

Anaconda - Installation Instructions

Last Updated: October 18, 2020

NOTICE: I encountered issues with interactive plots in Jupyter after installing the latest Anaconda version 2020.07 on a Windows 10 machine per this document (and I do not know if this effects other platforms). I narrowed the issue being with the version of matplotlib installed which is easily resolved after installation and detailed in the troubleshooting appendix. Be sure to run the notebook for the first class after installation per Section 4 and if the plots only partially show refer to the resolution in the appendix.

Overview

The following is a procedure for installing Anaconda which includes Python and Jupyter Notebooks used to demonstrate course material. Allow yourself at least two hours to complete this procedure.

Please ensure the following is installed and running properly prior to the first class. The notebooks require Python 3.7 or higher to be installed. To ensure the notebooks provided will run seamlessly, please install Anaconda according to these instructions where possible, and contact me via email at boschen@loglin.com with any issues encountered as we will not have time during class to deal with installation issues. The following steps were done on a Windows platform, but Anaconda can be installed on Windows, MacOS or Linux. If there are differences encountered when installing on Linux or MacOS that would benefit others, or if you have any other comments, questions or corrections, please e-mail them to me as well.

If you currently have a version of Python installed and don't want to interfere with your current set-up, read the Appendix on Environments before proceeding if you want to continue to use what is already installed for outside of the workshop on your machine. If you have an earlier version of Anaconda, I recommend updating to the latest version in a new environment rather than updating the base environment to ensure nothing breaks in the process. This is detailed in the appendix.

Otherwise if your current version is not needed, completely uninstall the current Python installation and reboot your machine prior to proceeding.

Step 1: Download Anaconda Distribution

Go to <https://www.anaconda.com/products/individual> and click on “Download” to download the latest version of the Anaconda Individual Edition (When this was first written Python 3.7 was available so most of the graphics mention that, but select the most current Python 3.x version, NOT Python 2.x which is no longer offered through Anaconda). Choose Windows, MacOS or Linux based on your operating system of choice. After selecting the correct operating system, click on the large green “**Download**” button under the Python 3.x version after selecting either the 64 bit installer or 32 bit installer based on your processor (If you aren’t sure which processor you have, on Windows you can right-click on the Start Icon in the lower left hand corner, select Open Windows Explorer, then right-click on Computer and select Properties where the Processor type will be listed under “System”. If you accidentally make the wrong choice, the installer will detect and notify you at the start of the installation).

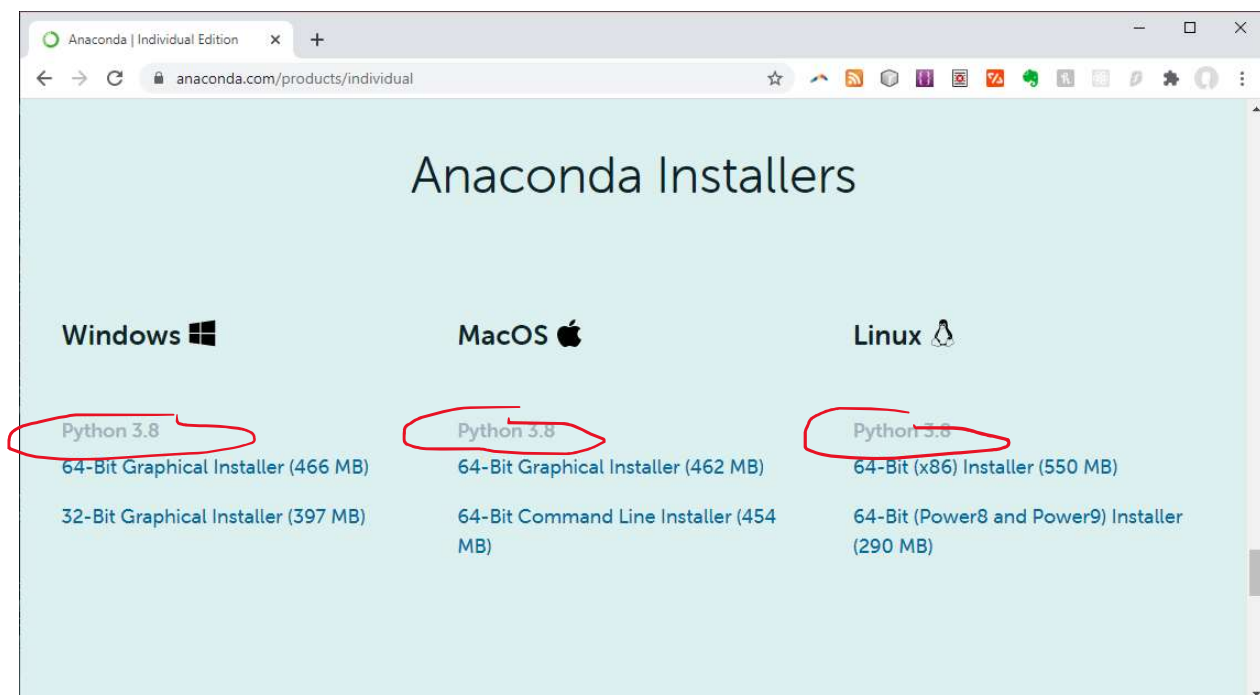
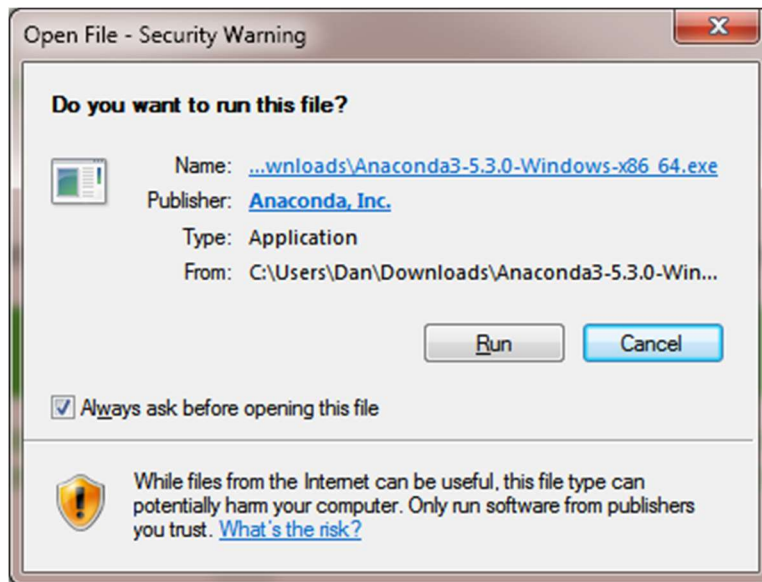


