Using GitHub Classroom To Teach Statistics

Jacob Fiksel⁎, Leah R. Jager⁎, Johannna S. Hardin⁎, and Margaret A. Taub⁎

⁎Department of Biostatistics, Johns Hopkins University, Baltimore, MD; ⁎Department of Mathematics, Pomona College, Claremont, CA

ABSTRACT

Git and GitHub are common tools for keeping track of multiple versions of data analytic content, which allow for more than one person to simultaneously work on a project. GitHub Classroom aims to provide a way for students to work on and submit their assignments via Git and GitHub, giving teachers an opportunity to facilitate the integration of these version control tools into their undergraduate statistics courses. In the Fall 2017 semester, we implemented GitHub Classroom in two educational settings—an introductory computational statistics lab and a more advanced computational statistics course. We found many educational benefits of implementing GitHub Classroom, such as easily providing coding feedback during assignments and making students more confident in their ability to collaborate and use version control tools for future data science work. To encourage and ease the transition into using GitHub Classroom, we provide free and publicly available resources—both for students to begin using Git/GitHub and for teachers to use GitHub Classroom for their own courses.

1. Introduction and Motivation

As more businesses, governments, and researchers make analysis-driven decisions, students of statistics and data analysis should be taught how to collaborate with others in managing data, code, and results that are part of a reproducible analysis pipeline. Version control, a system for organizing and tracking changes to files associated with a project (Tichy 1985; Chacon and Straub 2014), has long been recommended for inclusions in the modern statistics curriculum (Nolan and Temple Lang 2010). More recently, both the American Statistical Association (2014) and the National Academy of Sciences (2018) have emphasized that the modern statistics curriculum should teach students project documentation and collaboration.

To integrate version control into the statistics curriculum, instructors must choose amongst the many formal version control systems that are available (Ram 2013). We recommend using Git. Git has become a widely used tool for enabling collaboration—in an industry-wide survey of over 16,000 data scientists conducted by Kaggle, 58.4% of the 16,000 respondents said that Git was the main tool used for sharing code in their workplace (Kaggle 2017). Furthermore, Git is linked to GitHub, an online hosting service for Git repositories (see Table 1 for definitions of Git and GitHub terms and commands). GitHub has been the most popular hosting service for Git repositories for at least six years (Ram 2013), and currently hosts over 100 million Git repositories (GitHub, n.d.).

Recent work presents clear motivation and examples of using Git for statistics and data analysis. Bryan (2016) has created an impressive and comprehensive website for using GitHub with RStudio. Furthermore, Bryan (2018) argues that incorporating Git and GitHub into data science workflows is considered best practice, and provides thoughtful advice on how to conceptualize the GitHub workflow. Other work (Çetinkaya-Rundel and Rundel 2018) describes a method of integrating Git and GitHub into statistics courses targeted toward students with computational backgrounds. Finally, surveys conducted by GitHub Education (Hsing and Gennarelli 2018) show that using GitHub in the classroom can lead to vastly improved understanding of project management by students. However, the previous literature does not describe best practices for handling the potential multitudes of classroom assignments, nor for introducing version control tools in statistics courses with nonmathematical emphases, such as those in public health or the life sciences.

We believe that version control can and should be integrated into all statistics courses, no matter the target audience. In order to achieve this goal, we advocate for the use of GitHub Classroom. GitHub Classroom is software that aims to provide a way for students to work on and submit their assignments via Git and GitHub, while also giving teachers an opportunity to present version control tools as part of the course material. Benefits of using GitHub Classroom in an educational setting include unlimited private repositories for student work, in compliance with U.S. FERPA (Family Educational Rights and Privacy Act of 1974) regulations, flexible workflows for grading assignments, and ease of distribution of starter materials for various assignments.

