Computer Software Setup

Generally the procedures for setting up your computer are common to all types of systems, but there are some steps that change depending on whether or not you have a Mac or PC. Please see the notes below for details. The full software setup for this class will take approximately **2-2.5 GB** of hard drive space.

The common tools we will be using are:

1. The Anaconda Python Distribution (Python 2.7)
2. The Anaconda R-Essentials Package (Installed from within Anaconda)
3. The Atom Code Editor (Open-Source from Github)

We will also be downloading materials from Github, so you will want to install the Github desktop application for your operating system:

4. Github Desktop

In addition, Windows users should install:

5. Gow (Some basic command-line utilities for Windows that make it look like UNIX)

Some of these tools may require Mac users to also install:

6. XCode

If you already have some of the tools, or prefer different ones, you may skip installing some of the things here and use your own, but keep in mind that the instructor may have **no idea whatsoever** how your tools work.

(1) Anaconda (Scientific Python 2.7 Distribution)

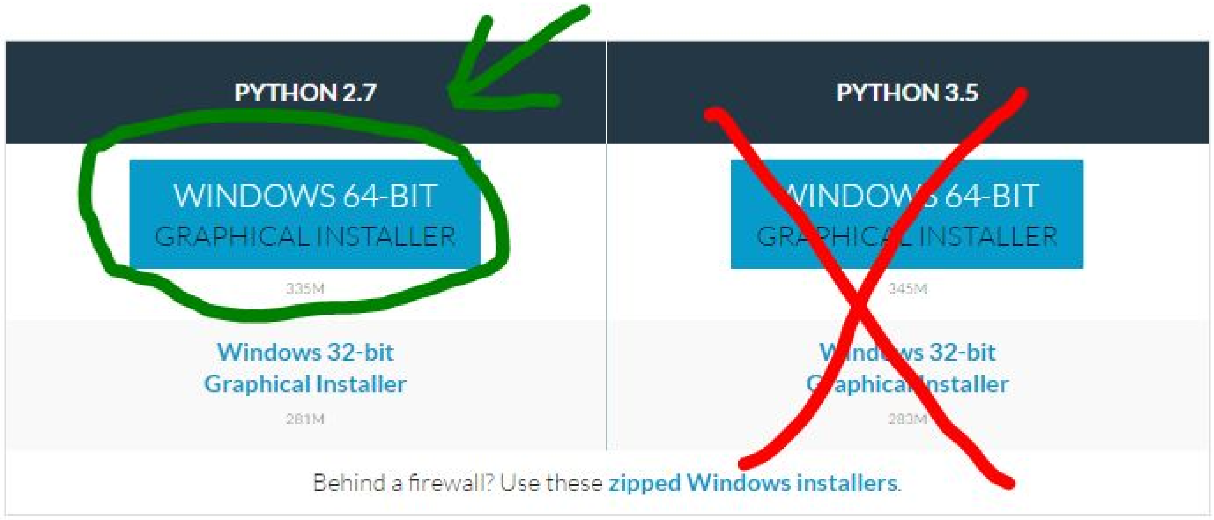
Anaconda is a powerful and fairly comprehensive collection of open-source scientific computing software based on Python. It is definitely way more than we need for this class, but it is easier to install the whole thing than to install the pieces we need bit by bit. It is released by Continuum Analytics (Austin, TX). The system is open and based entirely on free software, so there is no cost to download and install it. (If you are curious, the company makes money by selling support packages for Anaconda and also consulting services that build software on top of the Anaconda collection of open software.)

The software download is at:

https://www.continuum.io/downloads

Install the **64-bit, Python 2.7 Version** for most computers. Do not click any of the other versions shown. At the first class we will have a USB stick with copies of the software installers as well. Please see the important notes *below* for different system types before installing!

Here is the relevant screen for Windows (Mac should be similar):


PYTHON 2.7
WINDOWS 64-BIT GRAPHICAL INSTALLER
335M
Windows 32-bit Graphical Installer
281M
PYTHON 3.5
WINDOWS 64-BIT GRAPHICAL INSTALLER
345M
Windows 32-bit Graphical Installer
283M
Behind a firewall? Use these zipped Windows installers.

During the install you will be asked if you want to install for "all users" or the current user. (That latter should be the default and it is what you want.) Generally the defaults are what you should accept.

Notes by system type:

Important notes for NETBOOK users: If you are using a small, very lightweight netbook (the type of computer with 8+ hour battery life) for this class you **may** have a **32-bit** processor. You would need to pick a different installer, specifically *the one underneath the green circle* in the image above. The easiest way to be sure is to try the 64 bit version first, and if it fails switch to 32 bits. *Please be warned that installation on netbooks can take a very long time for Anaconda.*

Important notes for OS X (Mac) users: Installing Anaconda on Macs is simple enough, but for maximum system stability, you should install the system in such a way that it does **not** override your default system Python installation.

