Introduction to the Course

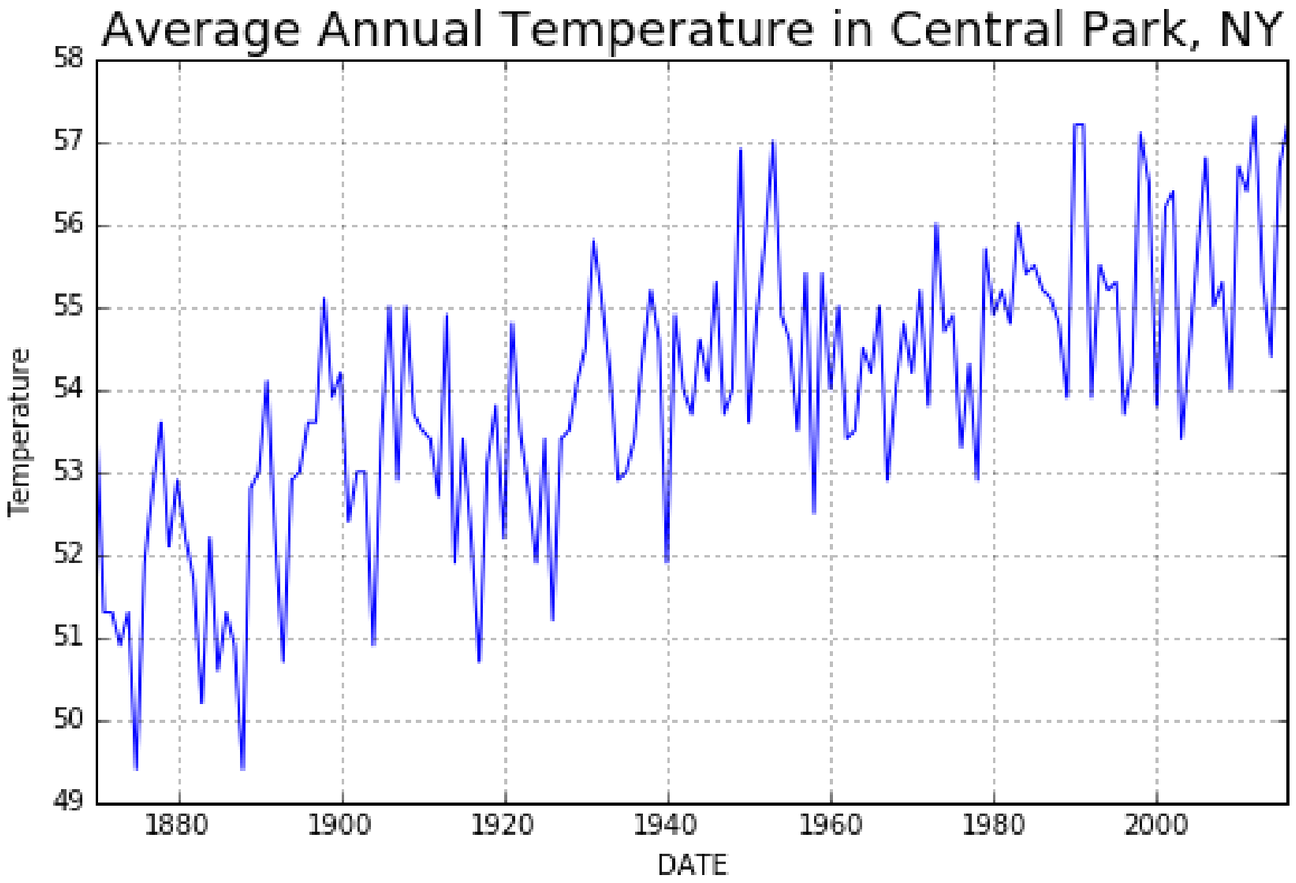
TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

