Setting Up a Python Computing Environment

March 18, 2022

This document gives instructions to set up your computer with some helpful computing tools: scientific Python, the package and environment manager conda, Jupyter, and a text editor.

1 Terminal/command line

Try opening the Terminal (Mac or Linux) or Powershell (Windows) to see what it looks like. (Windows users, once you've installed Miniconda, below, this document will walk you through using using Anaconda Prompt for your command line tool instead of Powershell).

This setup document involves occasional use of the Terminal/command line, in particular the following commands:

`ls` (Mac/Linux) or `dir` (Windows) - Lists the contents of a directory.

`cd <directory_name>` - Change to another directory within the working directory. You use `cd` with a folder name, like `cd Documents`. If you want to go up one directory in the hierarchy, you can use `cd ..`

`pwd` (Mac/Linux) or `cd` (Windows) - Print the working directory (a "where am I?" sort of command).

There is a short tutorial that you can use for future reference on these and other commands here, but don't worry about it much now. All we want you to take away from this step is that the command line exists and you can use it to enter commands!

2 Install Miniconda (Python, conda, pip)

Miniconda is a minimal installer for conda, which is a package, environment, and dependency manager for Python and other languages. Installing conda will also get you the latest Python, as well as a few other useful tools like the standard Python package manager, `pip`.

Mac

- Download miniconda from: `https://docs.conda.io/en/latest/miniconda.html`
- In the Terminal, navigate to the Downloads folder (or wherever the Miniconda installer went). This will involve using `cd` to change directory until you are in the correct folder (e.g. `cd Documents`).
- Install miniconda by typing into the Terminal the command as directed here: `https://conda.io/projects/conda/en/latest/user-guide/install/macos.html`, that is:

  ```
  bash Miniconda3-latest-MacOSX-x86_64.sh
  ```

  or something analogous.
- Follow the instructions in the terminal during the Miniconda installation. They will probably ask you to confirm where it's being installed (probably something like `Users/karin/miniconda3`), and you can confirm that. When it asks you whether you want to intialize, you can confirm.
- Close the Terminal and open a new one. You should now see `(base)` at the start of the current line, indicating that you are in the base conda environment.

Linux

- Download miniconda from: https://docs.conda.io/en/latest/miniconda.html.
- Follow the directions here -just steps 1-6 under "Installing on Linux". Make sure you do step 6 and can open Anaconda Prompt.

Windows

- Download miniconda from: https://docs.conda.io/en/latest/miniconda.html.
- Follow the directions here -just steps 1-5 under "Installing on Windows". Make sure you do step 5 and can open Anaconda Prompt.

3 Get started with conda environments

In this step, you'll create an environment with conda. Eventually, you might have a few different environments set up, corresponding to different projects. Each environment can have different versions of Python and/or packages installed. This means you don't have to install some package everywhere if you just need it for one project. Using conda environments can help you avoid dependency issues when you have lots of packages down the line. You might not see the benefit right away, and that's fine. Getting this set up now will still be helpful later on.

- In the Terminal (or Anaconda Prompt on Windows), create a conda environment with a command like:

  ```
  conda create --name myenv
  ```

  Replace `myenv` with whatever name you like. For example, you might write:

  ```
  conda create --name data_science
  ```

  or

  ```
  conda create --name wearable_sensors
  ```

  When conda prompts you for `y/n`, enter `y` to proceed.

- Enter

  ```
  conda activate myenv
  ```

  substituting for `myenv` whatever environment name you chose previously (e.g.: `conda activate wearable_sensors`). Now you should see at the start of the line something like `(myenv)` or `(wearable_sensors)`, indicating you are no working in the environment you created, instead of the `(base)` conda environment. Now that you are working in the environment you just made, you can install packages here, for use only in this environment. In the next step, we will install a number of useful packages.

- Later, if you want to, you can deactivate the environment with :

  ```
  conda deactivate
  ```

  which will send you back to `base`. When you eventually want or need them, you can create new environments with different names, and activate the right one as you need it.

- Whenever you open a new Terminal, activate the environment you want to use with

  ```
  conda activate myenv
  ```

  replacing `myenv` with the name of your environment. Note: most of the instructions in the document are for one-time setups, but you will activate your conda environment every time you open a new terminal and get back to work on a project.

- If you want to see the names of all the environments you have already made, you can type `conda info --envs`.

- You can check which packages are installed in the active environment with `conda list`

- For more information about conda environments, look here. For a very helpful conda cheat sheet, look here.

