

# Tutorial: Market Simulator

# Outline

1. Install Python and some libraries
2. Download Template File
3. Do MC1-P1 together
  - <http://quantsoftware.gatech.edu/MC1-Project-1>
  - Edit the analysis.py file
4. Watch Videos (Udacity)
5. Do MC2-P1 on your own:  
<http://quantsoftware.gatech.edu/MC1-Project-1>  
Which is a Market Simulator (it will use the functions from analysis.py). **[Project 4] which is more like a homework**

# Installation:

Step 1: Install your python platform

a): Install Anaconda

Step 2: Install Market Simulator Templates, Project 4 is Part 1 below.

Part 1: Read. <http://quantsoftware.gatech.edu/MC2-Project-1>

It needs SciPy — so:

Note: The **Anaconda python distribution** includes

\* NumPy, Pandas, SciPy, Matplotlib, and Python,  
and over 250 more packages available via a  
simple “conda install <packagename>”

It also has an IDE.

Instructor got 2.7, and the anaconda distribution of python

To get the appropriate software you'll need:

python (scripting language 1301)

sci.py (numerical routines),

num.py (matrices, linear algebra), and

matplotlib (enables generating plots of data)

Installing Python (2.7) and OpenCV (3.1): (you only need do install (3)

1) OpenCV Site (reference only – don't need this for this project).

<http://opencv.org/>

2) [SciPy.org site, and launched to installation notes:](http://scipy.org/install.html)

<http://scipy.org/install.html>

3) **Anaconda instruction site** including lots of libraries with python.

<https://docs.continuum.io/anaconda/install>

Mac Installation:

1) Instruction that the instructor used:

a) installed anaconda (got required packages)

[https://www.continuum.io/downloads\(2.7\)](https://www.continuum.io/downloads(2.7))

includes, sci.py, num.py, and matplotlib

# Videos

- Sign up on Udacity (Free):
  - <https://classroom.udacity.com/courses/ud501/lessons/3909458794/concepts/42693317700923#>
- **Create The Analysis Tool Relevant Videos**
  - 01-01 – 20 minutes
  - 01-02 – 30 minutes      Pandas/Frames/Slices
  - 01-04 – 23 minutes      Daily Returns/Cumulative Returns
  - 01-07 – 22 minutes      Sharp Ratio (Statistics)
- **Market Simulator**
  - 02-02 – 27 minutes      Market Mechanics
  - <https://www.youtube.com/watch?v=TstVUVbu-Tk> (long)

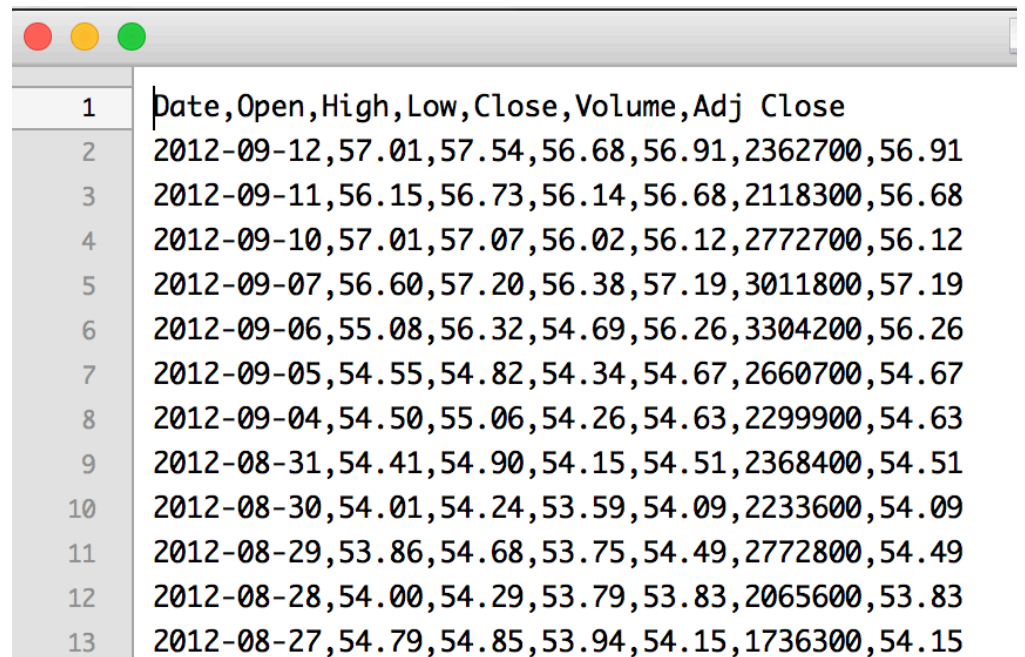
- Read Stock Data from a CSV File and input it into a pandas DataFrame
  - Pandas.DataFrame
  - Pandas.read\_csv
- Select desired rows and columns
  - Indexing and slicing data
  - Gotchas: Label-based slicing convention
- Visual data by generating plots
  - Plotting
  - Pandas.DataFrame.Plot
  - Matplotlib.pyplot.plot

