Installation Guide

This guide provides a brief overview of the steps required to set up Python on your machine and implement code efficiently for your problem sets. Of course, there are many ways to set up your Python, and you might already have found your perfect way.

1. Download a Python distribution with Anaconda

First, you need a Python distribution. You can download and then install the Anaconda Python Distribution from https://www.anaconda.com/download/success.

2. Set up an environment and install packages

Background:

In contrast to MATLAB or Stata, Python is a programming language developed to execute any computational task. Therefore, the basic Python distribution has only a few functionalities, and to use it for our purposes, we need to install packages, i.e., specialized libraries for each task. The most common packages for our purposes are:

- numpy (for numeric computations)
- scipy (for scientific numeric methods, e.g., optimization)
- statsmodels (for regressions)
- linearmodels (for regressions)
- pandas (for data-related tasks)
- matplotlib (for plotting)
- Jupyter (for Jupyter notebooks)

As packages are constantly evolving and new versions are released, it is good practice to set up a so-called environment for a project and only install the packages you need there for the project. This ensures the stability of your project code.

In the next steps, we create an empty environment and fill it with the packages listed above. If you need additional packages, simply customize the following steps to install them into your environment.

Steps:

Below are two methods for creating an empty environment and installing the packages. The first uses only the shell. If you plan to work with any kind of programming in your research (or future career), I can highly recommend getting familiar with working in a shell:

- Initialize conda in your shell after the installation of the Anaconda Navigator
    - Windows: Either use the Anaconda Prompt as your shell or the native Windows PowerShell. Many people find PowerShell more convenient to use. However, it comes with additional setup costs. How to initialize conda is explained on this Stack Overflow
    - Mac + Linux: Should work out of the box
- Create an empty environment(replace env-name with own choice):
  `conda create --name env-name`
- Activate conda environment:
  `conda activate env-name`
- With an activated conda environment installation command directly install a package into the environment. Try installing numpy with:
  `conda install numpy`

- Install the other packages listed above

The second way is using the interface of the Anaconda Navigator:

- First, create an empty environment. It is explained here, how to do this.
- Install packages into the environment. It is explained here, how to do this.

3. Visual Studio Code (or other IDEs)

With an installed environment, we are ready to execute code. You can develop code in a simple text editor, but there are many better ways to do so. The most widely used integrated development environment (IDE) is Visual Studio Code, developed by Microsoft.

IDEs not only look nicer than your basic text editor but also have a lot of nice extensions, e.g., code formatters, debuggers, or GitHub co-pilot (by the way, GitHub has a free premium education account, which allows you to use co-pilot).

After installing your IDE, open your project folder and start creating scripts for the code. Always open your project folder, not just individual files. You can also set the Python environment created before as the Python distribution you use to execute your code.

Useful resources:

Introduction to Python - Beginner:

- Quantecon has a good introduction to Python:
    - https://python-programming.quantecon.org/intro.html
- https://www.datacamp.com/ has great courses for starting with Python - It should be free with your university email. However, I have also created a classroom that you can join by writing me(mblesch@diw.de) an email. Here are some example courses, but there are more than you could ever do:
    - https://www.datacamp.com/courses/introduction-to-numpy
    - https://www.datacamp.com/courses/intro-to-python-for-data-science
- Of course, there are also many open-access books, YouTube tutorials, and much more

Advanced resources:

- Economic Model packages on https://github.com/OpenSourceEconomics
- Lectures on quantecon: https://quantecon.org/lectures/
- EconAark tool package https://econ-ark.org/
- Introduction to Jax for HPC computing YouTube tutorial

Of course, the most helpful resource is the problems of other programmers on Stack Overflow. Given that Python is (one of) the most used programming languages, ChatGPT has learned to write excellent Python code.

Not necessary, but a recommended tool: Git

Install:

- https://git-scm.com/book/en/v2/Getting-Started-Installing-Git
- https://git-scm.com/downloads

How to use git:

- https://www.youtube.com/watch?v=USjZcfj8yxE
- https://docs.github.com/en/get-started/quickstart/git-and-github-learning-resources