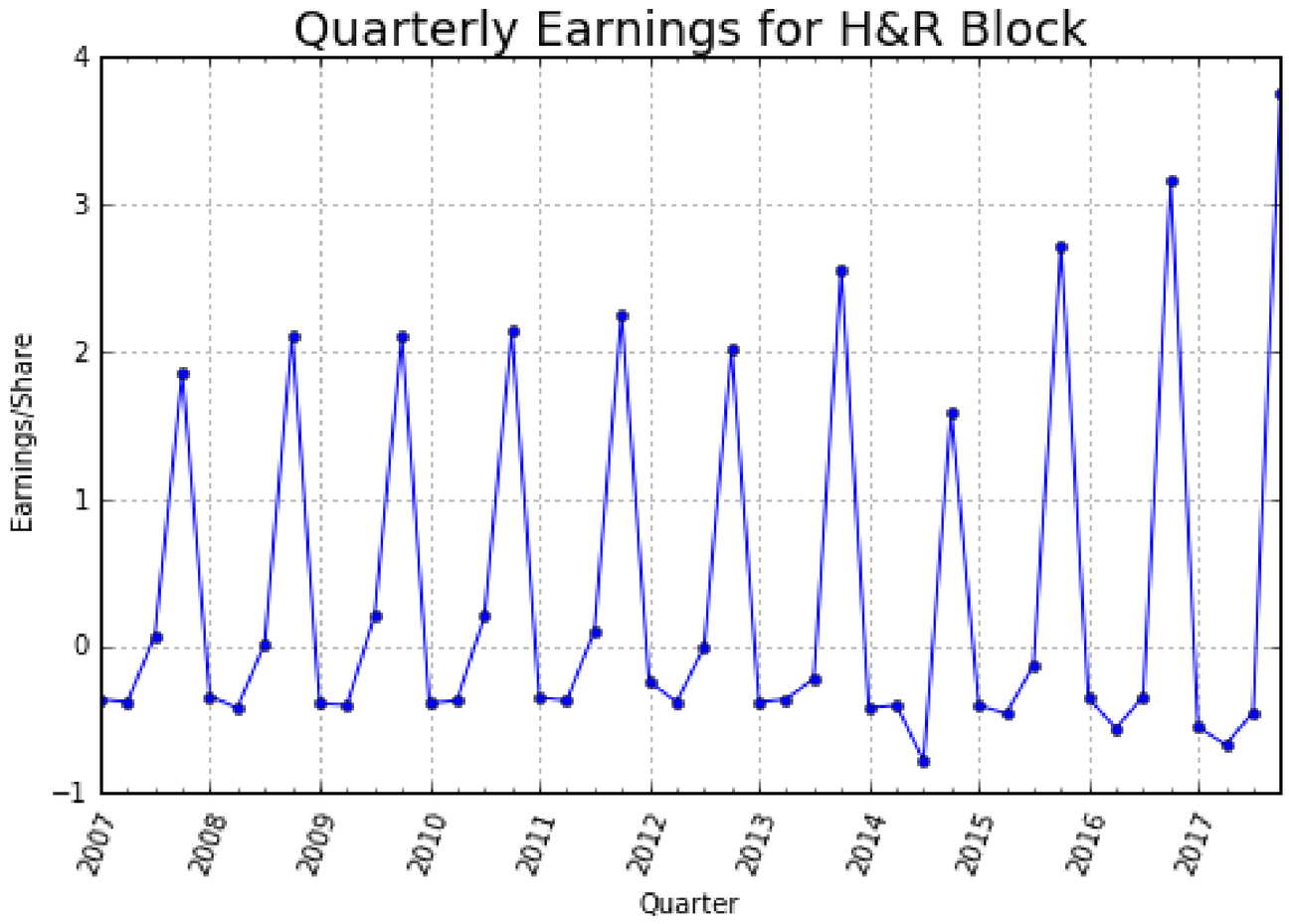
datacamp

Example of Time Series: Google Trends

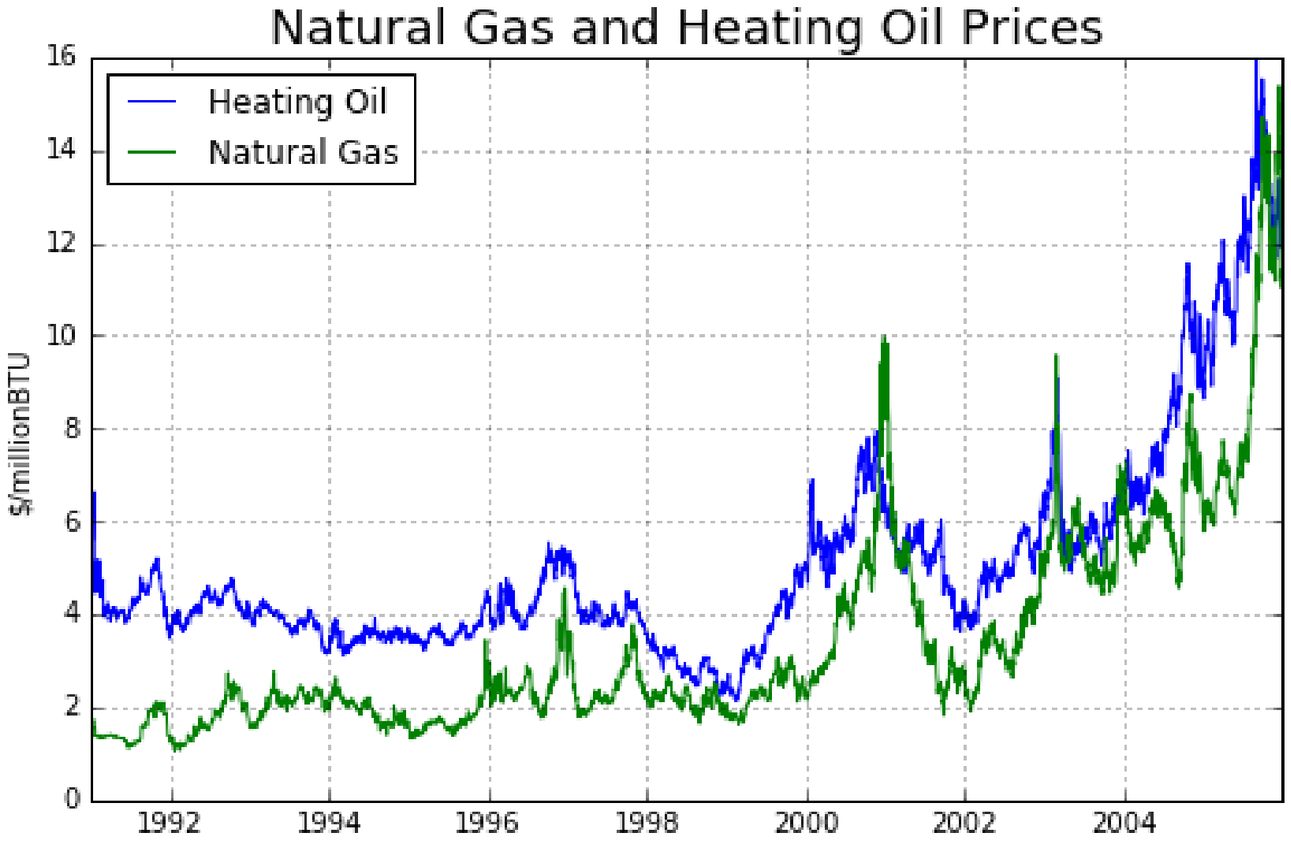
Example of Time Series: Climate Data



Example of Time Series: Quarterly Earnings Data



Example of Multiple Series: Natural Gas and Heating Oil



Goals of Course

- Learn about time series models
- Fit data to a times series model
- Use the models to make forecasts of the future
- Learn how to use the relevant statistical packages in Python
- Provide concrete examples of how these models are used

Some Useful Pandas Tools

- Changing an index to datetime

```
df.index = pd.to_datetime(df.index)
```

- Plotting data

```
df.plot()
```

- Slicing data

```
df['2012']
```

Some Useful Pandas Tools

- Join two DataFrames

```
df1.join(df2)
```

- Resample data (e.g. from daily to weekly)

```
df = df.resample(rule='W', how='last')
```

More pandas Functions

- Computing percent changes and differences of a time series

```
df['col'].pct_change()
df['col'].diff()
```

- `pandas` correlation method of Series

```
df['ABC'].corr(df['XYZ'])
```

- `pandas` autocorrelation

```
df['ABC'].autocorr()
```

Let's practice!