- Scrape S&P 500 ticker list and industry sectors from list of S&P 500 companies on Wikipedia.
  - [https://en.wikipedia.org/wiki/List\\_of\\_S%26P\\_500\\_companies](https://en.wikipedia.org/wiki/List_of_S%26P_500_companies)
- Download daily close data for each industry sector from Yahoo finance
  - using pandas DataReader.
- Adjust the open, high and low data using the ratio of the adjusted close to close.

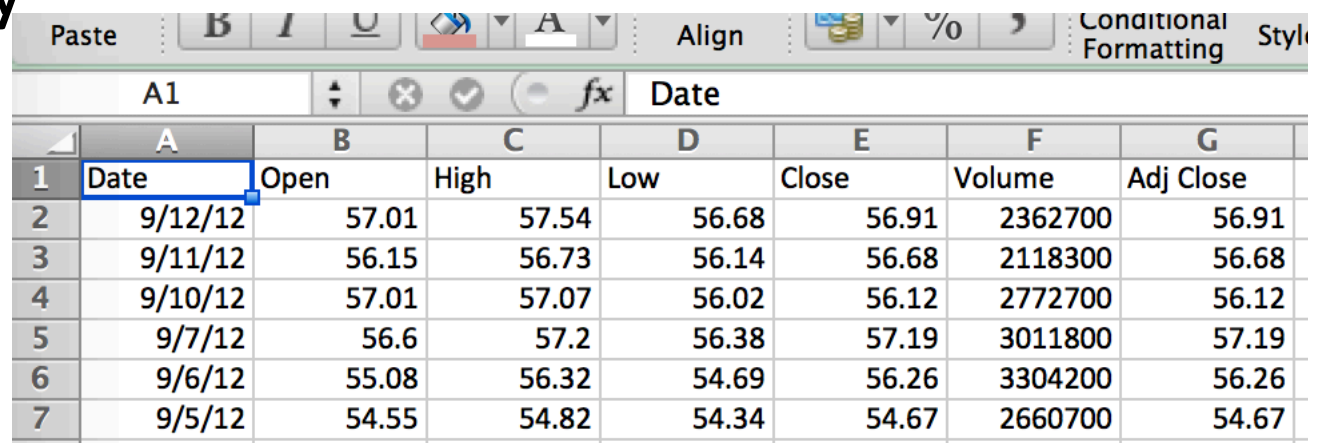


# Comma Separated Values (.CSV)

- CSV File
- Header Files
- Lines/Rows of Dates
- Each Element is separated by columns



```
1 Date,Open,High,Low,Close,Volume,Adj Close
2 2012-09-12,57.01,57.54,56.68,56.91,2362700,56.91
3 2012-09-11,56.15,56.73,56.14,56.68,2118300,56.68
4 2012-09-10,57.01,57.07,56.02,56.12,2772700,56.12
5 2012-09-07,56.60,57.20,56.38,57.19,3011800,57.19
6 2012-09-06,55.08,56.32,54.69,56.26,3304200,56.26
7 2012-09-05,54.55,54.82,54.34,54.67,2660700,54.67
8 2012-09-04,54.50,55.06,54.26,54.63,2299900,54.63
9 2012-08-31,54.41,54.90,54.15,54.51,2368400,54.51
10 2012-08-30,54.01,54.24,53.59,54.09,2233600,54.09
11 2012-08-29,53.86,54.68,53.75,54.49,2772800,54.49
12 2012-08-28,54.00,54.29,53.79,53.83,2065600,53.83
13 2012-08-27,54.79,54.85,53.94,54.15,1736300,54.15
```



	A	B	C	D	E	F	G
1	Date	Open	High	Low	Close	Volume	Adj Close
2	9/12/12	57.01	57.54	56.68	56.91	2362700	56.91
3	9/11/12	56.15	56.73	56.14	56.68	2118300	56.68
4	9/10/12	57.01	57.07	56.02	56.12	2772700	56.12
5	9/7/12	56.6	57.2	56.38	57.19	3011800	57.19
6	9/6/12	55.08	56.32	54.69	56.26	3304200	56.26
7	9/5/12	54.55	54.82	54.34	54.67	2660700	54.67



