SOC 4650/5650 User's Guide

Christopher Prener, Ph.D.

2016-11-12

Contents

Preface **5**
License . . . 5

1 Getting Started **7**
1.1 Prep Your Computer . . . 7
1.2 Create Accounts . . . 7
1.3 Download and Install Software . . . 8
1.4 Buy Course Materials . . . 8
1.5 Download Course Data . . . 9

2 Approaching this Course **11**
2.1 Zen and the Art of Data Analysis . . . 11
2.2 An Apple a Day . . . 11
2.3 Reading with Purpose . . . 12
2.4 Active Lectures and Labs . . . 12
2.5 Typefaces, Fonts, Files, and Examples . . . 12

3 "Good Enough" Research Practices **15**

4 Protecting Your Work **17**
4.1 Creating a Sustainable File System . . . 17
4.2 Backing Up Your Data . . . 21

5 Introduction to GitHub **23**
5.1 Git . . . 23
5.2 More Git-lingo . . . 23
5.3 GitHub.com . . . 23
5.4 GitHub Repositories . . . 24
5.5 Storing GitHub Repositories . . . 24
5.6 GitHub Issues . . . 24
5.7 GitHub Desktop Application . . . 25
5.8 Learning More . . . 25

6 Final Words **27**

Preface

This text is a companion document for **SOC 4650/5650 - Introduction to Geographic Information Sciences**. It is designed to help you be *successful* in this course. The idea behind a course **User's Guide** is to create a reference for many of the intangible, subtle or disparate skills and ideas that contribute to being a successful researcher. In creating a **User's Guide**, I draw inspiration from the work of Donald Knuth.[1] Knuth has discussed his experiences in designing new software languages, nothing that the developer of a new language

> …must not only be the implementer and the first large-scale user; the designer should also write the first user manual… If I had not participated fully in all these activities, literally hundreds of improvements would never have been made, because I would never have thought of them or perceived why they were important…

While there is nothing particularly new about what I am writing here, and I am certainly not developing a new language for computing, the goal of the **User's Guide** remains similar to Knuth's experience. By distilling some of key elements for making a successful transition to being a *professional developer* of knowledge rather than a *casual consumer*, I hope to both improve the course experience itself and also create an environment that fosters a successful learning experience for you.

If you read through the course objectives included in the syllabus, you will note that creating maps is only one of them. As much as this is a GIS course, it is a course in research methods. In particular, we are concerned with *high quality* research methods and the *process* of conducting research. We therefore focus on a combination of mental habits and technical practices that make you a successful researcher. Some of the skills and techniques that we will discuss this semester are not taught as often in graduate programs. Instead, they are often the products of "learning the hard way". These "habits of mind and habits of method" are broadly applicable across methodologies and disciplines.

License

[1] Donald Knuth is the developer of TeX, a computer typesetting system that is widely used today for scientific publishing in the form of LaTeX. He also established the concept of literatue programming, which forms the basis of some of the practices we will follow with Stata this semester.

Chapter 1

Getting Started

Before you begin the semester, there are a number of things that I recommend that you do to help set yourself up for success. Before you do *anything* else, you should read through the **Syllabus** and the **Reading List**. Make sure you have a good sense of what is *required* for the course. If you have questions, bring them to the first day of class!

1.1 Prep Your Computer

Before you do anything else for this course, make sure you get your computer ready for the work you are about to undertake:

1. Make sure your operating system is up-to-date. If you are able, I would also recommend upgrading your computer to the most recent release of its operating system that the computer can run.
2. We'll be sharing computer files throughout the semester, so you should ensure that you have functioning anti-virus software and that it is up-to-date.
3. You'll also need to download files, so you'll need to make sure you have some free space on your hard drive. If you have less than 10GB of free space, you should de-clutter!
4. Make sure you know how to access your computer's file management system.
    - On macOS, this means being comfortable with Finder.app.
    - On Windows, this means being comfortable with Windows Explorer.