TIME SERIES ANALYSIS IN PYTHON

Correlation of Two Time Series

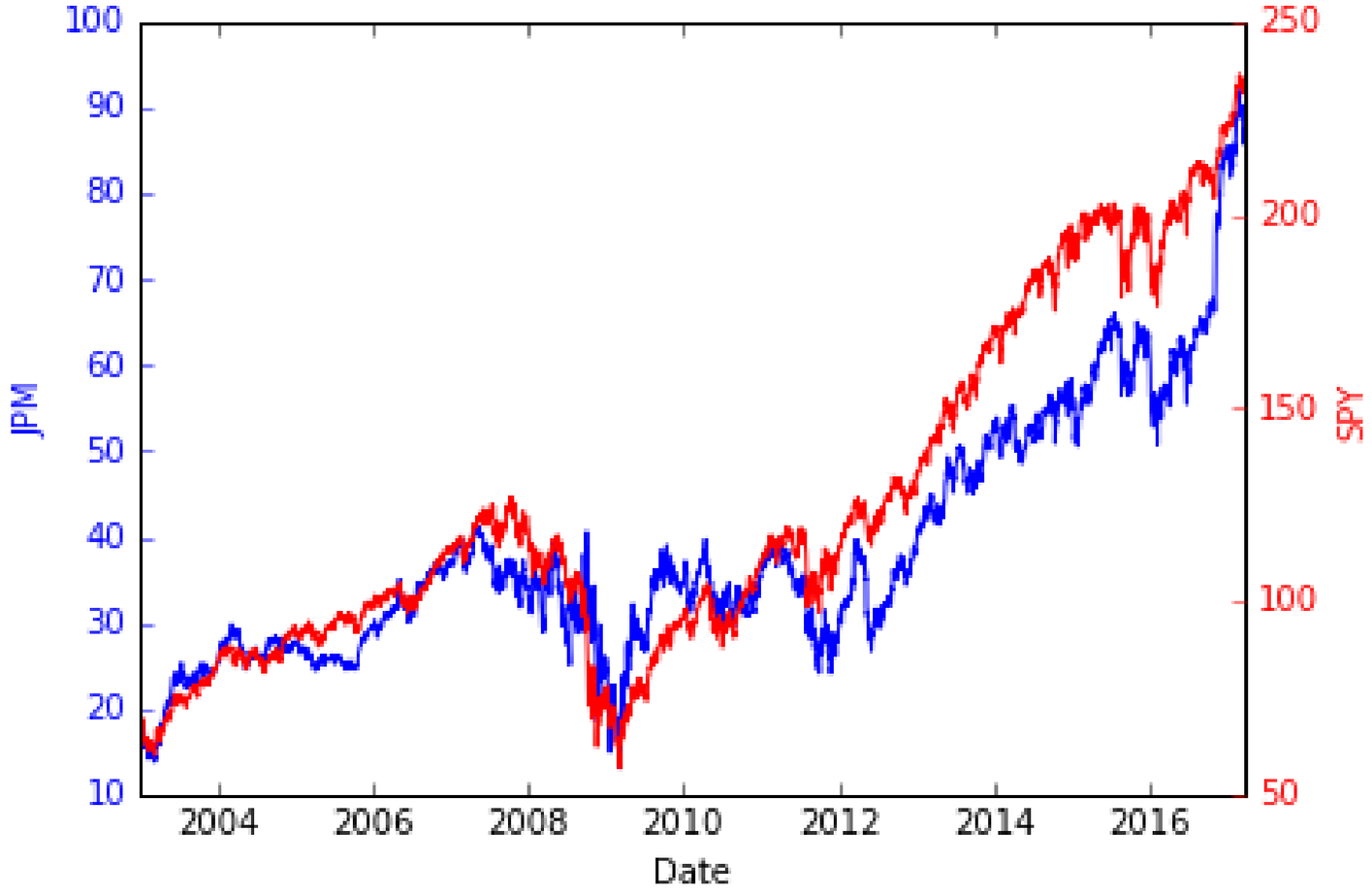
TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

