# NR995 Module 9

**2017 Fall**
**Part I: Introduction to Version Control Systems**
**Why do we need version control?**

Version control is a way to keep track of progress & changes to a set of files. Whether you're collaborating or working alone, it's a good idea to maintain versions of your files through time. This could mean R scripts for data cleaning, analysis, or plotting, but it also includes successive drafts of thesis chapters or manuscripts. It's also a very good idea to keep secure backups in case your computer crashes.

Depending on where you are in your career as a scientist, you might have used a system where you append a date, version number, and/or initials to files as they change. You end up with a huge pile of documents that gets confusing even if you somehow manage to keep it organized. Backing your files up requires either manually copying files onto an external hard drive every so often, or using a service like Dropbox to sync everything. With a lot of care, this can accomplish the tasks of tracked changes + backup, but it's a lot of work, it's very error prone, and if you do need to restore previous work, it can be very difficult to find the correct version that you're looking for.

Version control systems were developed by software developers to track their software projects. The main issues have already been solved and as biologists become increasingly computer-reliant, we can take advantage of these thoroughly tested resources to improve transparency in the scientific process, to increase access to code, and to ultimately save a lot of time and frustration once the initial investment in learning the programs has been made.

## Version control systems overview

These are the basic structures of VCSs:

- Local
- Centralized
    - Google docs are an example of a centralized VCS. You get a lot of benefits like revision history, simultaneous editing, etc. You can work from a snapshot on your local system and then re-upload, but you need to have access to the internet to access the revision history and you run the risk of working from an out-dated version (i.e., someone else editing the document *after* you downloaded it) and overwriting their work.
- Distributed
    - Full version database on *each* computer

# Git

Git is a distributed VCS Git works by taking full snapshots of files that have changed. You can use git completely on your local computer to track your project files, but there are several great cloud-based services for connecting your local repository to a remote repository. They're free, it gives you a full backup, and they have a bunch of extra features, so there's no reason not to.
This is a typical workflow when you're using git:

- Work like usual
- Save like usual
- Stage changes -- tell git to prepare to store this version
- Commit changes -- tell git to store this version; requires commit message
- Push changes -- update the commit(s) from the local repo to the remote repo
- Pull changes -- update any commits from elsewhere to the local repo