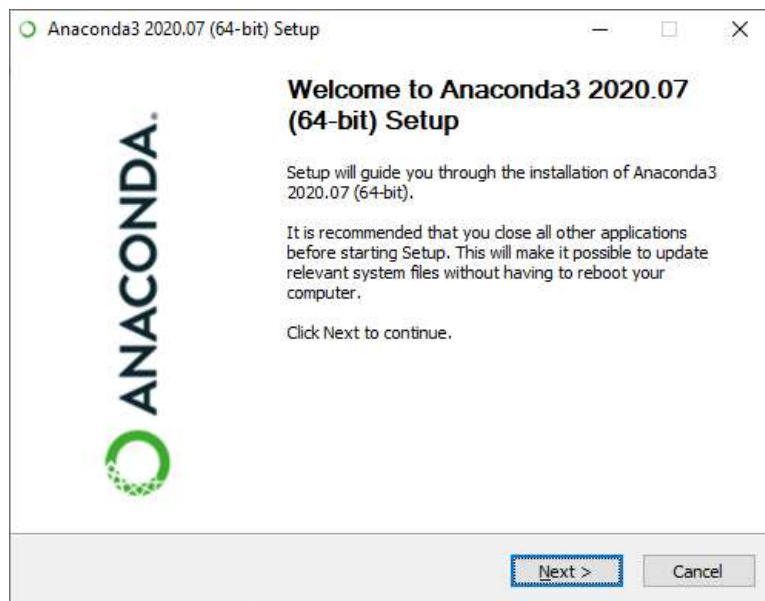
Figure 1: Select Python 3.x for installation for your operating system

STEP 2: Install

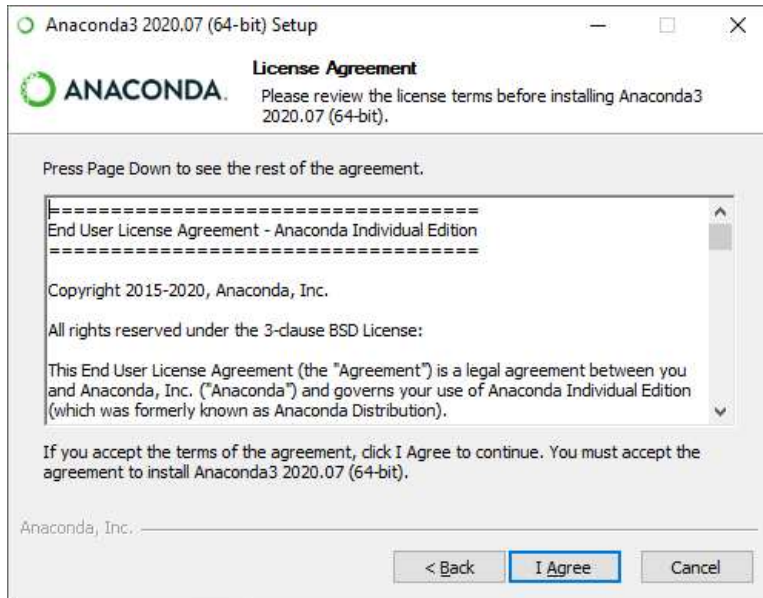
Click on the downloaded executable and click on Run to begin installation. If you get a security warning prompt such as that shown below, press Run to proceed.



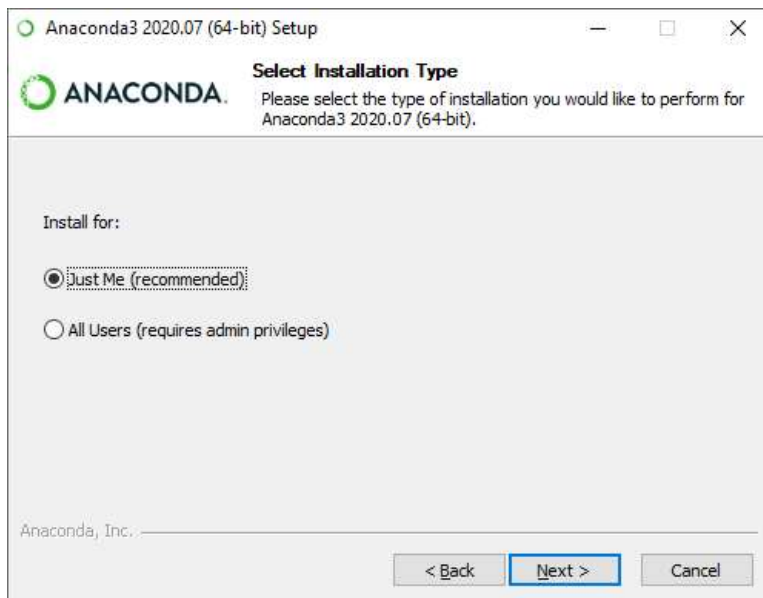
Click on "Next" to start the setup:



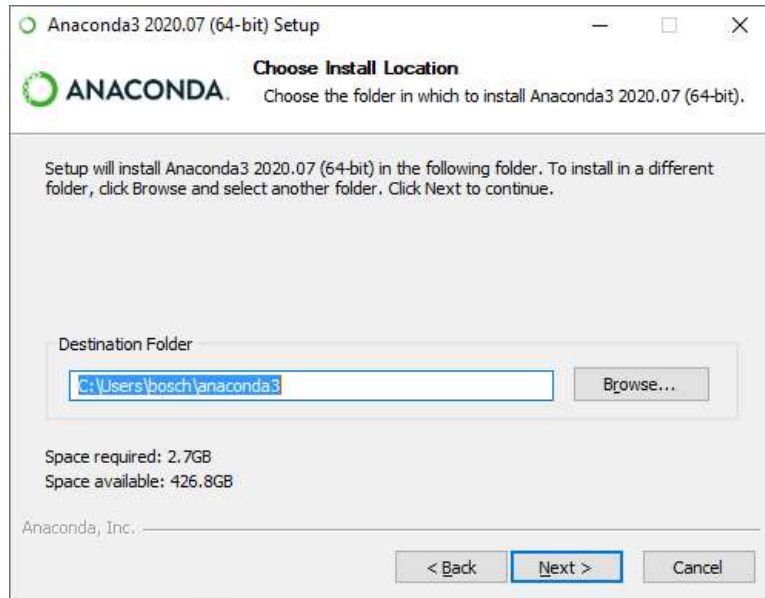
Click on **"I Agree"** to accept the License Agreement:



Select **"Just me"** for the Installation Type and click on **"Next"**.

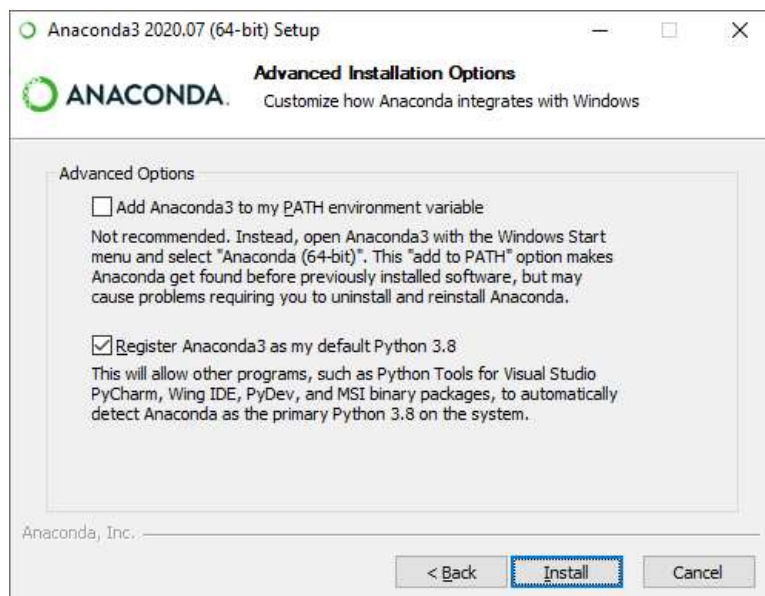


Click on “**Next**” for the install folder location. Ensure that it does not contain spaces.

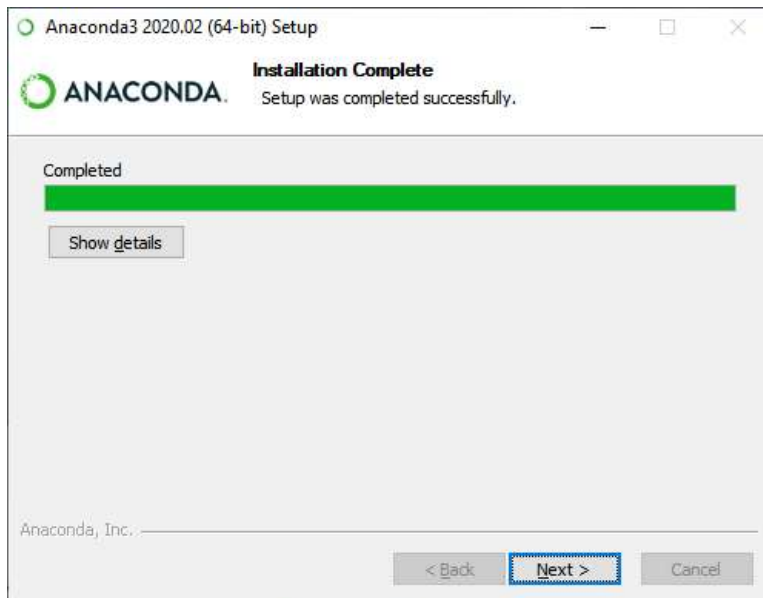


Remember this directory location as the location that is directly above it (D:\Users\Dan in this case) will be the default directory that is opened, and where we will later create a “Workspace” directory for all the files for the course.