datacamp

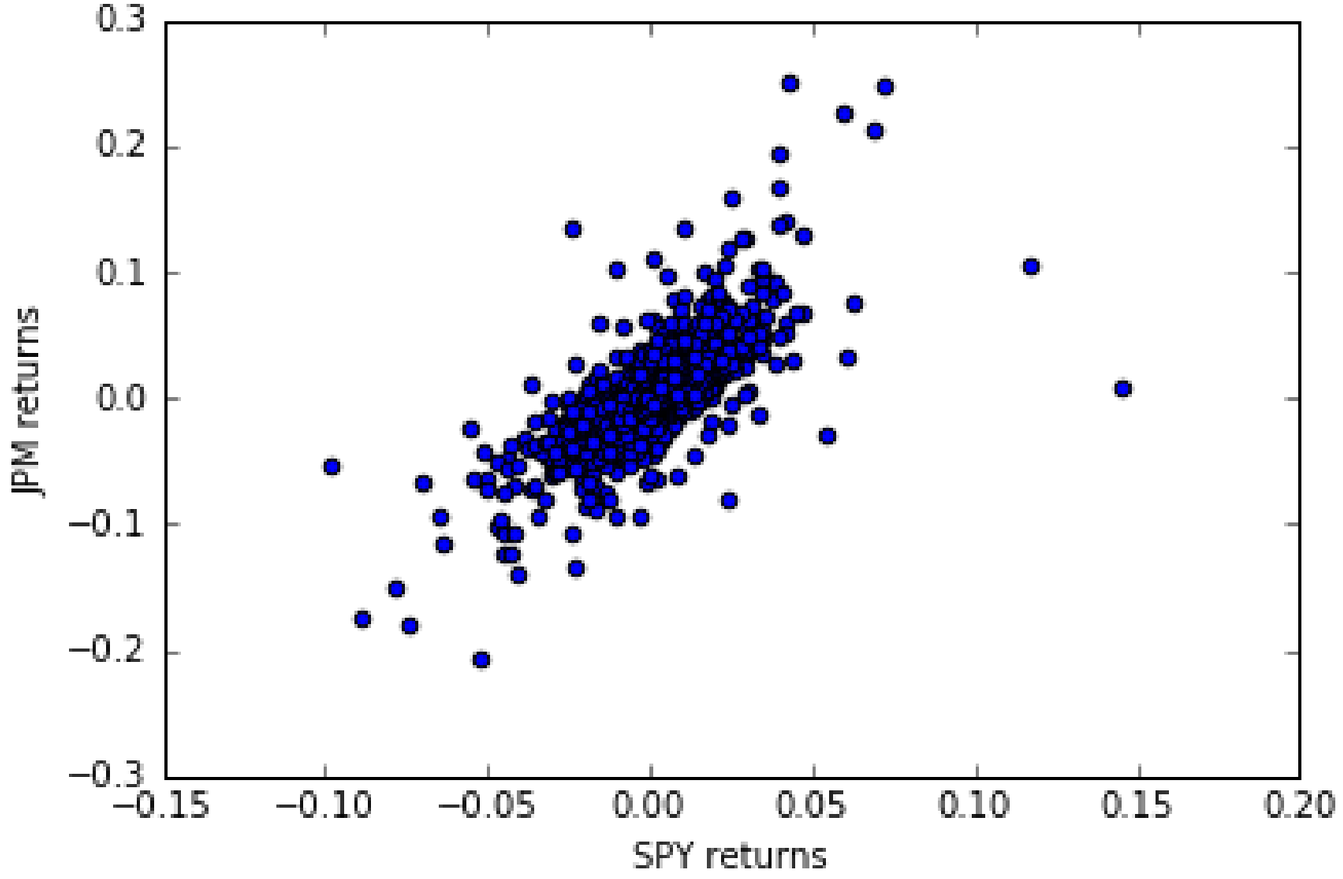
Correlation of Two Time Series

- Plot of S&P500 and JPMorgan stock



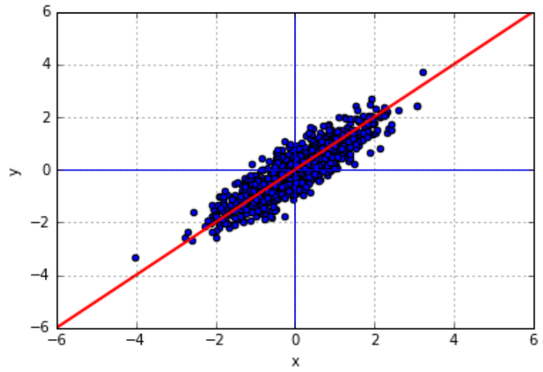
Correlation of Two Time Series

- Scatter plot of S&P500 and JP Morgan returns

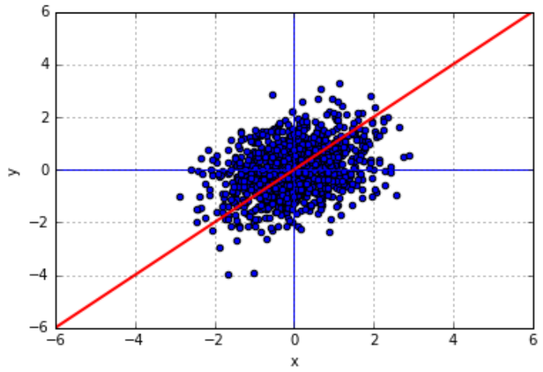


More Scatter Plots

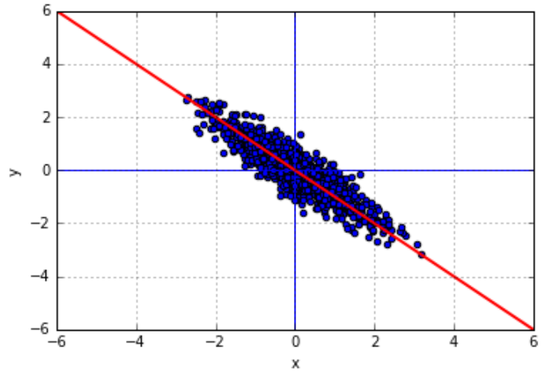
- Correlation = 0.9



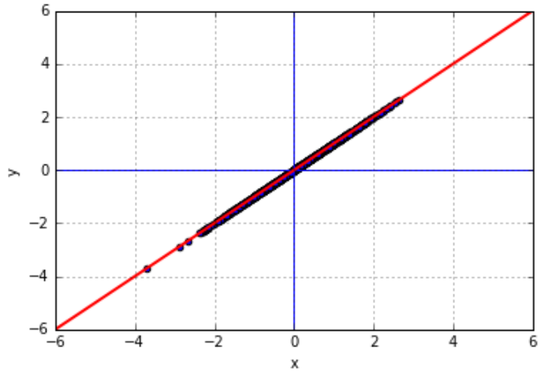
- Correlation = 0.4



- Correlation = -0.9

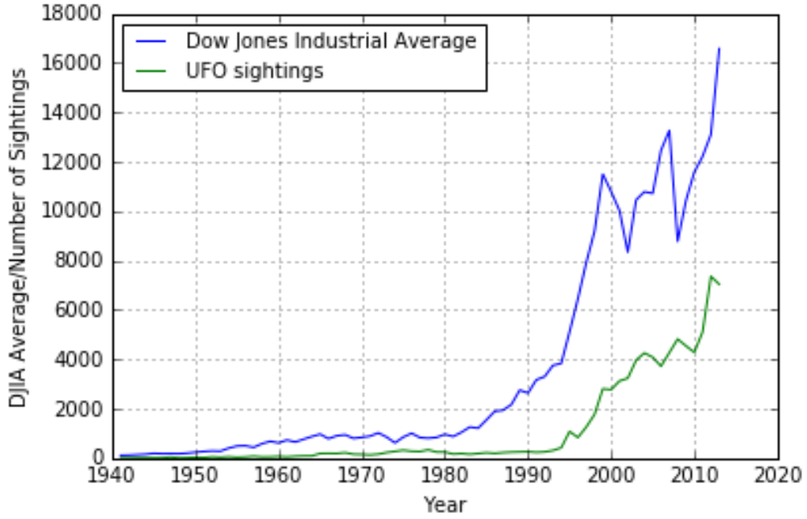


- Corelation = 1.0



Common Mistake: Correlation of Two Trending Series

- Dow Jones Industrial Average and UFO Sightings (**www.nuforc.org**)



- Correlation of levels: 0.94
- Correlation of percent changes: ≈ 0

Example: Correlation of Large Cap and Small Cap Stocks

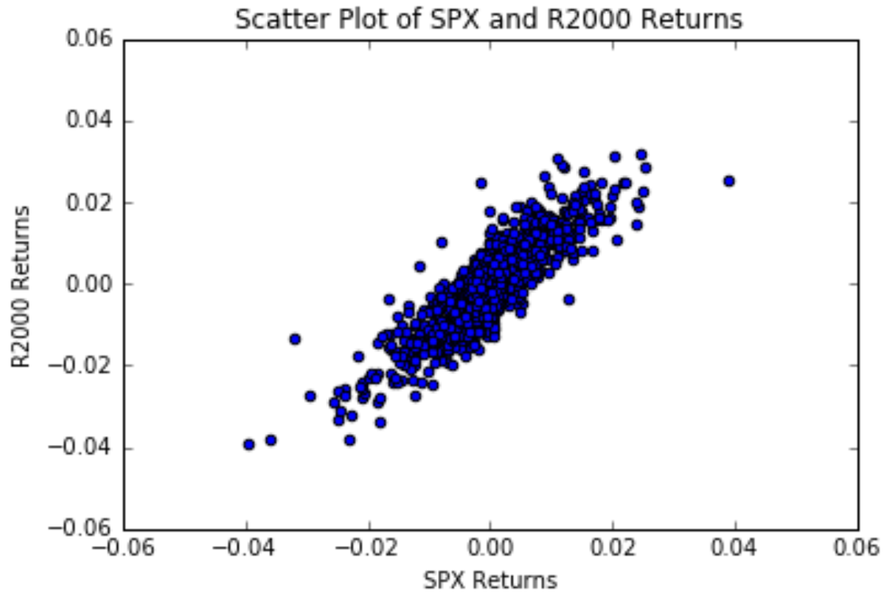
- Start with stock prices of SPX (large cap) and R2000 (small cap)
- First step: Compute percentage changes of both series

```
df['SPX_Ret'] = df['SPX_Prices'].pct_change()
df['R2000_Ret'] = df['R2000_Prices'].pct_change()
```

Example: Correlation of Large Cap and Small Cap Stocks

- Visualize correlation with scattter plot

```
plt.scatter(df['SPX_Ret'], df['R2000_Ret'])
plt.show()
```



Example: Correlation of Large Cap and Small Cap Stocks

- Use pandas correlation method for Series

```
correlation = df['SPX_Ret'].corr(df['R2000_Ret'])
print("Correlation is: ", correlation)
```

```
Correlation is: 0.868
```

Let's practice!