This of course assumes that you own a computer. Owning a computer is not required for this course. All students who are enrolled in SOC 4650 or SOC 5650 will be given 24-hour swipe access (*just what you always wanted!*) to Morrissey Hall to facilitate access to lab computers.

1.2 Create Accounts

There are two major web services that we will use this semester, and you'll need to create accounts for both:

- **GitHub** - you can sign-up at GitHub.com. Once you've signed up, fill out your profile, set-up two-factor authentication, and let Chris know (via email) what your user name is. Once he has it, he can add you to the SOC 4650/5650 organization.
- **Slack** - you can ask Chris (via email) for an invitation to sign-up for our team. Once the sign-up process is complete, you can log-in by going to our team's Slack site. Fill out your profile, set-up two-factor authentication, and change your timezone.

1.3 Download and Install Software

There are a number of software applications that we will use this semester. Most of them are free, and I recommend downloading those free ones right away. All of these applications are available for macOS and Windows.

- **Atom** - Atom is a flexible, open-source text editor that is produced by GitHub. You can download it from Atom's website.
- **GitHub Desktop** - GitHub makes a desktop client that you can use to easily interact with repositories that are stored on the site. You can download it from GitHub's website after you sign-up for an account there. You'll need that account information to complete the desktop client's set-up process.
- **Slack** - Slack has a number of applications for desktop and mobile operating systems. I recommend downloading Slack on your personal computer, and optionally installing it on your mobile device as well. You can download their desktop applications from their website and the mobile applications from your App Store.

For Graduate Students *only*

If your computer meets the operating system requirements for ArcGIS and you think you'd benefit from having access to the software at home, let Chris know (via email).

If you are in the Public and Social Policy Ph.D. program and your computer meets the hardware and software requirements for Stata, you should consider purchasing it for yourself. I recommend purchasing a perpetual license for Stata/IC. This is the most cost-effective solution for typical students.

1.4 Buy Course Materials

Books

There are three required books for this course:

1. Brewer, Cynthia. 2015. *Designing Better Maps: A Guide for GIS Users*. Redlands, CA: ESRI Press. ISBN-13: 978-1589484405; List Price: $59.99; ebook versions available.
2. Gorr, Wilpen L. and Kristen S. Kurland. 2013. *GIS Tutorial 1: Basic Workbook*. 10.3.x edition. Redlands, CA: ESRI Press. ISBN-13: 978-1589484566; List Price: $79.99; ebook versions available.
3. Thomas, Christopher and Nancy Humenik-Sappington. 2009. *GIS for Decision Support and Public Policy Making*. Redlands, CA: ESRI Press. ISBN-13: 978-1589482319; List Price: $24.95.

There is one additional book that is optional:

- Mitchell, Michael N. 2010. *Data Management Using Stata: A Practical Handbook*. College Station, TX: Stata Press. ISBN-13: 978-1597180764; List Price: $48.00.

Buying Mitchell (2010) is *highly* recommended for graduate students who will continue using Stata in the future and those who are concerned about the command-line interface. I recommend waiting for a week or two before purchasing this.

External Media

You will need a USB external storage device (either an external hard drive or a thumb-style drive) that has at least 20GB of storage capacity. This will be used for storing spatial data for this course.

1.5 Download Course Data

Mots of the course data is available for download via Dropbox in a single `.zip` file. If you want, you can let Chris know (via email) that you'd like to download these data before the beginning of the semester. Once you download them, extract the data from the `.zip` file and transfer them to your external storage device.

Chapter 2

Approaching this Course

Students have varying experiences learning GIS techniques. For some, the spatial logic and programming that are the foundation for GIS methods come naturally. For others, being introduced to these concepts can be an anxiety producing experience. I am fond the phrase "your mileage will vary" for describing these differences - no two students have the exact same experience taking a methods course.