# What is in a Historical Stock Data File?

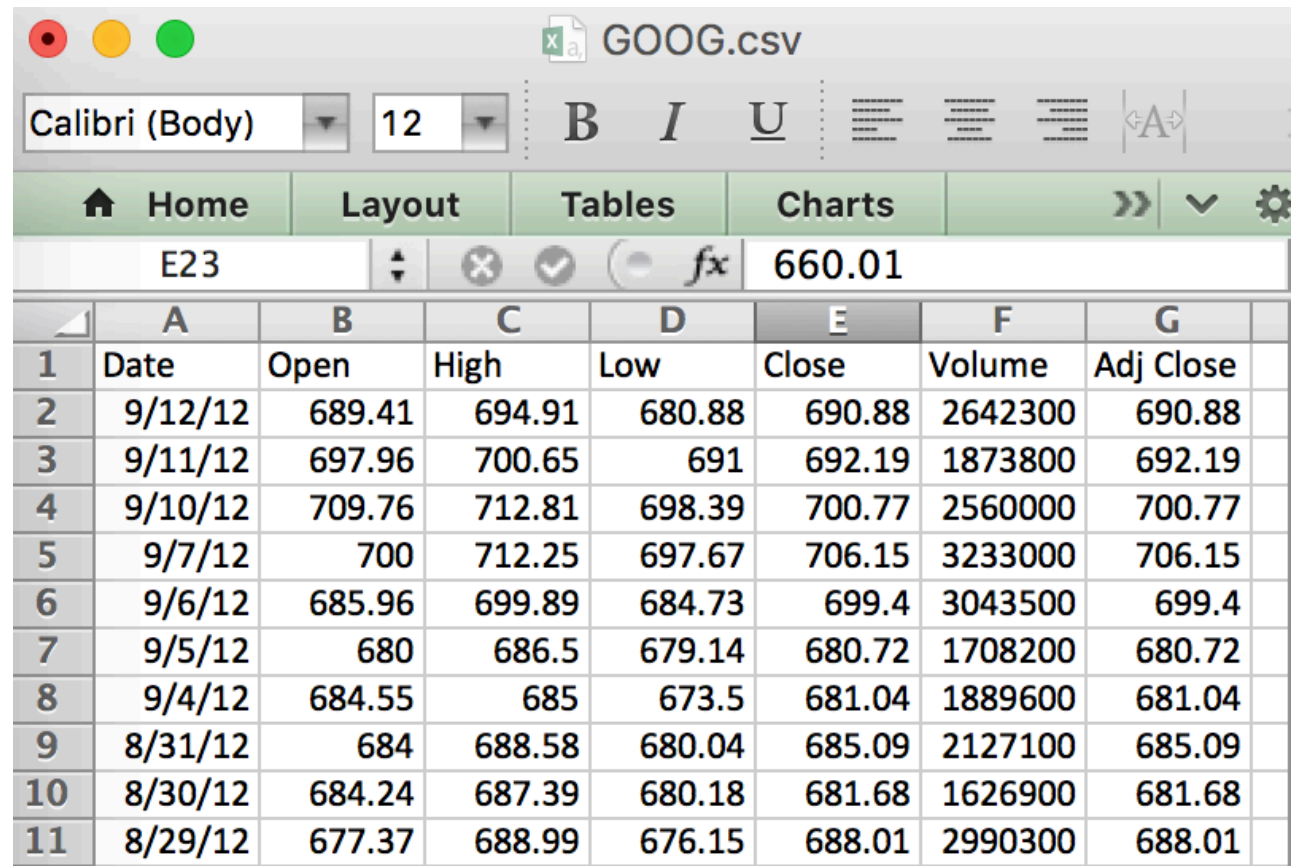
- # of employees
- Date/Time
- Company Name
- Price of the Stock
- Company's Hometown

# Stock Data Files

- **Date**
- **Open** – price stock opens at in the morning, first price in the day.
- **High** – highest price in the day
- **Low** – lowest price in the day
- **Close** – closing price at 4 PM.
- **Volume** – how many shares traded all together in the day.
- **Adjusted Close** – splits/and dividends – encapsulates the increase in value if you hold stock for a long time.

# GOOG.csv (from Yahoo).

- New dates on top, older descending.



The screenshot shows a spreadsheet application window titled "GOOG.csv". The interface includes a font menu with "Calibri (Body)" and size "12", and a ribbon with tabs for "Home", "Layout", "Tables", and "Charts". The active cell is E23, containing the value "660.01". The data table below has columns A through G and rows 1 through 11. The columns are labeled "Date", "Open", "High", "Low", "Close", "Volume", and "Adj Close". The data is sorted by date in descending order, with the most recent date (9/12/12) at the top.