4 Install some helpful Python libraries

- First, make sure you **have a conda environment activated**, as in the previous step. (If you see `(base)` at the start of the line in the Terminal/Anaconda prompt, you haven't yet activated your environment, and need to activate an environment before you proceed).

- We'll install some helpful libraries into the active environment. If you eventually add a new environment and want to use these packages in that one too, you will want to install them in that new environment. Use conda to install a bunch of helpful Python libraries for scientific computing by typing the following into the Terminal (or Anaconda Prompt if on Windows):

```
conda install numpy scipy scikit-learn pandas matplotlib
```

- NumPy is a scientific computing package that lots of other scientific Python libraries will use.

  Scipy is a library for mathematics.

  Scikit-learn is a machine learning library.

  Pandas is a helpful library for data preparation and moving data around.

  Matplotlib is a plotting library.

5 Get started in Jupyter

JupyterLab provides an interactive web interface or using with Jupyter notebooks that combine code and output with text, equations, and more. It's what Colab is based on, so it will feel familiar, except that now the code will be running locally on your own computer.

- First we'll install a few more libraries. Have a conda environment activated for all of the steps in this section. Enter the following

  ```
  conda install -c anaconda nb_conda
  ```

  ```
  conda install ipykernel
  ```

- Next, we'll install Jupyterlab.

  ```
  conda install -c conda-forge jupyterlab
  ```

- In the Terminal, navigate to a directory that corresponds to your project, and where you'd like to put your juptyer notebooks.

- Now that these tools are installed and we're in the right directory, we can run JupyterLab. Type `jupyter lab` into the terminal and it should open up a tab for JupyterLab in your default web browser (e.g. Chrome or Firefox). In this tab, create a new notebook (e.g. from the file menu), and try typing some valid Python, e.g. `1 + 1`, into the first cell in the notebook, then enter or press the forward arrow run button to run it and see the output.

6 Practice exercise for using pip, jupyter and conda together

Seaborn is a useful Python library for plotting. We're going to install it to practice the process of installation. Make sure you have a conda environment activated (not base!), and open a jupyter notebook if you don't already have one open. Write the following line of code in the jupyter notebook and try to run it:`import seaborn as sns`. You should get an error like "No module named 'seaborn'".

Go back to the Terminal. (or Anaconda Prompt). You can stop running the notebook with ctrl-C. You should still have a conda environment activated (don't be in base). We're going to use the tool `pip` to install seaborn into our conda environment so that we can use it! Execute the command `pip install seaborn` and the installation will begin. (You may have to confirm with a y/n. Also, if you get a message about pip not being found, you may have to run `conda install pip` before `pip install seaborn` will work.)

Once the installation finishes, relaunch your jupyter notebook (by typing `jupyter lab`), and try again to run `import seaborn as sns`. It should work this time, because now you have seaborn installed in the conda environment!

Note: You can use conda can manage environments AND install packages. Pip is not for managing environments, but it can install Python packages. This partial overlap can be a little confusing, but it is good to know how to use both conda and pip for installation, as well as how to use conda for managing an environments.

Because both pip and conda can install Python packages, you sometimes get to choose which one to use. The makers of a package may give you instructions using one or the other. It is good to know that pip can only install Python packages, while conda can install software in other languages. For that reason, I generally prefer to install with conda if I can, because it will better handle non-Python dependencies.

7 A text editor

- Download and install the text editor VSCode from https://code.visualstudio.com/. Or if you have another text editor you prefer, use that (Atom and Sublime are popular alternatives).

Once you have VSCode installed, open it and save a new file with a name that has no spaces in the name and that ends in `.py`., e.g. `hello.py`. Within `hello.py` you can put one or more lines of Python code, e.g.:

```
print("hello, everyone!")
```

In the Terminal, you can navigate to the directory that contains the file you just saved and execute the Python code that is in the file with the command `python hello.py` (Use your own filename instead of `hello.py` if applicable.)

8 Celebrate!

These installation and setup steps can be confusing and finicky. Good job working through them!!

9 Epilogue: each subsequent time you want to run Python using conda and Jupyter...

Open the Terminal or Anaconda prompt and do the following:

1. Use `cd` to navigate to the directory in which you want to work (think: where do I want to save my work?).

2. Activate the conda environment that you want to work in with

   `conda activate myenv` (substituting your chosen environment name for `myenv`). You've already installed a good set of libraries in this environment, but you can use conda or pip add more libraries if you later find other ones you need.

3. Launch Jupyter with `jupyter lab`.

Note: You can do steps 1 and 2 in either order.
Note: Skip step 3 if you don't want to use Jupyter.