# 1-01 Reading and plotting stock data

## Print rows in data frame

import pandas as pd

def test\_run():

"""Function called by Test Run."""

df = pd.read\_csv("data/AAPL.csv")

# TODO: Print last 5 rows of the data frame

print(df[-5:])

if \_\_name\_\_ == "\_\_main\_\_":

test\_run()

## Compute Max closing price

Graphical user interface, text, application, email

Description automatically generated

## Compute mean volume

"""Compute mean volume"""

import pandas as pd

def get\_mean\_volume(symbol):

"""Return the mean volume for stock indicated by symbol.

Note: Data for a stock is stored in file: data/<symbol>.csv

"""

df = pd.read\_csv("data/{}.csv".format(symbol)) # read in data

# TODO: Compute and return the mean volume for this stock

return df['Volume'].mean()

def test\_run():

"""Function called by Test Run."""

for symbol in ['AAPL', 'IBM']:

print "Mean Volume"

print symbol, get\_mean\_volume(symbol)

if \_\_name\_\_ == "\_\_main\_\_":

test\_run()

## Plot stock price data

Graphical user interface, text, application, email

Description automatically generated

## Plot high prices for IBM

"""Plot High prices for IBM"""

import pandas as pd

import matplotlib.pyplot as plt

def test\_run():

df = pd.read\_csv("data/IBM.csv")

# TODO: Your code here

# print df['High']

df['High'].plot()

plt.show() # must be called to show plots

if \_\_name\_\_ == "\_\_main\_\_":

test\_run()

## Plot two columns

Graphical user interface, text, application, email

Description automatically generated

# 1-02 Working with multiple stocks

## Lesson summary

To read multiple stocks into a single dataframe, you need to:

* Specify a set of dates using pandas.date\_range
* Create an empty dataframe with dates as index
* This helps align stock data and orders it by trading date
* Read in a reference stock (here SPY) and drop non-trading days using pandas.DataFrame.dropna
* Incrementally join dataframes using pandas.DataFrame.join

Once you have multiple stocks, you can:

* Select a subset of stocks by ticker symbols
* Slice by row (dates) and column (symbols)
* Plot multiple stocks at once (still using pandas.DataFrame.plot)
* Carry out arithmetic operations across stocks, e.g. normalize by the first day's price

## Build a dataframe in pandas

A close-up of a pen

Description automatically generated with medium confidence

## Join SPY data

Text

Description automatically generated

## Read in more stocks

Text

Description automatically generated

## Slicing Data frame

Text

Description automatically generated

**Use df.loc is more popular than df.ix**

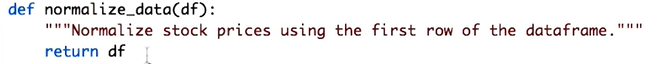
## Plotting a dataframe

Text

Description automatically generated with medium confidence

## Normalizing a dataframe

Text, letter

Description automatically generated

# 1-03 The Power of Numpy

## Lesson outline

* Create a NumPy array:
  + from a pandas dataframe: pandas.DataFrame.values
  + from a Python sequence: numpy.array
  + with constant initial values: numpy.ones, numpy.zeros
  + with random values: numpy.random
* Access array attributes: shape, ndim, size, dtype
* Compute statistics: sum, min, max, mean
* Carry out arithmetic operations: add, subtract, multiply, divide
* Measure execution time: time.time, profile
* Manipulate array elements: Using simple indices and slices, integer arrays, boolean arrays

## Notes on notation

Diagram

Description automatically generated

## Create NumPy arrays

**ND arrays: N dimensional arrays**

**Documentation:**

* [**pandas.DataFrame.values**](http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.values.html)**: Underlying values as ndarray**
* [**numpy.array**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.array.html)**: Create a NumPy ndarray from given sequence**
* [**numpy.ndarray**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.ndarray.html)**: NumPy n-dimensional array type**

Graphical user interface, text

Description automatically generatedGraphical user interface, text

Description automatically generated

Text

Description automatically generatedGraphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

**Documentation: [numpy.ones](http://docs.scipy.org/doc/numpy/reference/generated/numpy.ones.html" \t "_blank)**

* [**numpy.ones**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.ones.html)
* [**Data types**](http://docs.scipy.org/doc/numpy/user/basics.types.html)

**NumPy User Guide:**[**Data types**](http://docs.scipy.org/doc/numpy/user/basics.types.html)

**Documentation:**

* [**numpy.empty**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.empty.html)
* [**numpy.ones**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.ones.html)
* [**numpy.zeros**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.zeros.html)
* [**numpy.array**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.array.html)
* [**numpy.ndarray**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.ndarray.html)**(direct ndarray constructor)**

## Generate random numbers

Text, letter

Description automatically generated

**Sampling functions:**

* [**numpy.random.random**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.random.random.html)**: Samples a Uniform distribution in [0.0, 1.0)**
* [**numpy.random.rand**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.random.rand.html)**: Like random, but slightly different syntax**
* [**numpy.random.normal**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.random.normal.html)**: Normal or Gaussian distribution**
* [**numpy.random.randint**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.random.randint.html)**: Integers from Uniform distribution**