	A	B	C	D	E	F	G
1	Date	Open	High	Low	Close	Volume	Adj Close
2	9/12/12	689.41	694.91	680.88	690.88	2642300	690.88
3	9/11/12	697.96	700.65	691	692.19	1873800	692.19
4	9/10/12	709.76	712.81	698.39	700.77	2560000	700.77
5	9/7/12	700	712.25	697.67	706.15	3233000	706.15
6	9/6/12	685.96	699.89	684.73	699.4	3043500	699.4
7	9/5/12	680	686.5	679.14	680.72	1708200	680.72
8	9/4/12	684.55	685	673.5	681.04	1889600	681.04
9	8/31/12	684	688.58	680.04	685.09	2127100	685.09
10	8/30/12	684.24	687.39	680.18	681.68	1626900	681.68
11	8/29/12	677.37	688.99	676.15	688.01	2990300	688.01

- Adjusted Close -- for stocks splits and dividend payments.
- Current Day – Adj Close and Close are always the same,
  - But as we go back in time start they to differ
  - Actual Return splits that is not captured by closing price.

# Pandas: Included in Anaconda

- [https://en.wikipedia.org/wiki/Pandas\\_\(software\)](https://en.wikipedia.org/wiki/Pandas_(software))
- Developed by Wes McKinney while at AQR Capital Management to analyze financial data
  - Open Source.
  - Numerical Tables and Time Series
  - Data Frames
    - **Slicing**
  - Panel Data

# Data Frame

- ETF
  - Exchange Traded Fund SPY

# Warmup: Reading into a data frame

- Interactively
  - Import pandas
  - Rename it to pd
- By a Program see next slide.

```
{ingrid:632} python
Python 2.7.11 |Anaconda 4.1.0 (x86_64)| (default, Jun 15 2016, 16:09:16)
[GCC 4.2.1 (Based on Apple Inc. build 5658) (LLVM build 2336.11.00)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://anaconda.org
>>> import pandas as pd
>>> df = pd.read_csv("data/AAPL.csv")
>>> print df.head()
   Date      Open      High      Low      Close      Volume      Adj Close
0 2012-09-12  666.85  669.90  656.00  669.79  25410600      669.79
1 2012-09-11  665.11  670.10  656.50  660.59  17987400      660.59
2 2012-09-10  680.45  683.29  662.10  662.74  17428500      662.74
3 2012-09-07  678.05  682.48  675.77  680.44  11773800      680.44
4 2012-09-06  673.17  678.29  670.80  676.27  13971300      676.27
>>> print df
   Date      Open      High      Low      Close      Volume      Adj Close
0 2012-09-12  666.85  669.90  656.00  669.79  25410600      669.79
1 2012-09-11  665.11  670.10  656.50  660.59  17987400      660.59
2 2012-09-10  680.45  683.29  662.10  662.74  17428500      662.74
3 2012-09-07  678.05  682.48  675.77  680.44  11773800      680.44
4 2012-09-06  673.17  678.29  670.80  676.27  13971300      676.27
5 2012-09-05  675.57  676.35  669.60  670.23  12013400      670.23
6 2012-09-04  665.76  675.14  664.50  674.97  13139000      674.97
7 2012-08-31  667.25  668.60  657.25  665.24  12082900      665.24
8 2012-08-30  670.64  671.55  662.85  663.87  10810700      663.87
9 2012-08-29  675.25  677.67  672.60  673.47   7243100      673.47
10 2012-08-28  674.98  676.10  670.67  674.80   9550600      674.80
11 2012-08-27  679.99  680.87  673.54  675.68  15250300      675.68
12 2012-08-24  659.51  669.48  655.55  663.22  15619300      663.22
13 2012-08-23  666.11  669.90  661.15  662.63  15004600      662.63
14 2012-08-22  654.42  669.00  648.11  668.87  20190100      668.87
15 2012-08-21  670.82  674.88  650.33  656.06  29025700      656.06
16 2012-08-20  650.01  665.15  649.90  665.15  21906600      665.15
17 2012-08-17  640.00  648.10  638.81  648.11  15812000      648.11
```

# Make it a function

```
1 import pandas as pd
2
3
4 def test_run():
5     df = pd.read_csv("data/AAPL.csv")
6     print df #print entire dataframe
7
8
9 if __name__ == "__main__":
10     test_run()
```

- readframe.py
  - Head, Entire frame
  - df.head()
- Question: Print last 5 lines?



# Manipulating Frames

- Mean is simple
- meanframe