2.1 Zen and the Art of Data Analysis

One of the biggest challenges with this course can be controlling the anxiety that comes along with learning new skills. ArcGIS processes, Markdown syntax, and Stata commands can seem like foreign alphabets at first. Debugging Stata do-files can be both challenging and a large time suck, in part because you are not yet fluent with this language. Imagine trying to proofread a document written in a language that you only know in a cursory way but where you must find minute inconsistencies like misplaced commas.

For this reason, I also think it is worth reminding you that many students in the social sciences struggle with quantitative methods at first. It is normal to find this challenging and frustrating. I find that students who can recognize when they are beginning to go around in circles are often the most successful at managing the issues that will certainly arise during this course. Recognizing the signs that you are starting to spin your wheels and taking either ten minutes, an hour or two, or a day away from GIS coursework is often a much better approach than trying to power through problems.

2.2 An Apple a Day

Being able to walk away from an assignment for a day requires excellent time management. If you are waiting until the night before or the day of an assignment's due day to begin it, you give yourself little room for errors. I recommend approaching this course in bite size chuncks - a little each day. The most successful students do not do all of their reading, homework, and studying in a single sitting. I find that this approach not only creates unnecessary anxiety around assignments, it also dramatically limits the amount of course material you can absorb. Keep in mind that I expect the *median* student to spend approximately six hours on work for this class each week (twice the amount of in-class time).

A sample approach to the class might look something like this:

- Tuesday: class
- Wednesday: finish lab
- Thursday: Start problem set
- Friday: Finish problem set

- Saturday: First reading
- Monday: Second reading

2.3 Reading with Purpose

The book and article **reading assignments** for this course are different from most of the other reading you will do in your graduate program because they are often very technical. Students who are most successful in this course read twice. Read the first time to expose yourself to the material, then take a break from the reading. During this first read, I don't recommend trying to complete the example problems or programming examples. Focus on the *big picture* - what are the concepts and ideas that these readings introduce?

During the second read, try to focus in in the *details* - what are the technical details behind the big picture concepts? I recommend doing this second read with your computer open. Follow along with the examples and execute as much of them as you can. By using this second read through as a way to test the waters and experiment with the week's content, you can come into the lecture better prepared to take full advantage of the class period. Students who follow this approach are able make important connections and focus on the essential details during lectures because it is their third time being exposed to the course material. They are also in a much stronger position to ask questions.

2.4 Active Lectures and Labs

During **lectures**, I introduce many of the same topics that your readings cover. This again is intentional - it gives you yet another exposure to concepts and techniques that are central to geospatial science. One mistake students sometimes make is focusing on the details of *how* to do a particular task rather than focusing on *when* a task should be done. If you know when a task is needed but cannot remember how to do it in Stata or ArcGIS, you can look this information up. Conversely, detailed notes on executing Stata commands may not be helpful if you are unsure when to use a particular skill. There is no penalty in this course for not knowing how to execute a command from memory; this is what reference materials are for. The most successful students will therefore focus on *when* a particular skill is warranted first before focusing on *how* to execute that skill

Getting experience with executing tasks is the purpose of the **lab exercises**. Time for beginning these exercises is given at the end of each class meeting, and replication files will be posted on GitHub for each lab.

2.5 Typefaces, Fonts, Files, and Examples

2.5.1 Typefaces and Fonts

Stata publications use a `monospaced typewriter style typeface` to refer to Stata commands (inputs) and Stata results (outputs). I take the extra step of highlighting commands with a when they are referenced in a sentence. In some documents, like lecture slides and cheat-sheets, I may highlight a command by using a to increase the visibility of the command name itself.

The `typewriter typeface` is also used to refer to filenames (e.g. `auto.dta`) or filepaths (e.g. `C:\Users\JSmith\Desktop`). Finally, we will use the `typewriter typeface` to refer to GitHub repositories (e.g. `Core-Documents`, the repository that contains this file).