However, learning how to use Git and GitHub can be difficult and intimidating, for both instructors and students. Even for those familiar with version control tools, there remains a reasonably steep learning curve associated with GitHub Class-
Table 1. Definitions of terms associated with Git and GitHub. Many definitions are modified from Bryan (2018) and GitHub Help (n.d.)

| Term  | Definition |
|-------|------------|
| Repository | Similar to a folder on Dropbox, except hosted on GitHub. Repositories sync with Git so that users can view a human-readable format of a project’s version control history on GitHub. In GitHub Classroom, each student assignment is a repository. |
| Commit | Rather than saving a file with a new name after file revisions, users can commit their revised work with Git. Each commit is associated with a commit message and a unique ID, allowing users to revisit changes made throughout the project. An example usage would be students committing changed code after writing a new function. |
| Push | After a user makes changes to files within a GitHub repository on their local computer, they can push these changes back to GitHub, so the changes can be seen by other users with access to the repository. This is similar to updating a file on a local computer and syncing the changes to Dropbox. An example usage would be students pushing a completed assignment to GitHub. |
| Pull | If a user wants to obtain the most up-to-date version of a GitHub repository, they can pull the changes to their local computer. This is similar to syncing changed files from a Dropbox folder to a local computer. An example usage would be an instructor pulling updated student assignment repositories. |
| Pull request | If a student makes changes to a shared repository, such as lecture notes, they can submit a pull request for the instructor to review these changes and to choose whether or not to integrate these changes into the shared repository. |
| Branch | If a student clones the course materials to their local computer, they can create their own branch. Changes made on this branch will not be reflected in the main version of the course materials, but can be merged using a pull request. |
| Issue | An issue is a suggested improvement to a GitHub repository, made through GitHub. For each issue, users can discuss the suggested changes in a forum. An example usage would be students submitting an issue for a mistake in an instructor’s lecture notes, within the lectures repository. |
| Clone | Users can clone a full GitHub repository onto their local computer, so they can edit code locally. Similar to downloading a folder from Dropbox, cloned repositories are synced to GitHub repositories, allowing for users to easily push and pull changes. An example usage is students cloning the repository which contains lecture notes. |

room, and implementing it can introduce logistical challenges with respect to weekly homework assignments and projects. As it is a new tool, there are no well-documented and simple workflows published that outline how to best use GitHub Classroom. To that end, we have created easy-to-use and publicly available resources that give step-by-step instructions on implementing GitHub Classroom in any statistics or data analysis course. Our instructions not only help instructors set up their own GitHub Classroom, but also help students learn how to use Git and GitHub. This removes the need for instructors to develop their own lesson plans from scratch on how to teach Git and GitHub.

The main goal of this article is to expand on existing GitHub resources in order to share our recommended workflow for using GitHub Classroom as an educational tool and class management system. We begin in Section 2 with describing the practical and pedagogical benefits of using GitHub Classroom. In Section 3, we describe our experience in implementing GitHub Classroom in two educational settings—an introductory computational statistics (ICS) lab and a more advanced computational statistics (ACS) course. To allow instructors to more easily use GitHub Classroom, we describe the open source and publicly available tools and guides we have developed for using GitHub Classroom in Section 4.1. We have teacher-focused resources, which are targeted toward instructors (of all subjects) who wish to set up a GitHub Classroom, and student-focused resources, which can be distributed by instructors. Both resources provide visual guides to Git, GitHub, and GitHub Classroom for instructors and students who have never used version control before. The remainder of Section 4 discusses key aspects of our workflow for using GitHub Classroom, which are supplemented by our guides. We conclude the article with a brief summary and discussion in Section 5. We believe that our work, along with that of others, will help ease the larger statistics community into using Git and GitHub across the entire statistics curriculum.

2. Pedagogical and Practical Benefits of GitHub Classroom

Because there is a time investment associated with introducing GitHub and GitHub Classroom to students, it is worth discussing why instructors should implement GitHub Classroom to run a course, rather than using the default university course management system (CMS). An immediate advantage is for classes that have group projects. With GitHub Classroom, instructors can easily assign groups of students to teams and give each team their own GitHub repository within a GitHub Classroom. Students can then use Git and GitHub to collaborate on a project, just as they would in an academic or industry research project. Because teachers can see each student’s commit history, it is easy to see how each student contributed to the project. In addition, because instructors can easily apply for unlimited free private repositories associated with a GitHub Classroom, instructors do not have to limit the number of projects throughout a course for monetary reasons.