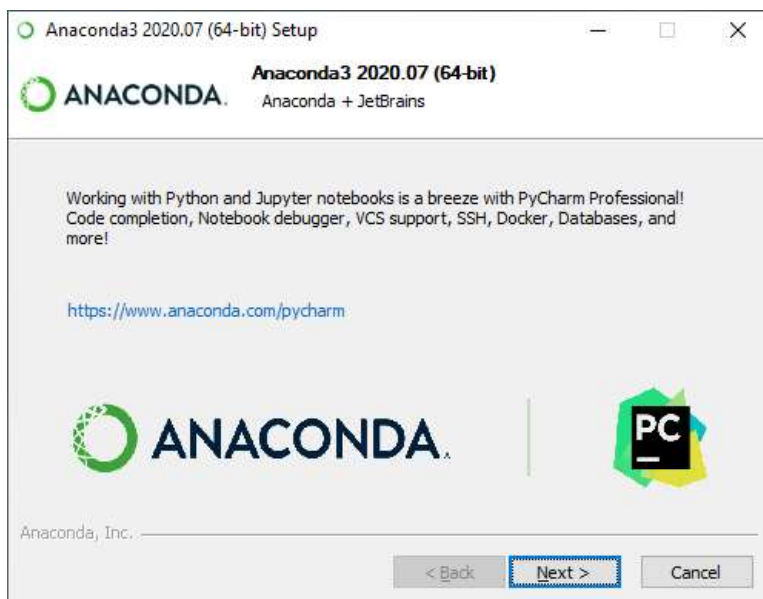
Ensure “Add Anconda to my Path environment variable” is **not** selected and “Register Anaconda as my default Python 3.x” is selected and click “Install”.



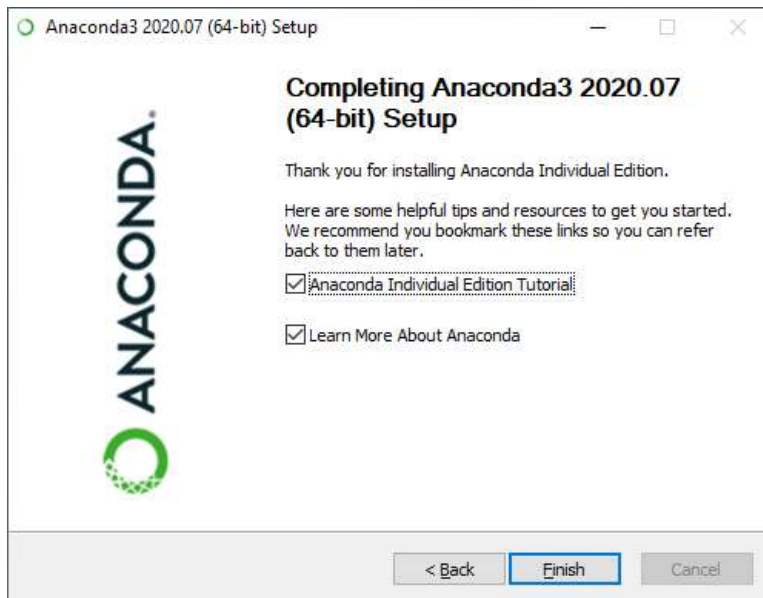
This process may take a long time. When you reach this next window- success! Anaconda is installed. Click on “**Next**”.



We will not be using PyCharm, press Next:



The installation is complete. You can optionally select the following items to learn more about each. Press **“Finish”** to end the installation process:



Step 3: Install Chrome

If you do not already have Chrome installed, it is recommended that you download it and use it as your default browser for the class (other browsers may be fine, but versions of Internet Explorer in particular will not support certain features of Jupyter Notebook that we will use, and using Chrome will provide a consistent experience for the class).

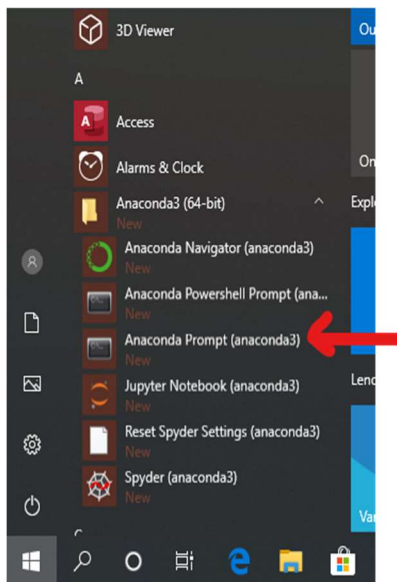
<https://www.google.com/chrome/browser/>

STEP 4: Confirm Installation and Install Plug-Ins

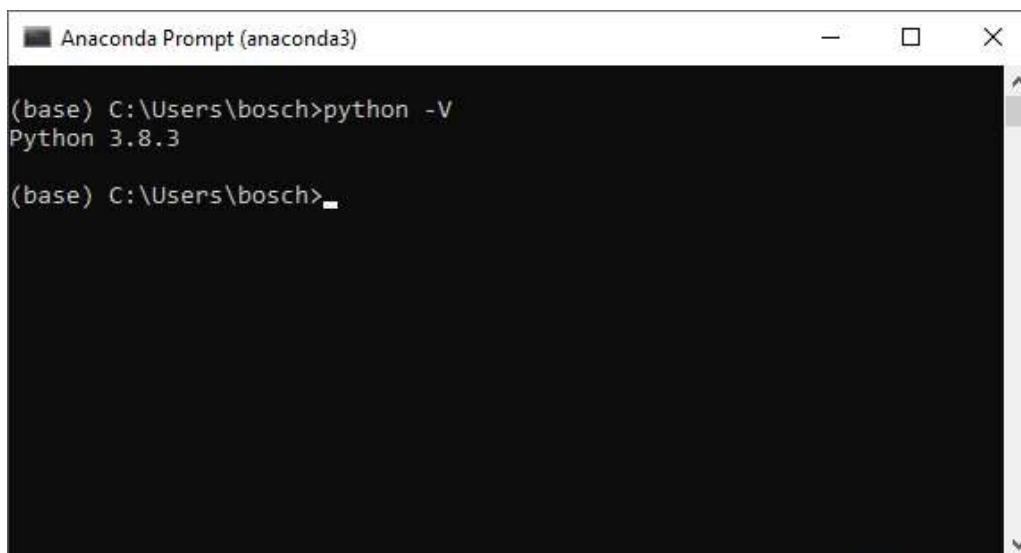
Perform the following steps to confirm everything is ready. If there are any issues, please email Dan at boschen@loglin.com at least two days before the start of the class so that we can try to resolve them. We will not have time during class to resolve installation issues.