Stata publications use *italicized text* to refer to text that is meant to be replaced. These references will typically appear in a `typewriter typeface` since they are often part of commands. For example, `describe`

`varname` (with `varname` *italicized*) indicates that you should replace the text `varname` with the appropriate variable name from your dataset.

Stata publications use a sans serif typeface to refer to areas of the Stata user interface, menu items, and buttons. Stata publications also use a sans serif typeface to refer to keyboard keys (e.g. Crtl+C) where the plus sign (+) indicates that you should press multiple keys at the same time.

A sans serif typeface combined with a right facing triangle-style arrow (>) is used to refer to actions that require clicking through a hierarchy of menus or windows (e.g. File > Save).

FYI, Since you are reading this document rendered as a Markdown file, you can't see exact examples of what a sans serif typeface looks like.

2.5.2 Stata Files

There are a number of file types that are important for our use of Stata. These are all likely file types that you have never come across before, and are all discussed in greater detail in the Introducing Stata chapter (see page).

- `.do` - "Stata Do-files" - These are code files that contain commands that Stata can execute automatically. All final analyses and manipulations for research should be done via do-files to increase project documentation and reproducibility.
- `.dta` - "Stata Datasets" - These are the format that Stata stores tabular data in. We call these "D-T-A" files.
- `.smcl` - "Stata Log-files" - The default file format for Stata log-files is the `.smcl` file format, which is a varient of `html`. It is pronounced "smick-el". I recommend avoiding this file format whenever possible since only Stata can read it. Instead, save your log-files using the `.txt` file extension and the `text` option. The `.txt` file-type is a so-called "plain text" file format that can be read by an innumerable number of applications. This makes it excellent for reproducibility.

2.5.3 Other Files

We will also use a number of other types of files throughout the semester. Some may be file types that you have come across before.

- `.md` - "Markdown files" - These are plain text files that contain Markdown syntax (see page). They are saved with a special file extension so that software applications and web browsers know to take advantage of the embedded Markdown syntax.
- `.png` - "Portable Network Graphics" or "PNG files" - These are image files designed primarily for use on the Internet and on computer displays.
- `.txt` - "Plain text files" or "Text files" - These are files that contain text without any formatting (like bold or italicized text, for example). These can be opened by a wide array of text editor applications across all major operating systems.

2.5.4 Examples

Throughout the semester, I will give you examples both in lecture slides and in an example do-file. Examples in lectures and course documents can be easily identified by their use of the `typewriter typeface`:

```
. summarize mpg
    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+---------------------------------------------------------
```

```
        mpg |         74     21.2973    5.785503         12         41
```

Examples will almost always use the file `census.dta`, which comes pre-installed with Stata. To open it, use the `sysuse` command: `sysuse census.dta, clear`. This allows you to easily recreate examples by minimizing dependencies within do-files.

Chapter 3

"Good Enough" Research Practices

This section introduces some of the core concepts that we will emphasize in this course throughout the semester. The title takes inspiration from a recent article titled "Good Enough Practices in Scientific Computing". The premise of the article is twofold. The first premise is that scientific computing advice can sometimes be both overwhelming and focused on tools that are inaccessible to many analysts.

Chapter 4

Protecting Your Work

Each semester that I teach this course or SOC 5050 (Quantitative Analysis), two things happen. The first thing that happens is that students regularly lose files. The effects of losing files can range from being a minor frustration to a major headache depending on the file in question. Losing files often results in downloading multiple copies of the same data and recreating work. Both of these are wastes of your time. Moreover, files are rarely gone. They are typically just misplaced. This is bad for reproducibility, particularly when you happen across multiple versions of the same file and have to sort out which version is the version you last worked on.

The second thing that happens is that students lose their thumb drives. Depending on the timing of this loss, this can again range from being a minor frustration (very early in the semester) to being downright anxiety attack producing (last few weeks of the semester). Recreating an entire semester's worth of work on the final project is both a tremendous waste of your time and a particularly unpleasant experience.