Even without group projects, however, GitHub Classroom has benefits over standard academic CMSs such as Blackboard, Sakai, and CoursePlus (Zagalsky et al. 2015). First, GitHub Classroom can be used to distribute and update course materials; GitHub can provide course structure without the instructor relying on their CMS (Section 4.6). Students learn the most common Git commands such as clone, pull, and push, while staying up-to-date on course materials. Instructors can also encourage students to contribute to course materials (i.e., correcting mistakes in the lecture notes) through the Git and GitHub infrastructure (Zagalsky et al. 2015), keeping them engaged in the course.

It reduces the amount of work and number of chances for errors during the assignment creation workflow (Section 4.3), relative to a standard CMS. We diagram how GitHub Classroom simplifies the assignment creation process in Figure 1.
Instructors can maintain starter material for assignments on their local computer and can give students their own versions of the starter material with the push of a single button, as opposed to individually uploading each piece of the assignment to the CMS for students to download. Students can use Git to bring the full assignment onto their personal computer, without individually downloading each part of the assignment. We see in Figure 1 how individually downloading files can result in different file structures on each student's computer, which makes it far more likely that the resulting code will have errors or not be reproducible.

Instructors can easily use GitHub Classroom to provide feedback on code while students are working on individual assignments. In our experiences with classes taught without GitHub Classroom, students asked for coding help by either emailing code and data as an attachment, or by scheduling an in-person appointment. To provide help remotely, the professor would have to

1. Download the code and data.
2. Ensure that the data are in the correct directory, as specified by the code.
3. Run the code and provide feedback.
4. Email all documents back as attachments.

By using GitHub Classroom, we provide all students with the same initial directory for each assignment. Instructors are automatically added as collaborators to each repository, and can provide feedback through GitHub's push and pull functionality. We describe the exact workflow later in Section 4.4.

Finally, GitHub Classroom significantly reduces the amount of overhead required to grade and redistribute a large number of assignments (Section 4.5). Figure 2 visually compares the grading workflow when using GitHub Classroom as compared to an academic CMS. By using GitHub Classroom to collect student assignments, the instructor guarantees that the file structure will be identical in each student's assignment directory, making it easier to check student code for reproducibility. Furthermore, instructors can upload every graded file with one keyboard command, as opposed to individually uploading each graded file back to the CMS for students to access. If there are 8 assignments in a semester, and 50 students, an instructor using their default CMS will have to upload a minimum of 400 individual files, as opposed to an instructor using GitHub Classroom who will have to enter only one command per assignment.

3. Experiences With GitHub Classroom

Having discussed the educational value of GitHub Classroom, we now delve into our specific experiences with using GitHub Classroom in two statistics education settings, one introductory and one advanced. Based on our experiences, we offer practical suggestions for introducing and motivating GitHub Classroom to students, which are targeted to student background
knowledge. These more general suggestions supplement the more detailed GitHub Classroom guides that we provide in Section 4.1.

### 3.1. Student Background in Our Two Statistics Courses

Around 20 students took the ICS lab, which was an optional one-credit statistical computing lab associated with an introductory course in biostatistics aimed at public health majors. This course met one hour per week. The vast majority of the students in this course had never done any coding, had never used a command-line interface, and had never heard of Git or GitHub. Seventy students took the ACS course, which met for 3 hr per week; the majority of the students were junior and senior undergraduates majoring in quantitative fields (e.g., statistics, mathematics, economics, and computer science). While most students in the ACS course had used R and RStudio before, very few had any knowledge or experience with Git or GitHub. In both classes, all assignments were completed using the R statistical computing language and coding was done inside the RStudio development environment (IDE).

The two classes had very different curricula due to the vast differences in the statistical and programming knowledge bases of the two student groups. For example, the first assignment in the ICS lab had the students write a for loop in which they simulated from a normal distribution and stored the means of the simulated datasets in a vector. On the other hand, the first assignment in the ACS course was an inferential analysis of a real dataset. We believe the two classes span a wide spectrum of the typical undergraduate material taught by many statistics departments.