TIME SERIES ANALYSIS IN PYTHON

Simple Linear Regressions

TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

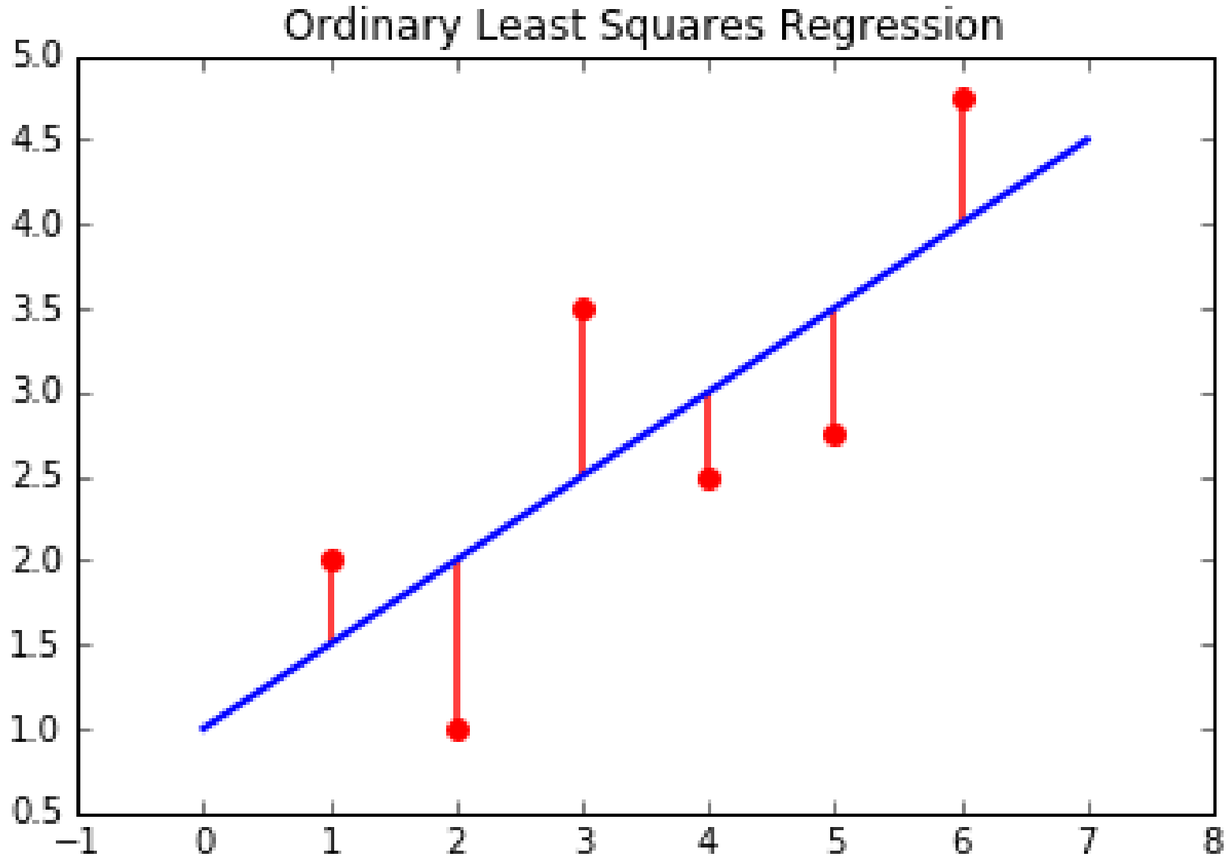
datacamp

What is a Regression?

- **Simple linear regression:** $y_t = \alpha + \beta x_t + \epsilon_t$

What is a Regression?

- **Ordinary Least Squares (OLS)**



Python Packages to Perform Regressions

- In statsmodels:

```
import statsmodels.api as sm
sm.OLS(y, x).fit()
```

- In numpy:

```
np.polyfit(x, y, deg=1)
```

- In pandas:

```
pd.ols(y, x)
```

- In scipy:

```
from scipy import stats
stats.linregress(x, y)
```

Warning: the order of `x` and `y` is not consistent across packages

Example: Regression of Small Cap Returns on Large Cap

- Import the statsmodels module

```
import statsmodels.api as sm
```

- As before, compute percentage changes in both series

```
df['SPX_Ret'] = df['SPX_Prices'].pct_change()
df['R2000_Ret'] = df['R2000_Prices'].pct_change()
```

- Add a constant to the DataFrame for the regression intercept

```
df = sm.add_constant(df)
```

Regression Example (continued)

- Notice that the first row of returns is NaN

```
            SPX_Price  R2000_Price   SPX_Ret  R2000_Ret
Date
2012-11-01  1427.589966   827.849976       NaN        NaN
2012-11-02  1414.199951   814.369995 -0.009379  -0.016283
```

- Delete the row of NaN

```
df = df.dropna()
```

- Run the regression

```
results = sm.OLS(df['R2000_Ret'],df[['const','SPX_Ret']]).fit()
print(results.summary())
```

Regression Example (continued)

```
                            OLS Regression Results
==============================================================================
Dep. Variable:              R2000_Ret   R-squared:                       0.753
Model:                            OLS   Adj. R-squared:                  0.753
Method:                 Least Squares   F-statistic:                     3829.
Date:                Fri, 26 Jan 2018   Prob (F-statistic):               0.00
Time:                        13:29:55   Log-Likelihood:                 4882.4
No. Observations:                1257   AIC:                            -9761.
Df Residuals:                    1255   BIC:                            -9751.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [95.0% Conf. Int.]
------------------------------------------------------------------------------
const      -4.964e-05      0.000     -0.353      0.724        -0.000      0.000
SPX_Ret        1.1412      0.018     61.877      0.000         1.105      1.177
==============================================================================
Omnibus:                       61.950   Durbin-Watson:                   1.991
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              148.100
Skew:                           0.266   Prob(JB):                     6.93e-33
Kurtosis:                       4.595   Cond. No.                         131.
==============================================================================
```

- Regression output
- Intercept in `results.params[0]`
- Slope in `results.params[1]`

Regression Example (continued)

- Regression output

```
                            OLS Regression Results
==============================================================================
Dep. Variable:              R2000_Ret   R-squared:                       0.753
Model:                            OLS   Adj. R-squared:                  0.753
Method:                 Least Squares   F-statistic:                     3829.
Date:                Fri, 26 Jan 2018   Prob (F-statistic):               0.00
Time:                        13:29:55   Log-Likelihood:                 4882.4
No. Observations:                1257   AIC:                            -9761.
Df Residuals:                    1255   BIC:                            -9751.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [95.0% Conf. Int.]
------------------------------------------------------------------------------
const      -4.964e-05      0.000     -0.353      0.724        -0.000     0.000
SPX_Ret        1.1412      0.018     61.877      0.000         1.105     1.177
==============================================================================
Omnibus:                       61.950   Durbin-Watson:                   1.991
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              148.100
Skew:                           0.266   Prob(JB):                     6.93e-33
Kurtosis:                       4.595   Cond. No.                         131.
==============================================================================
```

Relationship Between R-Squared and Correlation

- $[\text{corr}(x, y)]^2 = R^2$ (or R-squared)
- $\text{sign}(\text{corr}) = \text{sign}(\text{regression slope})$
- In last example:
  - R-Squared = 0.753
  - Slope is positive
  - correlation = $+\sqrt{0.753} = 0.868$

Let's practice!

