

A Gallery of Interesting IPython Notebooks

This ever-growing list of notebooks, powered by nbviewer, shows the depth and breadth of numerical analysis you can do with IPython. It includes everything from short examples and tutorials to full-blown courses and books composed in the notebook format!

Video tutorials

Searching the Internet, you will find many video-recorded tutorials on IPython. I'd especially recommend seeking tutorials from the PyCon, SciPy, and PyData conferences by Fernando Perez and Brian Granger, two of the primary creators and maintainers of IPython and Jupyter.

Books

Python for Data Analysis

Wes McKinney's book includes a chapter that covers using IPython as a data scientist. Although much of the material overlaps what we've discussed here, another perspective is always helpful.

Learning IPython for Interactive Computing and Data Visualization

This short book by Cyrille Rossant offers a good introduction to using IPython for data analysis.

IPython Interactive Computing and Visualization Cookbook

Also by Cyrille Rossant, this book is a longer and more advanced treatment of using IPython for data science. Despite its name, it's not just about IPython—it also goes into some depth on a broad range of data science topics.

Finally, a reminder that you can find help on your own: IPython's ?-based help functionality (discussed in “[Help and Documentation in IPython](#)” on page 3) can be very useful if you use it well and use it often. As you go through the examples here and elsewhere, you can use it to familiarize yourself with all the tools that IPython has to offer.

Introduction to NumPy

This chapter, along with [Chapter 3](#), outlines techniques for effectively loading, storing, and manipulating in-memory data in Python. The topic is very broad: datasets can come from a wide range of sources and a wide range of formats, including collections of documents, collections of images, collections of sound clips, collections of numerical measurements, or nearly anything else. Despite this apparent heterogeneity, it will help us to think of all data fundamentally as arrays of numbers.

For example, images—particularly digital images—can be thought of as simply two-dimensional arrays of numbers representing pixel brightness across the area. Sound clips can be thought of as one-dimensional arrays of intensity versus time. Text can be converted in various ways into numerical representations, perhaps binary digits representing the frequency of certain words or pairs of words. No matter what the data are, the first step in making them analyzable will be to transform them into arrays of numbers. (We will discuss some specific examples of this process later in [“Feature Engineering” on page 375](#).)

For this reason, efficient storage and manipulation of numerical arrays is absolutely fundamental to the process of doing data science. We’ll now take a look at the specialized tools that Python has for handling such numerical arrays: the NumPy package and the Pandas package (discussed in [Chapter 3](#).)

This chapter will cover NumPy in detail. NumPy (short for *Numerical Python*) provides an efficient interface to store and operate on dense data buffers. In some ways, NumPy arrays are like Python’s built-in `list` type, but NumPy arrays provide much more efficient storage and data operations as the arrays grow larger in size. NumPy arrays form the core of nearly the entire ecosystem of data science tools in Python, so time spent learning to use NumPy effectively will be valuable no matter what aspect of data science interests you.