(Cont'd)

When the installer asks you to “change the system path” say **no**. If this option is not offered to you, don’t panic, nothing tragic of unfixable happens. If you get the option to not change the path, take it. Once you are done, you will have to add the following two lines to your **`.bashrc`** file (in your **home** directory):

```
alias ma='source ~/anaconda2/bin/activate root'
alias da='source deactivate'
```

This will add two new commands to your shell: ma and da. These activate Anaconda (**`ma`**, short for “mount anaconda”) and deactivate it (**`da`**, or dismount anaconda). Before using Anaconda you will start up a terminal app and from there issue the **`ma`** command. Then, in **that** terminal, all commands will use Anaconda as needed. (It will not affect other terminals you have open.) You will have to do this every time you work with Anaconda. **Contact the instructor if there are any problems.**

Important notes for Windows Users: Windows users do not need to deal with the problem above, as Windows does not come with a Python installation that it depends on for system function. **However, Anaconda may fail to change the relevant path variable for Windows users. If this happens, please contact the instructor.**

(2) Anaconda R-Essentials Package

Installing a minimal R system for this class is easy if you use Anaconda as the starting point. Open a command (terminal) window and, if necessary, **mount Anaconda as needed**. Once Anaconda is available, the command to install R and some supporting tools is:

```
conda install -c r r-essentials
```

If the installer gets to a point where it stops and asks to proceed, you may have to hit the enter key to make it continue. The “conda” program is the Anaconda package manager, a very powerful program that is designed to easily install a wide variety of software.

Note to people with R already installed: If you have R installed on your system already, this **should** not cause a conflict. (**I have not completely tested this claim!**) When Anaconda is mounted and in use, its version of R will override your previously installed copy. But when Anaconda is not in use, your copy should run. If you wanted to run both the Anaconda version and your other version simultaneously, you could open two terminals and mount Anaconda in only one of them. Any links you have directly to your previously installed R should work normally. Some of the R programs you need to write can be done in other software, but most of the examples we work with will be in **R Jupyter notebooks** (part of Anaconda) so you will need these items installed to follow along.

(3) Code Editing Program (Atom)

You will need a “code editor” or a “programmer’s editor” for working in this class. Basically this is a text editor that knows that writing code is different from word processing. Don’t use a word processor as they sneak all sorts of invisible signs and sigils into your documents, and that will mess up any programming you do!

If you have an editor that is “code aware,” that is the best option. These editors color-code and highlight the text using rules that vary by which computer language you are working with, and they know important things like how much to indent your code or to tell you when you forget to type a balanced pair of parentheses or brackets. There are dozens of these available and if you are using one already, feel free to stick with it. (One exception is any editor that is completely dedicated to one language such as **RStudio** or **Microsoft Visual Studio**. Both of these are fine tools, but both are dedicated to just one language or set of languages and we need something that works more broadly.)

The **official editor for this class** is **Atom** (https://atom.io/), an open-source, free, and fairly easy to use editor from the people who build Github. Download it from the link on the project’s homepage. (It should auto-detect your computer type.)

In addition to being a good editor, it has highlighting rules for pretty much any computer language out there, it knows the idiomatic choices for most languages, and it is tightly integrated with Git/Github when you are using that.

(4) Github Desktop

We will be using the Git and Github software development management systems for code. All of the code we do in this class will be publically available, and all the code you write in class will also be public. (So if you are shy about showing your mistakes to the world, get over that now.)

Git is a challenge to get started with, but if you manage to do it, it will pay big dividends down the road. Git provides version management -- it keeps track of your changes and lets you compare versions and go back to prior versions if you make a mistake -- and while we use it with computer codes, it can also be used with documents (like papers and citation databases) to keep track of your work as you write. Git can even be used to manage your collections of data as you work, allowing you to roll back changes you make if you decide to change things as you analyze data.

However, we will be focusing on **Github** in this class. *Git is a version management system, but Github is a public website that uses git to host projects.* This distinction is important. Git is very challenging to learn to use fully, but the basic functionality of Github can be learned in an hour or so.

One nice feature of Github is that they offer a graphical user interface to working with their site and managing projects on both their website and locally on your computer. If in the future you want to manage projects locally without mirroring a copy to the web, you will have to break down and learn to use git without the sleek user interface. (Although there are lots of third-party GUIs for that, too. But they all work differently, so we're not doing that here.)

To install Github's desktop app, go to: https://desktop.github.com/ and click on the "download GitHub desktop" button. As with much modern software, the site should recognize your computer type and suggest the right download.

(5) GOW (Gnu on Windows)

Gow stands for **Gnu On Windows**, and it is a (relatively) compact, 18 MB collection of approximately 100 Linux/UNIX/BSD utilities that are modified and built to run on Microsoft Windows from the command line. For instance, it will give you the ability to use many of the Mac/Linux commands you see used in class.

I have been using Gow on one of my windows systems for some time now, and it does appear to work as advertised. I would be concerned about doing "mission critical" work with it, but for use in class it should be fine.

Download Gow at the website: https://github.com/bmatzelle/gow/releases

There is currently a bug in Gow that stops the "bash" program from working fully, but the fix is easy: go to the Gow directory and add a new, empty, directory called "etc" (all lowercase). I have no idea why this fixes it, but it did for me, and it is a well-known fix in the online community.

(6) XCode (Mac Only)

I hope that it does not come up, because it is a huge ~2 GB download, but at some point is working with scientific programming I was forced to install Apple's XCode system to make some things work. (Keep in mind I use more of this stuff than we will be covering in class. So fingers crossed it does not come up for you!) If that happens to you, it is easy to install, just go to the

Apple App Store and grab XCode. It is free. Installation is automatic. It may require you to run a program at the command line to set some things, if this comes up and is a problem, contact the instructor.