TIME SERIES ANALYSIS IN PYTHON

Autocorrelation

TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

datacamp

What is Autocorrelation?

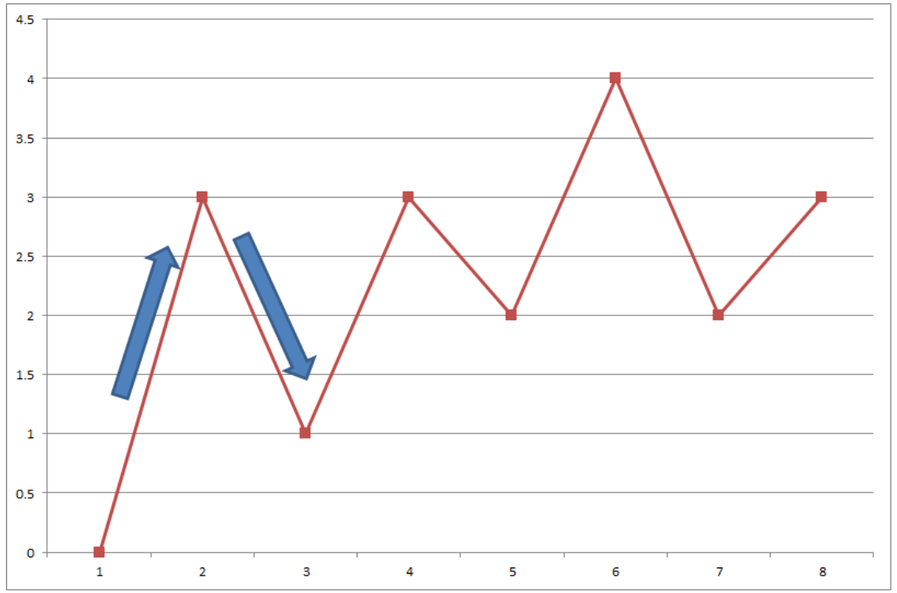
- Correlation of a time series with a lagged copy of itself

Series	Lagged Series
5	
10	5
15	10
20	15
25	20
⋮	⋮

- Lag-one autocorrelation
- Also called **serial correlation**

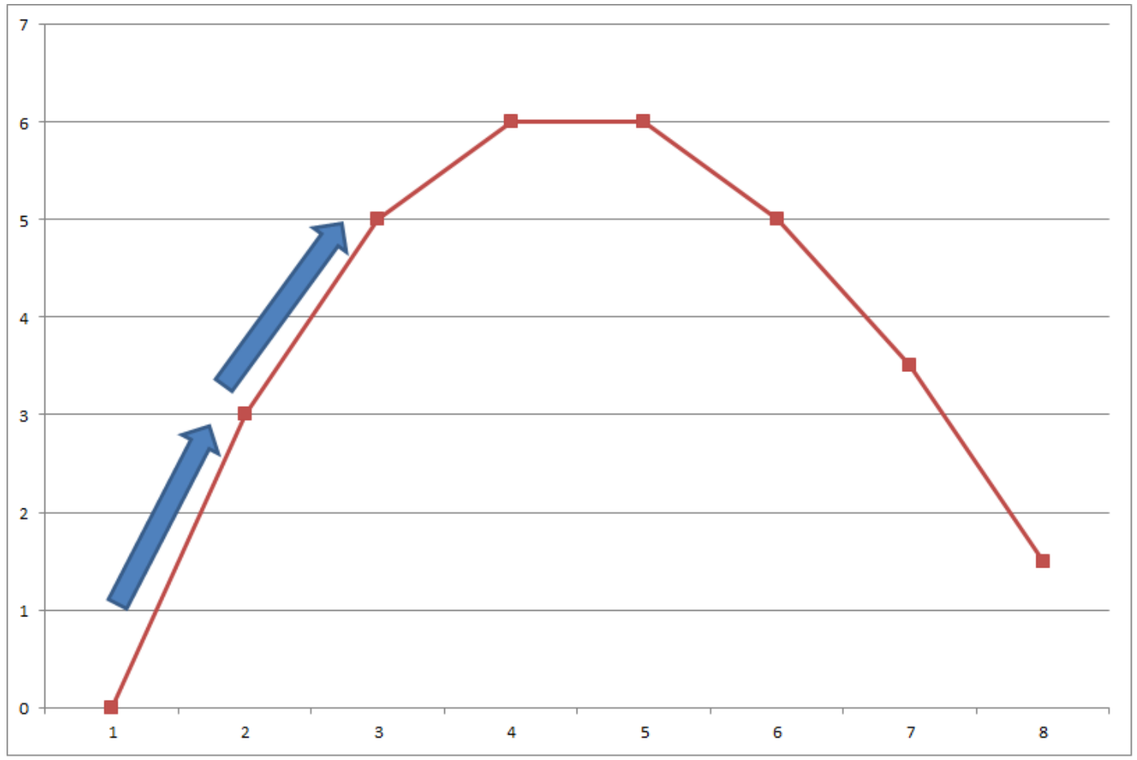
Interpretation of Autocorrelation

- **Mean Reversion** - Negative autocorrelation



Interpretation of Autocorrelation

- **Momentum**, or **Trend Following** - Positive autocorrelation



Traders Use Autocorrelation to Make Money

- Individual stocks
  - Historically have negative autocorrelation
  - Measured over short horizons (days)
  - Trading strategy: Buy losers and sell winners
- Commodities and currencies
  - Historically have positive autocorrelation
  - Measured over longer horizons (months)
  - Trading strategy: Buy winners and sell losers

Example of Positive Autocorrelation: Exchange Rates

- Use daily ¥/$ exchange rates in DataFrame `df` from **FRED**
- Convert index to datetime

```
# Convert index to datetime
df.index = pd.to_datetime(df.index)
# Downsample from daily to monthly data
df = df.resample(rule='M', how='last')
# Compute returns from prices
df['Return'] = df['Price'].pct_change()
# Compute autocorrelation
autocorrelation = df['Return'].autocorr()
print("The autocorrelation is: ",autocorrelation)
```

```
The autocorrelation is: 0.0567
```

Let's practice!