Navigate to the install folder and run “Anaconda Prompt”

This can be launched on Windows from Start-All Programs-Anaconda3 -Anaconda Prompt. (On a macOS and Linux open a standard console and type `conda activate base`). (This may also appear as “Anaconda Powershell Prompt”).



From the Anaconda Prompt console, type `python -V` and press enter. Confirm the response with the Python version as shown in the screen capture below:



nbextensions: Install a collection of community-contributed extensions that add useful functionality to the Jupyter notebook by typing the following in the Anaconda Prompt console:

```
conda install -c conda-forge jupyter_contrib_nbextensions
```

After a few minutes something like the following will appear in the console. Ignore warnings about newer versions of conda and enter y to proceed.

```

Anaconda Prompt - conda install -c conda-forge jupyter_contrib_nbextensions

(base) C:\Users\Dan>conda install -c conda-forge jupyter_contrib_nbextensions
Collecting package metadata (current_repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
  current version: 4.8.2
  latest version: 4.8.3

Please update conda by running

  $ conda update -n base -c defaults conda

## Package Plan ##

  environment location: D:\Users\Dan\anaconda3

  added / updated specs:
    - jupyter_contrib_nbextensions

The following packages will be downloaded:

  package | build | size | channel
  -----|-----|-----|-----
  ca-certificates-2020.1.1 | 0 | 165 KB | anaconda
  certifi-2019.11.28 | py37_0 | 157 KB | anaconda
  conda-4.8.3 | py37hc8dfbb8_1 | 3.1 MB | conda-forge
  jupyter_contrib_core-0.3.3 | py_2 | 15 KB | conda-forge
  jupyter_contrib_nbextensions-0.5.1 | py37_0 | 19.5 MB | conda-forge
  jupyter_highlight_selected_word-0.2.0 | py37_1000 | 13 KB | conda-forge
  jupyter_latex_envs-1.4.4 | py37_1000 | 753 KB | conda-forge
  jupyter_nbextensions_configurator-0.4.1 | py37_0 | 487 KB | conda-forge
  openssl-1.1.1d | he774522_4 | 5.7 MB | anaconda
  python_abi-3.7 | 1_cp37m | 4 KB | conda-forge
  -----|-----|-----|-----
  Total: | | 29.8 MB |

The following NEW packages will be INSTALLED:

  jupyter_contrib_c~ conda-forge/noarch::jupyter_contrib_core-0.3.3-py_2
  jupyter_contrib_n~ conda-forge/win-64::jupyter_contrib_nbextensions-0.5.1-py37_0
  jupyter_highlight~ conda-forge/win-64::jupyter_highlight_selected_word-0.2.0-py37_1000
  jupyter_latex_envs conda-forge/win-64::jupyter_latex_envs-1.4.4-py37_1000
  jupyter_nbextensi~ conda-forge/win-64::jupyter_nbextensions_configurator-0.4.1-py37_0
  python_abi conda-forge/win-64::python_abi-3.7-1_cp37m

The following packages will be UPDATED:

  conda pkgs/main::conda-4.8.2-py37_0 --> conda-forge::conda-4.8.3-py37hc8dfbb8_1

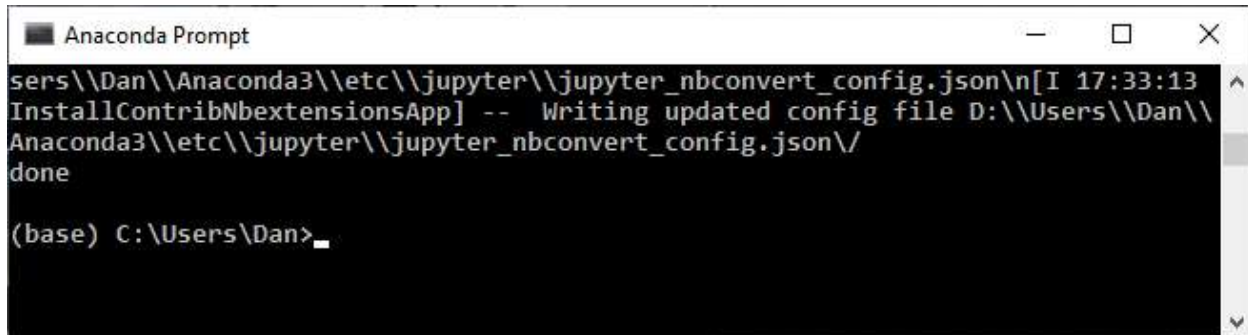
The following packages will be SUPERSEDED by a higher-priority channel:

  ca-certificates pkgs/main --> anaconda
  certifi pkgs/main --> anaconda
  openssl pkgs/main --> anaconda

Proceed ([y]/n)?

```

When successfully completed after entering “y”, a simple “done” should be the last item displayed.



