Github

This tool backs up your code to a cloud “repository”. It has at least three components: the cloud repository, a local repository, and software to manage the repositories. Github.com is the most widely used cloud repository. Detailed instructions on how to get started with GitHub are available. In brief, the steps are:

1. Install git on your computer and create a GitHub account. Note that GitHub accounts are public unless you pay for a private account. The University of Texas also has an institutional repository for UT employees.
2. Download and install a client to manage your repository. One possibility is GitHub Desktop (streamlined), another is Sourcetree (fancier). Alternatively, you can control the repository using your command line.
3. Create a new repository or clone an existing one. If you want to take code that is existing on your computer and back it up to the cloud, then you’ll want to create a new repository. If you want to download code that someone else has stored in a repository, then you want to clone a repository.
4. After you create a new repository and/or after you have edited files on your local computer, you will want to “commit” your changes to your local version of the repository. This involves two steps: adding or staging the file and committing the staged changes. When you commit the changes you’ll include a (usually brief) comment about what changes you made to the code.
5. To back up the changes to the cloud, you’ll “push” the changes to the cloud repository.

Say that a collaborator has made changes to the code and you want to access the revised code. You will want to “pull” the changes from the cloud to your local repository. Generally you will want to get into the habit of pulling changes from the cloud repository before you make any local changes to the code. If you make changes to a file that was changed in the cloud repository since you last updated your files, you’ll need to reconcile and merge the two versions of the code. It’s not a disaster, which is a good thing because it will happen. When it happens you will have to learn more of your client’s features for reconciling conflicts.

For similar reasons you will want to be sure to commit AND PUSH your changes to the repository frequently, at least daily.

How does GitHub help?

1. It keeps a record of all of the earlier versions of your code making it trivial to roll changes back to an earlier version. There is no reason to litter your directories with multiple versions of analytical files.
2. You can tag versions of the code when you submit a paper so that you are always in a position to replicate a specific set of earlier results.
3. You and your collaborators’ analysis files are in sync.
4. Your analysis files can be synced across all your computers.
5. The comments you write when you commit changes help to document code development.
6. You can share your code for replication.
7. It is easy to create a branch of a project when you start a new analysis that builds on old files.

There's a Nature blog posting with more detail on how git can help,

How do I use GitHub?

A few basic pointers with illustration are available on my GitHub Illustrated page. In addition, other people have produced tutorials on Version Control, Basics of using Git, and workflow using Git. I'm not aware of a tutorial of social scientists. If you find one, please let me know.

Differences between GitHub.com and The University of Texas’s repository (github.austin.utexas.edu)

GitHub.com is a location where you may store your open repository for free and you can explore the open repositories of other individuals and research teams that archive their materials there. For example, I just found a repository for DHS indicators, which provides stata "code for all DHS Program indicators listed in the Guide to DHS Statistics" by searching for "demography." It is possible to pay to create a private repository should you not want to make your code openly available immediately. Note that even open repositories are only readable to the world. One can't alter an open repository without becoming a collaborator, but one can "fork" a repository to create a copy for oneself that remembers its origins.

The University of Texas repository is designed to be used only by UT employees and is a secure location for code that you want to keep within a team of people employed by UT-Austin (including people with zero-time appointments). Each person needs to login to github by going to github.austin.utexas.edu and using their uteid and password to login. After that one can create and be added to existing github.austin.utexas.edu repositories. Note that people not employed at UT cannot access the files in this repository. You can (and should) move a repository to GitHub.com when you leave UT Austin.