### 3.2. Instructor Background in Our Two Statistics Courses

Keeping in mind that we are training the next generation of statisticians, we are well aware of (and most of us belong to!) the previous generation of statisticians. That is, some statistics instructors may not currently be comfortable using Git/GitHub in their own work but want to teach it to their students. None of us were comfortable with the complete suite of Git/GitHub functionality, and all of us learned how to use the basic Git commands (clone, commit, pull, and push; see Table 1) when implementing GitHub Classroom in our courses. Our combined experience is that a rudimentary understanding of these basic Git commands, along with a TA who has worked independently with Git, is sufficient for a successful introduction of Git/GitHub into a statistics class at any level. We refer instructors who want to familiarize themselves with Git and GitHub to the supplementary material in Bryan (2018).
3.3. Student Feedback on GitHub Classroom

In reviewing student feedback from the two courses, we found generally positive feedback from students regarding the use of GitHub Classroom. A student from the ICS lab wrote on their course review, “I enjoyed the ability to constantly pull updates from the Class Material repository and stay up to date with minimal effort.”

We encouraged students to constantly commit (Table 1) their work through Git as they completed assignments (i.e., after each time they wrote or updated a function), so they had a documented record of how they solved problems. Throughout the semester, there were many times when students went back to their commit history to see what they had done or to remind themselves about important changes. A student from the ICS lab wrote “I liked being able to easily see the changes made to the projects and the comments as well.” We cannot overemphasize the importance of tracking student work both for real-time knowledge and for practice in using reproducible methods as part of any data analysis.

Finally, we found that students appreciated learning Git and GitHub so that they could apply it to their own research, or so they could become a more attractive job candidate for data science positions. Previous research indicates that students find learning GitHub benefits them in their careers (Zagalsky et al. 2015). Although all assignments in both classes were in private repositories on GitHub, any work that students want to be made public can be moved to their own public Git presence and can then be shared, for example, with a prospective employer. Below are some comments on the benefits of GitHub received from students on course reviews for the ICS lab:

- “I really enjoyed using GitHub because it's applicable to the other things that I do in my lab”
- “I didn’t mind including [GitHub] in the curriculum for this semester’s course, since it is something that I can now say I know roughly how to use”
- “I liked that we can save our past work on GitHub and that it taught me how to take advantage of it for other projects”
- “This platform is good for a resumé...overall I would say to keep this aspect of the course if possible”

From the ACS course evaluations:

- “GitHub is a skill that I think I will value in the future.”
- “GitHub was awesome.”

Negative student feedback mostly concerned the installation and initial learning curve of Git. The following comments, from the same students in the ICS lab who provided positive feedback above, capture these sentiments:

- “I did not like that [Git] was quite hard to install and that issues occur when you reupgrade your computer.”
- “The only thing I didn’t like about [Git] was getting used to it...going over it in class would probably be more useful.”
- “I did not enjoy the growing pains of [Git], but a more streamlined and effective introduction to it may help alleviate that burden.”

3.4. Instructor Reflections on Using GitHub Classroom

Based on the feedback in Section 3.3, we believe that we gave students in both courses a valuable introduction to version control through the use of Git and GitHub. In the ACS course, students were comfortable enough with the Git and GitHub infrastructure to complete collaborative semester projects through GitHub Classroom, with some of these being made public for viewing by potential collaborators and employers. Students in the ICS lab gained exposure to Git and GitHub, tools widely used in research and industry, in a controlled environment where they were allowed to make mistakes. Students in both of these courses can now easily revisit their old assignments on GitHub, even if they change local computers, meaning they can adapt their old code for new projects. This also allows students to realize that their “future selves” (Bryan 2018) are also considered collaborators, as they can see how detailed commit messages enables future understanding of code. As many nonstatistics majors are required to take at least an introductory statistics course, such as our ICS lab, we believe these courses are opportunity for introducing Git and GitHub to nonstatistics majors through GitHub Classroom.