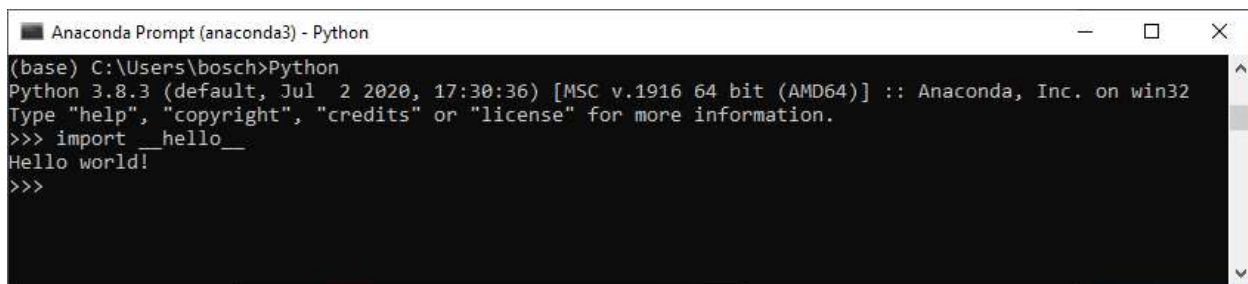
```

Anaconda Prompt
sers\\Dan\\Anaconda3\\etc\\jupyter\\jupyter_nbconvert_config.json\n[I 17:33:13
InstallContribNbextensionsApp] -- Writing updated config file D:\\Users\\Dan\\
Anaconda3\\etc\\jupyter\\jupyter_nbconvert_config.json\
done

(base) C:\Users\Dan>

```

To confirm installation and operation, the following are some of the Easter Eggs to try that are buried in Python. Start python by typing `python` in the Anaconda Prompt then from the Python command prompt, type `import __hello__`, (note there are two underscores in a row before and after hello) and confirm that you get the “Hello world!” response as shown below.



```

Anaconda Prompt (anaconda3) - Python
(base) C:\Users\bosch>Python
Python 3.8.3 (default, Jul 2 2020, 17:30:36) [MSC v.1916 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import __hello__
Hello world!
>>>

```

Congratulations if that was your first Python program.

Next, type `import this`

- The Zen of Python, a poem by Tim Peters should print in the console.

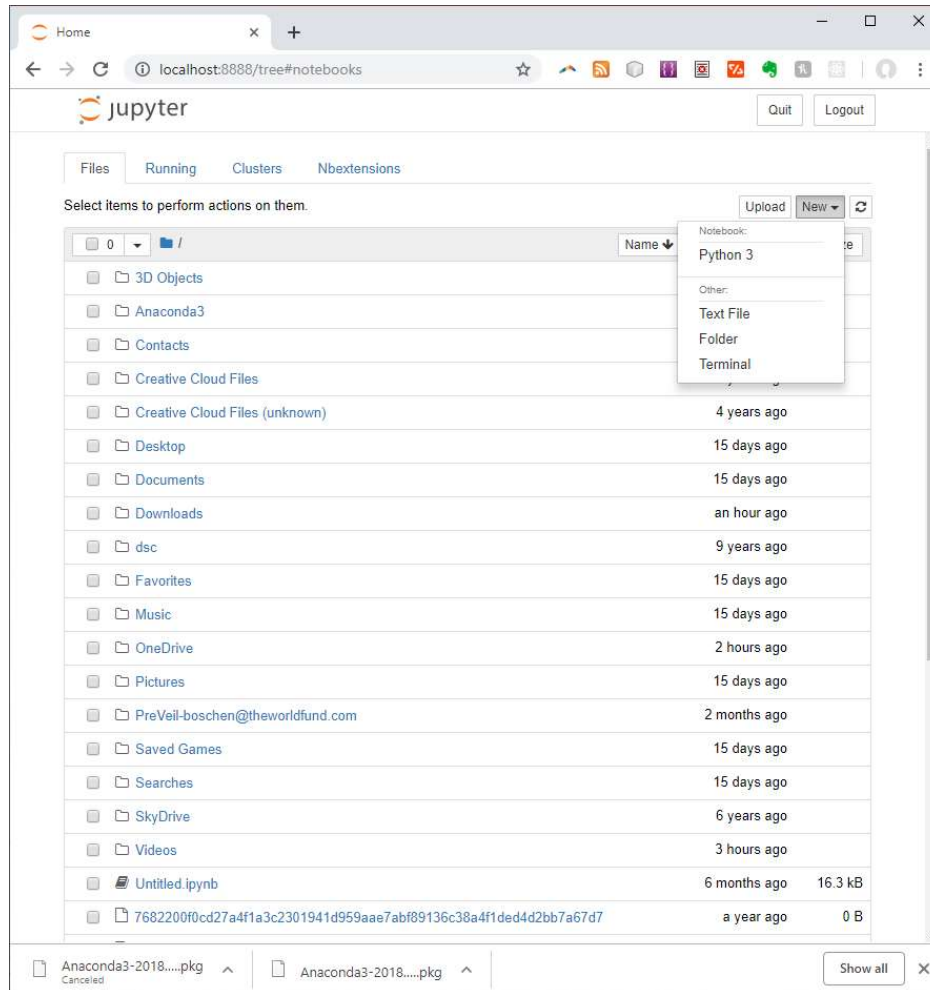
Next, type `import antigravity`

- A browser should open displaying an xkcd comic about Python

Confirm that the Jupyter Notebook and the `jupyter_contrib_nbextensions` are properly installed by running Jupyter Notebook either from the Jupyter Console (by first typing `exit()` if still at the Python command prompt, and then typing `jupyter notebook`, or from the same location that the Anaconda Prompt was run (On Windows this can be launched from Start-All Programs-Anaconda3 - Jupyter Notebook). If given an option of what browser to use, choose Chrome. If successful, a browser window similar to the screenshot below will appear with a file listing that depends on whatever was in the root directory chosen for installation. A console window will also appear which can be minimized but must not be closed or the Jupyter server now running in the background will stop.