TIME SERIES ANALYSIS IN PYTHON

Autocorrelation Function

TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

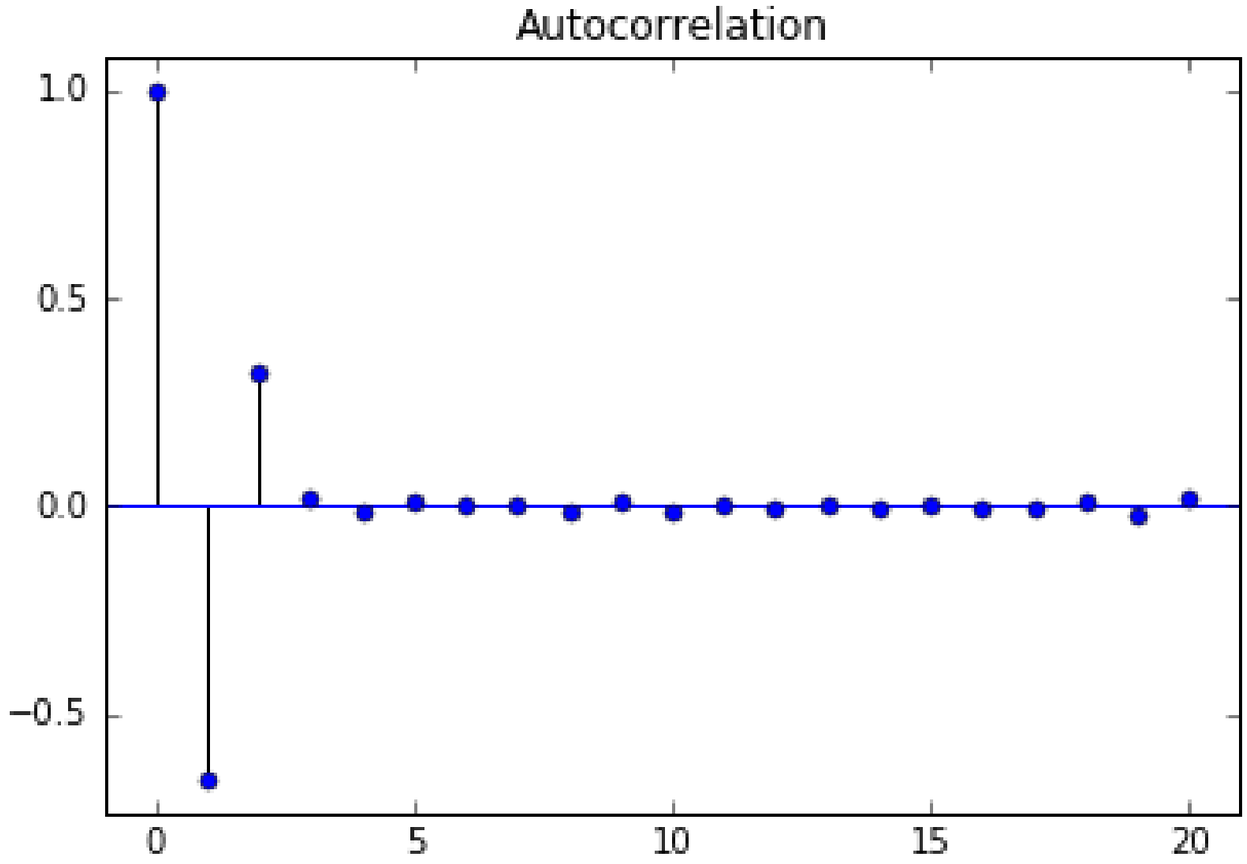
datacamp

Autocorrelation Function

- Autocorrelation Function (ACF): The autocorrelation as a function of the lag
- Equals one at lag-zero
- Interesting information beyond lag-one

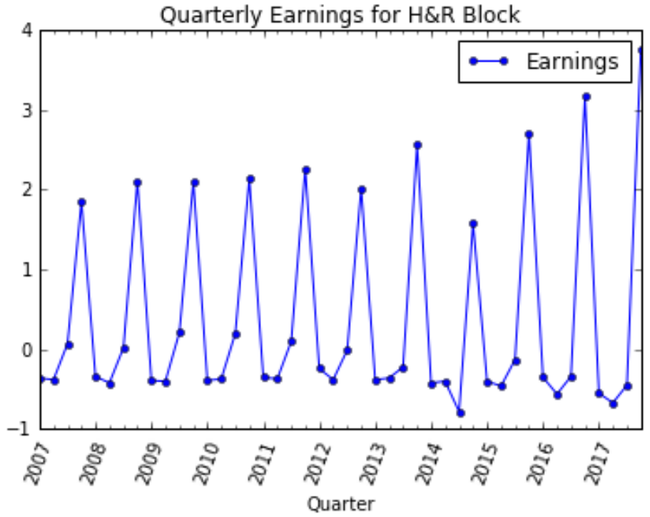
ACF Example 1: Simple Autocorrelation Function