3.4.1. Motivating the Use of GitHub Classroom

An important lesson learned in implementing GitHub Classroom in the two courses is how the introduction of version control should be modified based on the student background. First, different instructors may choose different methods to motivate the use of Git and GitHub. While the importance of version control and reproducibility are indeed huge benefits of using Git and GitHub, they are also abstract concepts to undergraduate students, especially those with minimal statistical and computational experience like the students in the ICS lab.

In the ICS lab, we motivated the use of Git and GitHub by promoting the advantage of being able to put the skills on a resumé. As students found this to be an important benefit of learning Git and GitHub, we recommend this as a future strategy for motivating Git in introductory courses. While we also discussed the importance of reproducible research and collaborative coding, the limited research experience of our students meant that ideas of “best practices” concepts were more theoretical than practical to them. We emphasized the growing use of tools used for reproducibility in computing and business.

In the ACS course, we motivated the use of Git through lectures on the reproducibility crisis, giving examples of research projects that went awry (Coombes, Wang, and Baggerly 2007; Chakrabarti, Topf, and Schroth 2013; Kern et al. 2013). In class, we discuss that Git is not perfect and can be difficult to learn, but can be helpful in avoiding issues such as lack of reproducibility. If the students want to pursue quantitative work in fields outside of statistics and data science, such as biology and economics, in today’s world, they need to learn how to use today’s tools.

3.4.2. Modifying the Introduction of Version Control in Introductory Classes

Based on student feedback on the difficulty of installing and getting started with Git and GitHub, in the future we would not introduce these topics at the very beginning of a course targeted toward students without prior computational experi-
ence. Instead, we would wait until at least the second half of the course when students have more familiarity with coding and are more comfortable with debugging errors in R (or whichever software is being used in the course). We would have a specific unit on using Git and GitHub for version control, addressing student concerns on the inadequate amount of class time devoted to Git and GitHub in the ICS lab. Students could then focus on learning the Git skills without also having to focus on starting to learn a new coding language. One strategy for integrating the material learned in the Git unit with the rest of the class would be to assign a final team project within GitHub Classroom that required students to collaborate using a shared GitHub repository, demonstrating to students the benefit of using GitHub for collaboration (as discussed in Section 2 and in Bryan (2018)). Instructors that use this strategy would then have to teach students how to solve problems that arise with using GitHub for collaboration, such as when two students make conflicting changes to the same line of code (also known as a merge conflict). As these problems are specific to GitHub, rather than GitHub Classroom, we refer instructors to Bryan (2018) for suggestions on preventing and solving merge conflicts and other technical issues associated with GitHub collaboration.

4. Resources and Workflow for Implementing GitHub Classroom

4.1. Publicly Available Guides

Throughout the start-up and use of GitHub Classroom, we documented our workflows for setting up a GitHub Classroom, sending out assignments, and grading assignments. Our workflows are publicly available in our GitHub Classroom Guide for Teachers, which includes GIFs for each step as a visual guide to supplement the written instructions (we provide a modified, text-only version in the supplemental material). Furthermore, we have made a detailed GitHub Classroom Guide for Students, complete with written instructions, GIFs, and YouTube videos which will guide students with limited-to-no computing experience through the process of setting up Git and GitHub. These guides can be accessed through the following URLs:

- https://github.com/jfiksel/github-classroom-for-teachers
- https://github.com/jfiksel/github-classroom-for-students

In the following sections, we will outline key aspects of our workflow and our guides.

4.2. Creating a GitHub Classroom

A GitHub Classroom is organized around GitHub Organizations. A GitHub Organization is a way to share GitHub repositories among multiple users. In creating a classroom, you create an organization for the specific class and semester (i.e., Intro to Statistics Fall 2019), and then link it to the GitHub Classroom software as shown in our guides. Throughout the semester, all student assignments are created as GitHub repositories within the given organization. Our guides demonstrate how to easily apply for unlimited private repositories, as discussed in Section 2, allowing instructors to use GitHub Classroom without monetary costs.

GitHub does not provide a way for students to add themselves to the organization without the permission of the instructor. To work around this, we recommend first providing the whole class with a link to an assignment (we will more fully describe this in Section 4.3.2). By clicking on the link, students add themselves to the organization.

4.3. Managing and Creating Assignments

4.3.1. Using Master Organizations for Starter Code

The main purpose of GitHub Classroom is to automate the process of creating and assessing assignments which reside in separate GitHub repositories for each student (or team). Furthermore, each assignment can be based upon a previously existing repository with starter code.

To best take advantage of the GitHub Classroom functionality, we recommend having a “master” organization which serves as a shared account between course instructors to manage starter code for assignments. If you have a class called Introduction to Statistics, you would have one master organization called intro-statistics-master, and then organizations for each iteration of the class (i.e., intro-statistics-fall-2019, intro-statistics-fall-2020, etc.). You would then have repositories that contain starter code for each assignment within the master organization. If an instructor wants to change the starter code for an assignment between iterations of the class, they can use the basic Git commands (Table 1) to do so. Furthermore, assignments can easily be shared with other instructors by either adding the instructors to the organization, or by giving them access to individual assignment repositories. This is especially useful if the instructor for a class changes, as the new instructor can be added to the master organization and does not have to be added to an organization for one of the previous class iterations.

4.3.2. Creating Assignments with GitHub Classroom

To create assignments for every student in a class, instructors use the GitHub Classroom software (available online) and click the “New assignment” button. After choosing whether the assignment will be an individual or team assignment, instructors provide GitHub Classroom the name of the assignment, and, optionally, a starter repository (ideally from the master organization). The instructor is then given a link to the assignment, which the instructor provides to students. Note that the assignment creation process is nearly identical for both individual and group assignments. For individual assignments, when a student clicks on the link, an assignment repository is created for that student, with the student’s GitHub username as part of the repository name, within the class organization. When a student clicks on the link for a team assignment, GitHub Classroom will first prompt the student to either join an existing team or create a new team, and will then create assignment repository with the team’s name as part of the repository name.

As mentioned in Section 1, GitHub allows for unlimited private repositories within education based organizations. GitHub Classroom gives instructors the option to make each assignment
private during the assignment creation process. This means that each student (or team) can only view their own repositories, while the instructors (and TAs) have access to all repositories. By using the functionality of GitHub Classroom, instructors comply with FERPA privacy rules by ensuring that homework assignments are not viewed publicly. If students want to share their work to prospective employers, they can make their repositories publicly available. After the final team project in the ACS course (Section 2), several teams made their projects available to the public for increased exposure.

4.3.3. Managing Assignments Over Time with GitHub Classroom

Figure 3 illustrates the concepts of maintaining a master organization over time and using starter code to create assignments for each class iteration. To demonstrate, we have a master organization called “intro-statistics-master”, which contains one repository (“HW1”) in our example, although it can contain an unlimited number of repositories (assignments). The instructor assigns HW1 to each student in the Fall 2019 iteration of the course.

After the 2019 course, the instructor updates the data used for the assignment on their local computer, and then pushes this change to the repository in the master organization. Because the change was introduced to the starter code in the repository owned by the master organization, all of the student repositories from the Fall 2019 course are unchanged. The instructor creates a new class-specific organization for the Fall 2020 course, and gives the students in the 2020 class starter code for HW1, which uses the updated data.

Finally, suppose a new instructor takes over after the 2020 course ends. This new instructor completely updates the HW1 starter code by adding an additional analysis, and pushes all the changes to the repository in the master organization. The new instructor then creates a class-specific organization for the 2021 course, and gives each student the updated starter code for their assignment.

4.4. Giving Feedback During Assignments

As mentioned in Section 2, one of the main benefits of using GitHub Classroom is the ability to provide feedback in the middle of assignments. If a student is regularly making commits (although this is not a given in an introductory course), instructors can see the specific steps a student has taken to solve a problem. More importantly, however, by cloning a student’s GitHub Classroom repository, instructors retain the exact same file structure of data and scripts that the student has. Rather than downloading every piece of data and R script, as described in Section 2, instructors can run the code that the student has implemented to see where the problem with the student’s approach lies. The workflow that we employed has the instructor:

1. Clone the student’s directory to their own computer. Or pull the latest changes if the repository has been previously cloned.
2. Provide feedback through either a pull request, directly into the code, or as an issue in GitHub.
3. Push all changes back to GitHub.