Create a directory titled “workspace” where we will keep the class notebook files. This can be done through the main Jupyter page by clicking on New – Folder as in the screen shot below. (Or through your

regular file explorer).



All files distributed from the course will be available in the same shared folder where this document was obtained, and the files from there that are below the “workspace” directory should be placed below the same “workspace” directory on your own machine in the same structure similar to what is shown below, (we will be using directory names “module_x” instead of “Class_x”. The shared folder containing the files will be updated throughout the course, so please be sure to download the latest files for each session from the shared folder prior to start of each session.

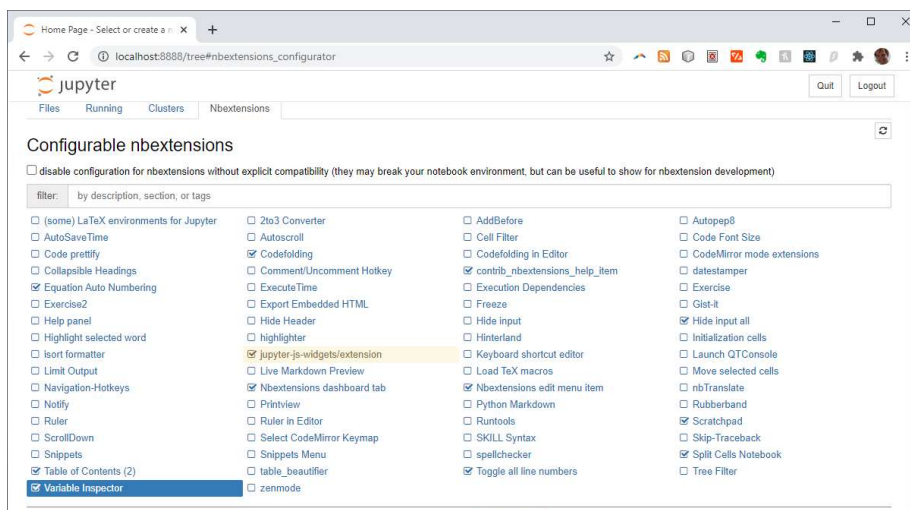
Workspace

- Class1
- Class2
- Class3
- Class4
- Class5
- styles
- tools

After launching Jupyter, from the main Jupyter page the tab “Nbextensions” should now be available across the top. Click on it to see all configurable extensions. Uncheck the option to “disable configuration...” shown at the top and then check off the following items to be consistent with the extension I have been using:

- Code Folding
- contrib_nbextensions_help_item
- Equation Auto Numbering
- Hide Input all
- Jupyter-js-widgets/extension
- Nbextensions dashboard tab
- Nbextensions edit menu item
- Scratchpad
- Split Cells Notebook
- Toggle all line numbers
- Table of Contents (2)
- Variable Inspector

Below is a screen-shot showing these selections. Clicking on any one item will bring up a help screen at the bottom with additional information on how to use that extension.



TESTING THE NOTEBOOK: Navigate to the workspace/module1 folder from the main Jupyter page under Files and click on TestNotebook.pynb to open it. Click on each cell in turn from the start of the notebook and press the run button in the top menu bar to execute each cell. Importantly confirm that the final plot displays completely as explained on the cover page of this document (it will be very obvious if there is an issue).

Once complete, close the notebook by selecting File – Close and Halt from the top level menu in the open Notebook to quit the Notebook that was open, and then from the main Jupyter page click on the Quit button in the upper right-hand corner to close the Jupyter Notebook server and then the browser window can be closed normally. The console window may also be closed at this time.

This completes the installation and test of Anaconda sufficient for the class. If anything did not work as expected, or if you have any other feedback from the installation that would be helpful to others, please e-mail boschen@loglin.com with your comments.

Appendix A: Interoperating with an existing Python installation

If you have Python installed and need to continue using it (for example, your coworkers are using Python 2.7), this is done using Conda Environments to avoid any conflicts. I also highly recommend working in an environment to test out major updates rather than updating the base environment except for individual package updates. Please read this entire section before proceeding to first understand the possible options to doing this.

If Anaconda is not already installed: Create a file that will list the current libraries used (and version).

From the command prompt in a terminal window enter:

```
pip freeze > legacy_env.txt
```

Then uninstall everything related to the current Python installation (except your own source code files) so that we can run everything without conflict within the conda environment management system.

If Conda is already installed, it would make more sense to install the latest Anaconda in a new environment as further detailed below, but if want to replicate the existing installation in a new environment the same process above can be done from the Anaconda Prompt in Windows (or terminal in macOS or linux) by entering the following Conda command (this can also be utilized as a roll-back option).

```
conda list --export > legacy_env.txt
```

Proceed with the instructions from the beginning of this document to install Anaconda.

Once Anaconda is installed, all of the following commands listed are done from the Anaconda Prompt console. Create a virtual environment for the legacy Python as follows: (please be patient as this may take a while)

```
conda update conda
conda create -n <environment name> --file legacy_env.txt`
```

Where <environment name> is replaced with your desired name for the environment. You may need to add the full path to the file name legacy_env.txt or move the file to the directory location where the conda command is being entered.

The above will install all the packages that were previously installed. Optionally you can just install a bare Python version and then from that new environment install the packages needed. For example, entering the following installs Python version 2.7 in an environment names py27:

```
conda create -n py27 python=2.7
```

Once an environment is created, you can activate the new environment using:

```
conda activate py27
```

(Replacing py27 with whatever environment name is used.) To deactivate type the following:

```
conda deactivate
```


To see a list of environments, use:

```
conda info --envs
```

The environment name that is currently active will be indicated in the prompt itself from the console.