## Get the array attributes

**Attributes of [numpy.ndarray](http://docs.scipy.org/doc/numpy/reference/generated/numpy.ndarray.html" \t "_blank):**

* [**numpy.ndarray.shape**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.ndarray.shape.html)**: Dimensions (height, width, ...)**
* [**numpy.ndarray.ndim**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.ndarray.ndim.html)**: No. of dimensions = len(shape)**
* [**numpy.ndarray.size**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.ndarray.size.html)**: Total number of elements**
* [**numpy.ndarray.dtype**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.ndarray.dtype.html)**: Datatype**

## Operations of ndarrays

**NumPy Reference:**[**Mathematical functions**](http://docs.scipy.org/doc/numpy/reference/routines.math.html)

* [**numpy.sum**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.sum.html)**: Sum of elements - along rows, columns or all**
* [**numpy.min**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.min.html)**, [numpy.max](http://docs.scipy.org/doc/numpy/reference/generated/numpy.max.html" \t "_blank), [numpy.mean](http://docs.scipy.org/doc/numpy/reference/generated/numpy.mean.html" \t "_blank): Simple statistics**

**Also: [numpy.random.seed](http://docs.scipy.org/doc/numpy/reference/generated/numpy.random.seed.html" \t "_blank) to (re)set the random number generator.**

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, letter, email

Description automatically generated

A picture containing text

Description automatically generated

## Timing python operations

**Documentation:**

* [**time.time**](https://docs.python.org/2/library/time.html#time.time)**: Time in seconds, as a floating-point number**
* [**timeit**](https://docs.python.org/2/library/timeit.html)**: Average execution time measurement**
* [**profile**](https://docs.python.org/2/library/profile.html)**: Code profiling**

**iPython "**[**magics**](https://ipython.org/ipython-doc/dev/interactive/tutorial.html#magic-functions)**":**

* [**%time**](https://ipython.org/ipython-doc/dev/interactive/magics.html#magic-time)**: How long does it take to run once**
* [**%timeit**](https://ipython.org/ipython-doc/dev/interactive/magics.html#magic-timeit)**: Averaged over multiple runs**
* [**%prun**](https://ipython.org/ipython-doc/dev/interactive/magics.html#magic-prun)**/**[**%lprun**](https://ipython.org/ipython-doc/dev/interactive/magics.html#magic-lprun)**: Per-function/line profiling**

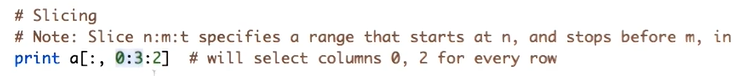
Graphical user interface, text, application, email

Description automatically generated

Text

Description automatically generated

## Slicing arrays



## Indexing NumPy

**NumPy Reference:**[**Indexing**](http://docs.scipy.org/doc/numpy/reference/arrays.indexing.html)

* [**Integer array indexing**](http://docs.scipy.org/doc/numpy/reference/arrays.indexing.html#integer-array-indexing)
* [**Boolean array indexing**](http://docs.scipy.org/doc/numpy/reference/arrays.indexing.html#boolean-array-indexing)

## Arithmetic operations

**NumPy Reference:**[**Arithmetic operations**](http://docs.scipy.org/doc/numpy/reference/routines.math.html#arithmetic-operations)

* [**numpy.add**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.add.html)**: Element-wise addition, same as + operator**
* [**numpy.subtract**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.subtract.html)**: Element-wise subtraction, same as -**
* [**numpy.multiply**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.multiply.html)**: Element-wise multiplication, same as \***
* [**numpy.divide**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.divide.html)**: Element-wise division, same as /**
* [**numpy.dot**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.dot.html)**: Dot product (1D arrays), matrix multiplication (2D)**

**Note: Arrays need to be *compatible* with each other for these operations to work (see:**[**Broadcasting**](http://docs.scipy.org/doc/numpy/user/basics.broadcasting.html)**).**

**For more matrix operations, see:**[**Linear algebra**](http://docs.scipy.org/doc/numpy/reference/routines.linalg.html)**and the**[**matrix**](http://docs.scipy.org/doc/numpy/reference/generated/numpy.matrix.html)**class.**

# 1-04 Statistical analysis of time series

## Lesson outline

Pandas makes it very convenient to compute various statistics on a dataframe:

* Global statistics: [mean](http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.mean.html), [median](http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.median.html), [std](http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.std.html), [sum](http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.sum.html), etc. [[more](http://pandas.pydata.org/pandas-docs/stable/api.html#api-dataframe-stats)]
* Rolling statistics: [rolling\_mean](http://pandas.pydata.org/pandas-docs/stable/generated/pandas.rolling_mean.html" \t "_blank), [rolling\_std](http://pandas.pydata.org/pandas-docs/stable/generated/pandas.rolling_std.html" \t "_blank), etc. [[more](http://pandas.pydata.org/pandas-docs/stable/computation.html?highlight=rolling%20statistics#moving-rolling-statistics-moments)]

You will use these functions to analyze stock movement over time.

Specifically, you will compute:

* Bollinger Bands: A way of quantifying how far stock price has deviated from some norm.
* Daily returns: Day-to-day change in stock price.

Rolling Statistics