Figure 3. Using a master GitHub Classroom organization improves management of course material. An instructor who wishes to make a change to an assignment would make a change to the assignment in the master organization (red, bold, and italicized font denotes a changed file), which will then be present when creating new assignments. In this example, the instructor changed the dataset for the 2020 course iteration. In 2021, the new instructor added an additional analysis, which resulted in changing the assignment instructions and starter code.
All feedback is then documented in the commit history, which can be a useful reference if the student wants to look back at the assignment after completion.

4.5. Grading Workflows

We have identified two potential ways to grade assignments from GitHub Classroom. The first method, which we did not use, is through the use of pull requests directly on GitHub. The pull request method is detailed in a GitHub blog post (Gennarelli 2017). The advantage to this approach is that instructors can provide line-by-line code comments, along with syntax highlighting, that is easily viewable by the student after grading. The immediate downside is that it requires manual overhead work, as the instructor has to click through each assignment and go through the pull request process on GitHub. GitHub Organizations do not organize assignment repositories in a convenient way (yet), and the number of repositories to keep track of can quickly become overwhelming—in the end, the ICS lab had 256 repositories in their class organization (Figure 4), and the ACS course had 878 repositories. GitHub classroom makes tracking repositories more manageable by allowing you to click on an assignment inside of the GitHub Classroom portal and then list all student repositories for this assignment, but it does not have an automated way of pulling the large number of repositories. Yet another downside to the “feedback through pull requests” approach is that the instructor does not have the code locally to run. However, integration tools such as Wercker can automatically check for reproducibility of a student’s assignment directly from GitHub, meaning instructors do not have to clone assignments onto their local computers to run the code (Çetinkaya-Rundel and Rundel 2018).

The second method for grading assignments, which we took advantage of, used a shell script https://github.com/jfiksel/mass_clone (modified from Konzy (2018)) to automatically clone all assignments to our local computers. We could then run all student code locally and provide comments directly on each student’s coding script. Detailed instructions, including optimal directory set-up, is available under the “Grading assignments” section of our GitHub Classroom Guide for Teachers. We give our method step by step below:

1. Clone all student assignment repositories to a local computer using our shell script
Students see instructor feedback in GitHub. After an instructor has provided feedback in a student's assignment, the instructor then commits the feedback and pushes the updated file to the student's assignment repository. By clicking on the commit message, the student then sees the feedback given by instructors, which is highlighted in green.

2. Open each student's assignment (all assignments can be opened simultaneously with a single command on the terminal command-line), and run the code inside of RStudio to ensure reproducibility. Add comments, suggestions, or edits within each student’s R script or Rmd file and save the altered file.

3. Use the shell script in step 1 to simultaneously add, commit, and push all edits for all student assignments. This step is done with one line of “command line” code, and students can view highlighted comments by visiting their assignment repository on GitHub and clicking on the commit (Figure 5), or by pulling the latest version of the repository.

The choice of grading method will most likely be determined by course size, emphasis on code reproducibility, and individual preference. For courses where grading is assisted or done solely by teaching assistants, instructors will have to ensure that teaching assistants are comfortable with Git and GitHub and have basic understanding of command line syntax used in the terminal.

4.6. Using GitHub Classroom to Distribute Lecture Materials

Both classes used GitHub Classroom not only for assignment distribution and grading but also for updating and sharing course material. Since class periods often consisted of live coding, lecture material was updated in real-time and could not be shared in its final version until after each class session was complete. So, in addition to the assignment repositories, we also each maintained a single repository within the class organization for class materials. Our guides describe how to give all students access to the class materials repository while also keeping it private from GitHub users outside of the class. Instructors can also choose to keep this specific repository public, if they wish to share lecture materials with those outside of the class. In our setup, the class materials repository has a folder for each class meeting, containing starter code and data to be used that day. Students pulled the material before class to follow along. After class, the instructor would push the new material (code written during the class) to the repository, and students could again pull the material to their own repository to get the complete files produced during the class. This not only helped students stay up to date with the class, but it also gave them weekly practice in using GitHub.