To update Anaconda in a new environment:

Create the new environment as detailed above with any environment name of choice (such as py38 below)

```
conda update conda
conda create -n py38
```

Activate the environment

```
conda activate py38
```

Install the latest version of Anaconda

```
conda install anaconda
```

Or, alternatively install only the minimum packages needed for this course by following the instructions to create an ieeex environment as detailed further in Appendix D: Minimum Installation.

You may get a `PackagesNotFoundError` if any of the packages from `legacy_env.txt` are not located within the channels loaded for conda. The available channels for a package can be found by going to <https://anaconda.org> and type the package name in the “Search Anaconda Cloud” search bar at the top of the page. From this you can locate available channels for the missing packages. Add the channels to your conda channel list by entering from the Anaconda prompt:

```
conda config --append channels <channel>
```

Which adds the channel to the bottom of the list of accessed channels. With this added, reenter the command that resulted in the error. Optionally you can install a package from a channel directly if using the conda install command:

```
conda install -c <channel> <package name>
```

If duplicating an existing environment using the process with the `legacy_env.txt` file described above, and the packages listed as not being found are not actually needed / used by you, then you can optionally make a copy of `legacy_env.txt` with those removed, and repeat

```
conda create -n <environment name> --file legacy_env_copy.txt
```

If you encounter any further challenges with the procedure above, please e-mail Dan at boschen@loglin.com for further support in the installation process for multiple environments.

Appendix B: Troubleshooting

The following lists possible issues and solutions.

Plotting Issue with Interactive Plots in Jupyter Notebook

After updating to Anaconda 2020.07, the interactive plots in Jupyter Notebook were no longer completely displaying (only the upper left-hand corner of the plot would appear). This may be specific to a Windows 10 OS which I am using, but the solution was to revert back from Matplotlib version 3.30 to 3.1.3 by doing the following:

Open the Anaconda prompt (or terminal in macOS or Linux), and type the following (after “activate <environment>” if using any other environment besides base):

```
conda install matplotlib=3.1.3
```

DLL Load failed related to scipy.signal

This should only occur if you had a previous version of Python installed, or may occur after installing the control package that is used in the course. If when running the notebook files you encounter an import error after running the import block, starting as captured below and ending with “ImportError: DLL Load failed: The specified procedure could not be found”:

```
In [2]: # packages used

import numpy as np
import scipy.signal as sig
import matplotlib.pyplot as plt
import sys
sys.path.append("../")

import tools.fftplot as fftplot
from numpy.random import randn

-----
ImportError                                Traceback (most recent call last)
<ipython-input-2-ab02231fad6d> in <module>
      2
      3 import numpy as np
----> 4 import scipy.signal as sig
      5 import matplotlib.pyplot as plt
      6 import sys
```

This solution worked for me: <https://stackoverflow.com/questions/55201924/scikit-learn-dll-load-failed-in-anaconda>

Specifically disable any of the following listed DLL's that are in the Windows/system32 directory by changing the suffix from .dll to .old as they are conflicting with the Anaconda installation:

mkf_core.dll, mkf_def.dll, mkf_intel_thread.dll, libiomp5md.dll, libmmd.dll

Unhandled Exception in event loop:

If the following error occurs in Ipython through the Anaconda Prompt on a Windows machine:

```
Unhandled exception in event loop:

File "C:\Users\bosch\anaconda3\lib\asyncio\proactor_events.py", line
768, in _loop_self_reading
    f.result() # may raise
File "C:\Users\bosch\anaconda3\lib\asyncio\windows_events.py", line
808, in _poll
    value = callback(transferred, key, ov)
File "C:\Users\bosch\anaconda3\lib\asyncio\windows_events.py", line
457, in finish_recv
    raise ConnectionResetError(*exc.args)

Exception [WinError 995] The I/O operation has been aborted because of
either a thread exit or an application request
Press ENTER to continue...
```

The solution according to <https://github.com/ipython/ipython/issues/12049> is to downgrad the prompt toolkit to 2.0.10 using conda by typing the following from the anaconda prompt:

```
conda install -c esrf-bcu prompt_toolkit=2.0.10
```

(if package not found, search on anaconda.org for alternate channel that has 2.0.10 version and replace with esrf-bcu above)

Appendix C: File Sharing

This section is not specific to Anaconda but contains debugging details related to the mechanism used to distribute course files (current Dropbox).

Unlocking all files recursively on macOS

The files once copied over from Dropbox may need to be unlocked. To do this efficiently, open the terminal and enter

```
chflags -R nouchg /usr/bin/workspace
```

Where `/usr/bin/workspace` is replaced with the actual top-level directory of all files to unlock

Appendix D: Minimum Installation

The following lists the minimum packages needed for the course if creating a new environment named `ieee`:

```
conda create --name ieee python
conda activate ieee
conda install numpy, scipy, matplotlib, ipython
conda install spyder, jupyter, notebook
conda install jupyter_contrib_nbextensions
```

(use `-c` option to specify a channel above if any packages are not found

```
conda install -c conda-forge jupyter_contrib_nbextensions
```

Once created, export a text file of the completed environment to easily recreate:

```
conda list -explicit > ieee-env.txt
```

To recreate environment from text file:

```
conda env create -file ieee-env.txt
```

To clone environment (for testing new packages for example):

```
conda create --clone ieee --name test
```

(There may be more that need to be added to this list above.... if you get a package not found error while running the notebooks, open the Anaconda prompt (or terminal in macOS and linux), activate the `ieee` environment and enter the line `conda install <package name>` where `<package name>` refers to the missing package.