carpentries.github.io/instructor-training/06-feedback/index.html https://carpentries.github.io/instructor-training/11-practice-teaching/index.html
	Break down research problem into manageable set of requirements Propose solutions (develop hypotheses); Make predictions; Assess the extent to which requirements are met; Write new requirements (develop new hypotheses) when necessary	
	Track progress, revert changes, and collaborate via remote server Write tests using `testthat` in RStudio Write code to pass tests in `R`	https://swcarpentry.github.io/r-novice-gapminder/02-project-intro/index.html

11/14/2022	Solicit feedback; Give thoughtful and useful feedback	
	Make predictions; Assess the extent to which requirements are met; Write new requirements (develop new hypotheses) when necessary; Assess what inferences can be made; Identify limitations	https://www.rdocumentation.org/packages/geiger/versions/1.3-1/topics/fitDiscrete
	Track progress, revert changes, and collaborate via remote server Write tests using `testthat` in RStudio Write code to pass tests in `R`	https://tibhannover.github.io/FAIR-R/05-testthat/
11/21/2022	Fall Break - No class	
11/28/2022	Demonstrate creating and maintaining an explicitly inclusive environment; Explain strategies to avoid demotivating learners; Solicit feedback; Give thoughtful and useful feedback	https://carpentries.github.io/instructor-training/17-live/
	Make predictions; Assess the extent to which requirements are met; Write new requirements (develop new hypotheses) when necessary; Assess what inferences can be made; Identify limitations	https://www.rdocumentation.org/packages/geiger/versions/1.3-1/topics/fitDiscrete
	Explain the benefits of software process, especially Agile Software Development; Provide overview of Test-Driven Development (TDD); Write unit tests for software requirements using pseudocode	https://imci-idaho.github.io/2022-03-01-WhatTheyForgot/slides/art-of-workflow.html#1
12/05/2022	Continue group project	
12/16/2022 @ 2:30 PM	Final Presentations and Discussion on Course	[Link to exit survey]

Grades

Those registered for this course will be graded on a Pass/Fail basis. There will be at least five deliverables in this course (creation of repository; descriptive commit messages; demonstration of live coding via final group project; submitted pull requests for two known issues). Students need to deliver on 3 of these five possible deliverables to receive a passing grade. Students may reach out to Breanna with any questions regarding grading.

Funding Acknowledgement

Funding for the development and delivery of this workshop was provided by Idaho EPSCoR and UI IMCI. Much of the content for this workshop was developed and inspired by The Carpentries (https://carpentries.org/). Breanna is a Certified Software Carpentry Instructor.

Disability-related academic adjustments

The University of Idaho is committed to ensuring an accessible learning environment where course or instructional content are usable by all students and faculty. If you believe that you require disability-related academic adjustments for this class (including pregnancy-related disabilities), please contact the Center for Disability Access and Resources (CDAR) to discuss eligibility. A current accommodation letter from CDAR is required before any modifications, above and beyond what is otherwise available for all other students in this class, will be provided. Please be advised that disability-related academic adjustments are not retroactive. CDAR is located at the Bruce Pitman Building, Suite 127. Phone is 208-885-6307 and e-mail is cdar@uidaho.edu. For a listing of services and current business hours visit https://www.uidaho.edu/cdar.

Learning Civility

In any environment in which people gather to learn, it is essential that all members feel as free and safe as possible in their participation. To this end, it is expected that everyone in this course will be treated with mutual respect and civility, with an understanding that all of us (students, instructors, professors, guests, and teaching assistants) will be respectful and civil to one another in discussion, in action, in teaching, and in learning.

Should you feel our classroom interactions do not reflect an environment of civility and respect, you are encouraged to meet with your instructor during office hours to discuss your concern. Additional resources for expression of concern or requesting support include the Dean of Students office and staff (208-885--6757), the UofI Counseling & Testing Center's confidential services (208-885-6716), the UofI Office of Equity and Diversity (208-885-2468), or the Office of Civil Rights and Investigations (208-885-4285).

Inclusivity Statement

As a professor/course instructor at the University of Idaho, I acknowledge the importance of diversity and inclusion and how these attributes contribute to the promotion of a positive educational experience. It is my intent to facilitate a healthy, productive, and safe learning environment where diverse thoughts, perspectives, and experiences are welcomed, and individuals' identities (including, but not limited to: race, sex, class, sexual orientation, gender identity, ability, religious beliefs, etc.) are valued and honored. I recognize that as an educator, it is my responsibility to take the initiative to continually learn about diverse perspectives and identities; therefore, if at any point during the course, you feel uncomfortable or concerned, I am more than willing to discuss suggestions, feedback, and anything else that might improve the general effectiveness of this course.

Firearms Policy

The University of Idaho bans firearms from its property with only limited exceptions. One exception applies to persons who hold a valid Idaho enhanced concealed carry license, provided those firearms remain concealed at all times. If an enhanced concealed carry license holder's firearm is displayed, other than in necessary self-defense, it is a violation of University policy. Please contact local law enforcement (call 911) to report firearms on University property. University of Idaho leadership remains committed to maintaining a safe work, living and learning environment on campus. We will not tolerate any threatening use of firearms or any other weapons. While authorized license holders may have familiarity and be at ease carrying a loaded firearm, we ask that they be aware that many people are not familiar with handguns and are uncomfortable in their presence.

Land Acknowledgement

Uofl Moscow is located on the homelands of the Nimiipuu (Nez Perce), Palus (Palouse) and Schitsu'umsh (Coeur d'Alene) tribes. We extend gratitude to the indigenous people that call this place home, since time immemorial. Uofl recognizes that it is our academic responsibility to build relationships with the indigenous people to ensure integrity of tribal voices.

References

Harmon, Luke J., et al. "GEIGER: investigating evolutionary radiations." *Bioinformatics* 24.1 (2008): 129-131.

https://lukejharmon.github.io/pcm/chapter8_fitdiscrete/

Greg Wilson: "Software Carpentry: Getting Scientists to Write Better Code by Making Them More Productive". Computing in Science & Engineering, Nov-Dec 2006.

Greg Wilson: "Software Carpentry: Lessons Learned". arXiv:1307.5448, July 2013.

Becker, Erin, and François Michonneau. "The carpentries curriculum development handbook." (2021).

McConnell, Steve. "The power of process [software processes]." Computer 31.5 (1998): 100-102.