Using GitHub Classroom for distributing lecture materials also facilitates student engagement, by encouraging them to spot and fix lecture errors or code bugs, and then using GitHub tools to integrate their changes into the shared course repository. If a student wishes to make a change to the course material, instructors can use this opportunity to teach the class how to commit their changes back to the class branch, push these changes back to GitHub, and then use a pull request to have these changes reviewed for incorporation. Students not only become more invested in the class, but also learn common, yet advanced tools for collaboration on GitHub.

5. Conclusion

Previous work has shown that GitHub can be used for educational purposes across a range of subjects, class sizes, and instructor knowledge of GitHub (Zagalsky et al. 2015). The contribution of our work is to provide a concrete and easy-to-implement workflow for instructors who want to bring version control into their classroom. By using our recommended workflow, instructors will not only benefit their students by teaching them skills desired by potential employers but will also significantly cut down on the administrative work required to distribute, grade, and return assignments. By saving time that was formerly spent on administrative duties, instructors can spend more time working with students and updating course material.

Through our experiences, we show that the Git workflow can be used in both introductory and advanced courses and by instructors without previous GitHub experience. We have used student feedback to construct a separate guide to Git and GitHub for students, so that instructors unfamiliar with version control do not have to create their own teaching materials. Our hope is that our guides serve as a starting point for instructors to use GitHub, who will then modify and improve our workflows for different class settings.

References
American Statistical Association and others (2014), "Curriculum Guidelines for Undergraduate Programs in Statistical Science," available at http://www.amstat.org/education/curriculunguidelines.cfm/.
Bryan, J. (2016), "Happy Git and GitHub for the useR," available at http://happygitwithr.com/.
Bryan, J. (2018), "Excuse Me, Do You Have a Moment to Talk About Version Control?" The American Statistician, 72, 20–27.
Çetinkaya-Rundel, M., and Rundel, C. (2018), "Infrastructure and Tools for Teaching Computing Throughout the Statistical Curriculum," The American Statistician, 72, 58–65.
Chacon, S., and Straub, B. (2014), Pro git. Apress.
Chakrabarti, R., Topf, H.-G., and Schroth, M. (2013), "Retraction Note To: Low Dose Lidocaine for Refractory Seizures in preterm Neonates," The Indian Journal of Pediatrics, 80, 529–529.
Coombes, K. R., Wang, J., and Baggerly, K. A. (2007), "Microarrays: Retracing Steps," Nature Medicine, 13, 1276.
Gennarelli, V. (2017), "How to Grade Programming Assignments on github," available at https://blog.github.com/2017-06-13-how-to-grade-programming-assignments-on-github/
GitHub (n.d.), "Github facts," available at https://github.com/about/facts.
GitHub Help (n.d.), "Github Glossary," available at https://help.github.com/articles/github-glossary/.
Hsing, C., and Gennarelli, V. (2018), "2018 Classroom Report," available at https://education.github.com/classroom-report/2018.
Kaggle (2017), "The State of Data Science & Machine Learning 2017," https://www.kaggle.com/surveys/2017.
Konzy (2018), "Mass Clone," available at https://github.com/konzy/mass-clone/.
National Academies of Sciences, Engineering, and Medicine and others (2018), Data Science for Undergraduates: Opportunities and Options. Washington, DC: National Academies Press.
Nolan, D., and Temple Lang, D. (2010), "Computing in the Statistics Curriculum," The American Statistician, 64, 97–107.
Ram, K. (2013), "Git Can Facilitate Greater Reproducibility and Increased Transparency in Science," Source Code For Biology and Medicine, 8, 7.
Tichy, W. F. (1985), "Rcs System for Version Control," Software: Practice and Experience 15, 637–654.
Zagalsky, A., Feliciano, J., Storey, M.-A., Zhao, Y., and Wang, W. (2015), "The Emergence of github as a Collaborative Platform for Education," In Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing, pp. 1906–1917. Vancouver, BC: ACM.