Fortunately, I have never had a student's computer hard drive die during the course of the semester. However, I assume that if I teach this course long enough a hard drive failure will indeed occur. The backup provider Backblaze has analyzed their own hard drives and found that about 5% of drives fail within the first year. After four years, a quarter (25%) of drives in their data center fail.

Similarly, it is only a matter of time before a student's computer is stolen along with all of their hard work. A less likely though still very plausible scenario involves the destruction of a student's belongings (computer and thumb drive included) in a fire, car accident, flood, earthquake, tornado, hurricane, avalanche, or mudslide.

Despite the likelihood that you will at some-point lose a thumb drive (if not during this semester than sometime down the road) and the near certainty that your computer's hard drive will eventually fail if a rogue wave does not get it first, few students and faculty take these risks seriously. While you cannot prevent many of these things from happening, I want to suggest to you that you can take some simple steps to sure that *when* (not if) they happen, you are well prepared to get back to work with minimal disruption.

4.1 Creating a Sustainable File System

In his excellent document *The Plain Person's Guide to Plain Text Social Science*, Kieran Healy describes two important revolutions in computing that are currently taking place. One of them is the advent of mobile touch-screen devices, which he notes

> hide from the user both the workings of the operating system and (especially) the structure of the file system where items are stored and moved around.

For most users, I would argue that this extends to their laptop or desktop computers as well. I would venture to guess that the majority of my students are used to keeping large numbers of files on their desktops or in

an (distressingly) disorganized `Documents` folder.

For research, particularly quantitative research, such an approach to file management is unsustainable. It is difficult to produce *any* research, let alone work that is reproducible, without an active approach to file management.

4.1.1 Create a *Single* Course Directory

With one major exception (see the secontion below on GitHub), the most successful approach to organizing files is to identify *one and only one* area that you will store course files in. Having files scattered around you hard drive between you `Desktop` directory, `Downloads`, `Documents`, and a half dozen other places is a recipe for lost files. It can also add complexity to the task of backing these files up. I recommend naming this directory simply `SOC5050`. This is short, has no punctuation or spaces (which can create conflicts with software), and explicitly connects the directory to this course as opposed to other courses you may take that are also statistics courses (a good reason to avoid naming the directory `Statistics`!).

4.1.2 Approach Organizing Systematically

Within your single course directory, I recommend following much of Long's (2009) advice on organization. Approach this task systematically and mindfully. This approach begins with having a number of dedicated subfolders within your course directory:

```
/SOC5050
  /CoreDocuments
  /Data
  /FinalProject
  /GitHub
  /Notes
  /Posted
  /Readings
  /Working
```

Note again how these directories are named - there are no spaces, special characters, and the names are deliberately short but specific. For a directory with two words (`CoreDocuments` or `FinalProject`), I use what is known as camelCase to name the file where the second (any any subsequent) words have their first character capitalized. You could also use dash-case (`Core-Documents`) or snake_case (`Core_Documents`) as a naming strategy.

A `.zip` file containing an empty folder structure that mirrors what is described below will be posted to GitHub early in the semester.

4.1.2.1 The `CoreDocuments` Directory

This directory should be used to store *copies* of the core documents repository files that you sync from GitHub - the Syllabus, the Reading List, and the User's Guide. The files on GitHub may be updated (and therefore updated on your computer when you sync), so having local copies that are independent of GitHub can be helpful if you want to make sure you retain original files.

4.1.2.2 The `Data` Directory

The data directory should have copies of all original data and their documentation. I recommend renaming the files that you download from Dropbox for simplicity (most files come from the ICPSR data repository, which has a complex standard for file naming) and removing some files from the `CPS` and `NHIS` directories:

```
/SOC5050
  /Data
    /Acock
    /CPS
      /cpsCodebook.pdf
      /cpsDescription.pdf
      /cpsOriginalData.dta
      /cpsUserGuide.pdf
    /Long
    /NHIS
```

Note that the 2012 General Social Survey has been removed - I keep that in the `FinalProject` directory (see below).

