Numerical Python

by Robert Johansson

2015

# 1. Introduction to Computing with Python

This book is about using Python for numerical computing. Python is a high-level, general-purpose interpreted programming language that is widely used in scientific computing and engineering. As a general-purpose language, Python was not specifically designed for numerical computing, but many of its characteristics make it well suited for this task. First and foremost, Python is well known for its clean and easy-to-read code syntax. Good code readability improves maintainability, which in general results in less bugs and better applications overall, but it also encourages rapid code development. This readability and expressiveness is essential in exploratory and interactive computing, which requires fast turnaround for testing various ideas and models.



# 2. Vectors, Matrices, and Multidimensional Arrays

# 3. Symbolic Computing

Symbolic computing is an entirely different paradigm in computing compared to the numerical array-based computing introduced in the previous chapter. In symbolic computing software, also known as computer algebra systems (CASs), representations of mathematical objects and expressions are manipulated and transformed analytically. In the scientific Python environment, the main library for symbolic computing is SymPy (Symbolic Python). SymPy is entirely written in Python, and provides tools for a wide range of analytical and symbolic problems.

## Importing SymPy

The SymPy project provides the Python module named sympy. It is common to import all symbols from this module when working with SymPy, using from sympy import \*.

# 4. Plotting and Visualization

Unlike most Python libraries, Matplotlib actually provides multiple entry points into the library, with different application programming interfaces (APIs). Specifically, it provides a stateful API and an object-oriented API, both provided by the module matplotlib.pyplot. I strongly recommend only using the object-oriented approach, and the remainder of this chapter will solely focus on this part of Matplotlib.

A graph in Matplotlib is structured in terms of a Figure instance and one or more Axes instances within the figure. The Figure instance provides a canvas area for drawing, and the Axes instances provide coordinate systems that are assigned to fixed regions of the total figure canvas; see the below Figure:

