Introduction to Git for Data Science

**Greg Wilson**

Co-founder of Software Carpentry

### Course Description

Version control is one of the power tools of programming. It allows you to keep track of what you did when, undo any changes you have decided you don't want, and collaborate at scale with other people. This course will introduce you to Git, a modern version control tool that is very popular with data scientists and software developers alike, and show you how it can help you get more done in less time and with less pain.

# Basic Workflow

This chapter explains what version control is and why you should use it, and introduces the most common steps in a common Git workflow.

|  |
| --- |
| **1.1 Want is version control?**  A version control system is a tool that manages changes made to the files and directories in a project. Many version control systems exist; this lesson focuses on one called Git, which is used by many of the data science tools covered in our other lessons. Its strengths are:  Nothing that is saved to Git is ever lost, so you can always go back to see which results were generated by which versions of your programs.  Git automatically notifies you when your work conflicts with someone else's, so it's harder (but not impossible) to accidentally overwrite work.  Git can synchronize work done by different people on different machines, so it scales as your team does.  Version control **isn't just for software**: books, papers, parameter sets, and anything that changes over time or needs to be shared can and should be stored and shared using something like Git. |
| **1.2 Where does Git store information?**  Each of your Git projects has two parts: the files and directories that you create and edit directly, and the extra information that Git records about the project's history. The combination of these two things is called a repository.  Git stores all of its extra information in a directory called .git located in the root directory of the repository. Git expects this information to be laid out in a very precise way, so you should never edit or delete anything in .git.  Suppose your home directory /home/repl contains a repository called dental, which has a sub-directory called data. Where is information about the history of the files in /home/repl/dental/data stored?  Answer: /home/repl/dental/.git (all of the information about a repository is stored under its root directory) |
| **1.3 How can I check the state of a repository?**  When you are using Git, you will frequently want to check the status of your repository. To do this, run the command ***git status***, which displays a list of the files that have been modified since the last time changes were saved.  You have been put in the dental repository. Use git status to discover which file(s) have been changed since the last save. Which file(s) are listed? |
| 1.4 How can I tell what I have changed?  Git has a staging area in which it stores files with changes you want to save that haven't been saved yet. Putting files in the staging area is like putting things in a box, while committing those changes is like putting that box in the mail: you can add more things to the box or take things out as often as you want, but once you put it in the mail, you can't make further changes.    git status shows you which files are in this staging area, and which files have changes that haven't yet been put there. In order to compare the file as it currently is to what you last saved, you can use git diff filename. git diff without any filenames will show you all the changes in your repository, while git diff directory will show you the changes to the files in some directory. |
|  |
|  |