4.1.2.3 The `FinalProject` Directory

The final project directory should be a microcosm of the larger directory structure, with most major directories replicated so that your final project files have a dedicated, organized home:

```
  /SOC5050
    /FinalProject
      /Data
        /gssCodebook.pdf
        /gssDescription.pdf
        /gssDocumentation.pdf
        /gssOriginalData.dta
        /gssQuestionarrie.pdf
        /gssRelatedLiterature.txt
        /gssReportQuickFacks.pdf
      /Directions
      /Notes
      /Posted
      /Readings
      /Working
```

I recommend keeping subdirectories within posted dedicated for different versions of your data analysis, paper, and presentation files.

I also recommend using some type of bibliography software (Endnote, for example, can be obtained for free by SLU students). Whatever application you choose, keep its primary database for your project in the `Readings` folder along with copies of all `.pdf` readings.

You will also be asked to create and maintain a research log for this project (see Long [2009]). I recommend keeping this and any other project notes in the `Notes` directory.

Details on the `Posted` and `Working` directories can be found below. You should replicate the practices that feed these directories for your final project.

4.1.2.4 The `GitHub` directory

This directory should contain the local copies of the repositories that you sync from GitHub's servers:

```
  /SOC5050
    /GitHub
      /Core-Documents
      /PrenerAssignments
```

```
/Week-01
/Week-02
...
/Week-15
```

You should clone and then sync the Core-Documents repository, your assignment repository, and all weekly repositories.

You will be in-charge of organizing the assignments repository. I recommend creating a subfolder for your final project deliverables, a subfolder for labs, and a subfolder for problem sets. Within those subfolders, create individual directories for assignments:

```
/SOC5050
  /GitHub
    /DoeAssignments
      /FinalProject
        /Drafts
        /Final
        /LiteratureReview
        /Memo
      /Labs
        /Lab01
        /Lab02
        ...
        /Lab16
      /ProblemSets
        /PS01
        /PS02
        ...
        /PS10
```

There are two important things to keep in mind about about your GitHub directory: 1. The first will only apply to students whose backup strategy involves using **Dropbox** or **Google Drive** to sync their course directory with a cloud storage service. If you do this, you **must** keep local copies of GitHub repositories outside of your course directory. GitHub uses software called Git for version tracking, and Git files (which are hidden from your view if you look at the directories in Apple's Finder or Windows Explorer) do not mix well with the version tracking software embedded in Dropbox and Google Drive. 2. Do not edit the files in these repositories - copy them to your working folder (or other relevant destination) and then edit versions. Editing files in the `Core-Documents` or weekly directories may create sync conflicts, and your assignments directory should be dedicated to *completed* and *posted* files that are required for submission.

4.1.2.5 The `Notes` Directory

Use this as a home for course notes and other resources.

4.1.2.6 The `Posted` Directory

I use this the way Long (2009) suggests - once a particular set of assignments, analyses, or writing is completed, I save it in a subfile within the `Posted` directory:

```
/SOC5050
  /Posted
    /Lab01
    /Lab02
    /PS01
```

These contain all of the files needed for an assignment and not just the deliverables requested for submission. Follow Long's (2009) advice - one files are saved in the `Posted` directory, do not edit them again. Copy any necessary data or other files out of the directory into the working directory, and edit them here. Once you are done, save them in a new subfolder within the `Posted` directory.

4.1.2.7 The `Readings` Directory

Use this as a home for `.pdf` copies of course readings.

4.1.2.8 The `Working` Directory

As with the `Posted` directory, I suggest following Long's (2009) advice and using this as a temporary holding place for files you are working on. Once they are done, move them out of the `Working` directory immediately and into a subfolder within the `Posted` directory.

4.2 Backing Up Your Data

There are a number of different ways to think about backing up your data. The most successful backup strategies will incorporate all of these elements.

4.2.1 Bootable Backups

"Bootable" backups are mirrored images of your *entire* hard drive, down to temporary files, icons, and system files. With a bootable backup, you can restore your entire computer in the event of a hard drive failure or a corruption of the operating system files. They are named as such because you can plug in the external drive that you are using for this backup and literally boot your computer up from that drive (typically a *very* slow process).

These backups are often made less frequently because they can be resource intensive and it is best not to use your operating system while creating a clone. They are typically made to an external hard drive, which is subject to similar failure rates as the hard drives inside your computer. So bootable drives need to be replaced every few years to maintain their reliability.

Both major operating systems come with applications for creating clones of your main hard drive that are bootable, and there are a number of third party applications that provide this service as well.

4.2.2 Incremental Backups

Incremental backups are designed to keep multiple copies of a single file (how often depends on the type of software you use and the settings you select). These can be used to restore an older copy of a file if work is lost or a newer file is corrupted.

Apple's TimeMachine is a great example of an incremental backup - when kept on, it creates hourly backups of files that have been changed, daily backups for the previous month, and weekly backups for previous months. Once the disk is full, the oldest backups are deleted. Dropbox also provides a similar service, retaining all previous versions of files (and deleted files) for thirty days.

Incremental backups are typically good options for recovering files that have been recently changed (again, depending on the software you use and the settings you select). Since they run frequently (every time a file is changed or every hour, for example), recent changes tend to get captured. They can be limited in terms of their long-term storage - it may not be possible to recover older versions of a file past a few weeks.

They are also not always good solutions for recreating your entire computer since they do not save all necessary program and operating system files, and may be cumbersome to work with if you need to recover a large quantity of files. Like bootable backups, these are typically stored on external hard drives that need to be replaced on a regular basis.

In addition to the aforementioned Apple TimeMachine, the Windows OS also comes with a built-in service for creating incremental backups. Dropbox is a good option if you have a small number of files, but you may find the need to upgrade to a paid account if you have a large amount of data.

4.2.3 Cloud Backups

Cloud backup services like Backblaze or Crashplan offer comprehensive backup solutions for customers. These plans typically require a monthly subscription fee to maintain access to your backups. While bootable backups protect against hard drive failure and incremental backups protect against data corruption, cloud backups protect against catastrophic events like robberies, fires, and other natural disasters. A fire or a tornado that affect your house may destroy your laptop and any external hard drives you use for backup, but your cloud backup will be unaffected.

4.2.4 A Workflow for Backups

Just as we need a workflow for approaching file management, it is also important to establish a routine for backups. With backups, the most successful workflows are those that require next to no effort on your part. If you primarily use a desktop, this can be as simple as leaving two external hard drives plugged into your computer since most backup software can be set to run automatically. If you have tasks that require you to manually do something (plug an external hard drive into your computer, for instance), create a reminder for yourself on a paper calendar or a digital calendar or to-do list application.

Chapter 5

Introduction to GitHub

Much of our interaction this semester outside of class will utilize GitHub.com (or just "GitHub"). GitHub is a web service that is a social network for programmers, developers, data scientists, researchers, and academics. It is also a tool for collaborating on projects, especially projects that involve writing code.

5.1 Git

GitHub is a web application that utilizes Git:

> Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Essentially, Git is a project-wide system for tracking changes to files. Think of it as Microsoft office's track changes feature on steroids - every change to every file in a directory (a "repository" or "repo" in Git-lingo) is tracked. You do no need to host files online to use Git. If you have a project saved locally (say, a doctoral thesis), you could utilize Git to version control that project without ever uploading it to the Internet.

For our purposes, this is just about all you need to know about Git. If you want to learn more, Git's 'About' page is a great place to start.

5.2 More Git-lingo

Beyond "repositories", there are a few additional terms that are specific to Git and that are helpful to know:

- **Clone**: Make an identical copy of a repository on your local hard drive.
- **Fork**:
- **Commit**: Approve any changes you have made to a repository.
- **Pull Request**:
- **Sync**: For cloned repositories, files that have been changed need to be pushed to GitHub.com after they are committed.

5.3 GitHub.com

GitHub is a web service that can host projects using Git's version tracking. It is widely used by programmers, software developers, data scientists, and academics to host and collaborate projects.

GitHub is an excellent way to backup files for a project since you can "sync" changes made to a repository up to GitHub's servers. It is also an excellent way to collaborate on files with colleagues while also using Git's version tracking. Repositories can be either public (like all of the repos for our seminar) or private, which means that only people who have been given access to can view the contents of the repo. Private repos require an upgraded account, which retails for $7/month.

Students can get access to GitHub's paid services for free, however, by signing up for a free student account. This will give you access to private repositories for as long as you are a student.

5.4 GitHub Repositories

Users of GitHub.com adhere to a couple of norms with their repositories that are worth knowing about. Repositories cannot have spaces in their names (much like variables in Stata), so the naming conventions that we will discuss in relation to Stata this semester all apply to GitHub as well!

Public GitHub repositories also contain (typically) at least three core files:

1. A **license** file - since the data is out there for public consumption, it is important to think about how that data is licensed. The norm among GitHub users has been to use open source licenses, which let others edit and adapt your work. There are a range of licenses that are commonly used on GitHub.
2. A **README** file - this describes the purpose and content of the project.
3. A **.gitignore** file - this stops certain types of files from being swept up by GitHub when a user syncs their files with a server.

Another norm is to write using a markup language known as Markdown. Markup languages allow users to specify exactly how they want their text to appear when it is parsed and processed by special software. This is different from, say, Microsoft Word, which is known as a "what you see is what you get" or **WYSIWYG** editor, which uses a graphical interface for constructing documents.

5.5 Storing GitHub Repositories

When you clone your repositories, you will be prompted to save them on your computer. There are a number of ways in which this process can introduce sources for trouble down the road:

1. External media - storing data on devices like thumb drives or external hard drives can be a part of a backup workflow. However, I have seen issues where this has appeared to contribute to sync errors with GitHub Desktop, particularly on Windows.
2. Cloud storage services (Dropbox, Google Drive, etc.) - like external drives, these services can be a part of a backup workflow. However, like external drives, I have seen issues whee this has to contribute to sync errors with GitHub Desktop.

In order to avoid any issues, I suggest storing GitHub repositories on your computer's hard drive and not a thumb drive or other external device. Make sure your are saving your files in a place not backed up to Dropbox or another cloud storage service.

5.6 GitHub Issues

GitHub has a powerful tool for interaction called Issues. These can be accessed by opening a repository and then clicking on the "Issues" tab. Issues can be "opened" by anyone with access to the repository. They allow for a conversation to occur in the form of messages posted within the Issue itself. Files can be attached

to Issues, and the messages can contain Markdown formatting. Once the conversation is complete, issues can be marked as "closed", which moves them into a secondary view on the website so that they are archived.

5.7 GitHub Desktop Application

GitHub Desktop is a tool that allows you to easily clone repositories hosted on GitHub, commit changes to them, and then sync those changes up to the website. You can also create new repositories, however this is not task you will have to do this semester. GitHub Desktop is not a fully functional desktop version of GitHub. For our purposes, it is important to note that the Desktop application will not let you easily identify when repositories have been updated by other users, view Wikis associated with repositories, or view Issues.

5.8 Learning More

GitHub has a resources page with links to websites that are great for helping you learn more about how Git and GitHub work!

Chapter 6

Final Words

We have finished a nice book.

Bibliography