- Can use last two values in series for forecasting

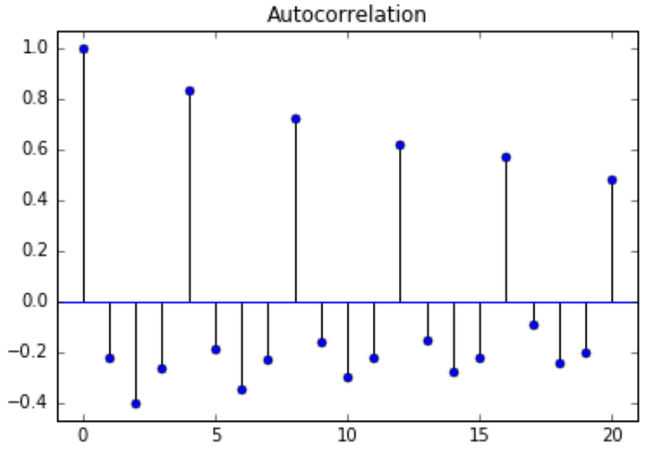


ACF Example 2: Seasonal Earnings

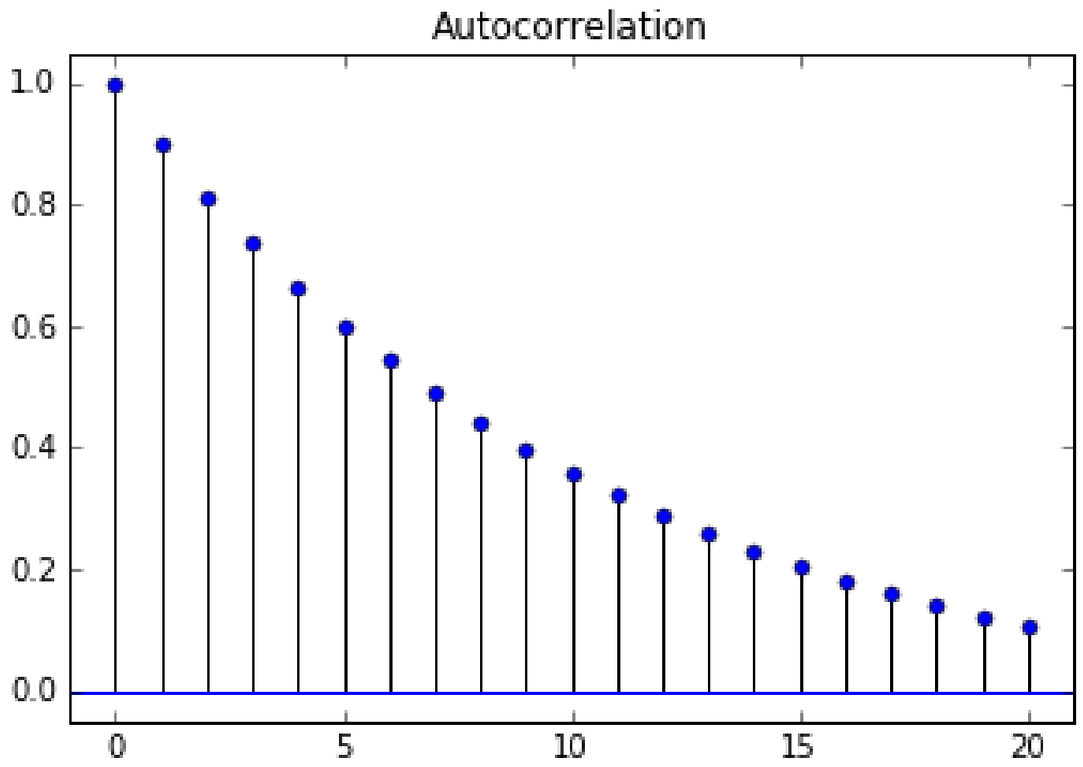
- Earnings for H&R Block



- ACF for H&R Block



ACF Example 3: Useful for Model Selection



- Model selection

Plot ACF in Python

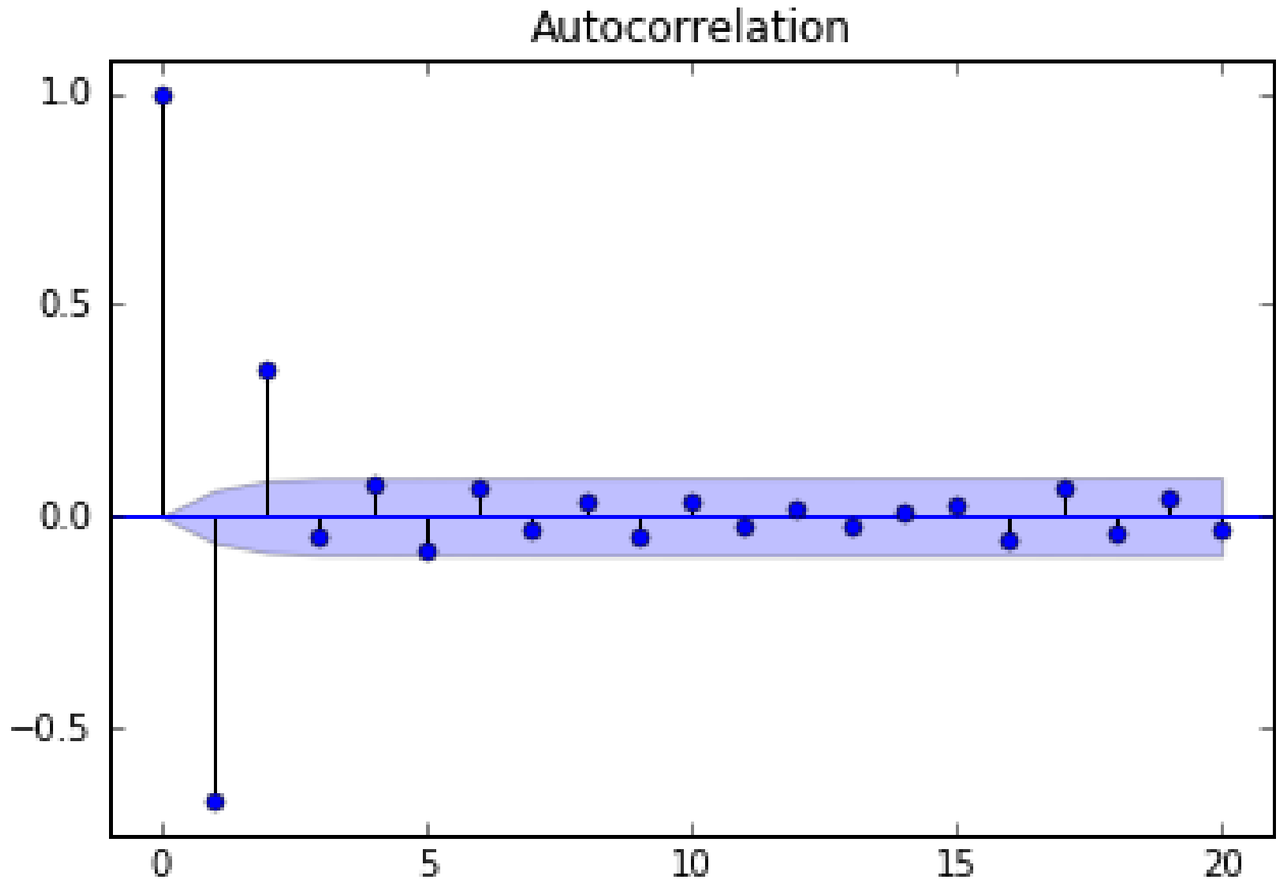
- Import module:

```
from statsmodels.graphics.tsaplots import plot_acf
```

- Plot the ACF:

```
plot_acf(x, lags= 20, alpha=0.05)
```

Confidence Interval of ACF



Confidence Interval of ACF

- Argument `alpha` sets the width of confidence interval
- Example: `alpha=0.05`
  - 5% chance that if true autocorrelation is zero, it will fall outside blue band
- Confidence bands are wider if:
  - Alpha lower
  - Fewer observations
- Under some simplifying assumptions, 95% confidence bands are $\pm 2/\sqrt{N}$
- If you want no bands on plot, set `alpha=1`

ACF Values Instead of Plot

```
from statsmodels.tsa.stattools import acf
print(acf(x))
```

```
[ 1.         -0.6765505   0.34989905 -0.01629415 -0.0250701
 -0.03186545  0.01399904 -0.03518128  0.02063168 -0.0262064
...
  0.07191516 -0.12211912  0.14514481 -0.09644228  0.0521588
```

Let's practice!

TIME SERIES ANALYSIS IN PYTHON

White Noise

TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

datacamp

What is White Noise?

- White Noise is a series with:
  - Constant mean
  - Constant variance
  - Zero autocorrelations at all lags
- Special Case: if data has normal distribution, then *Gaussian White Noise*

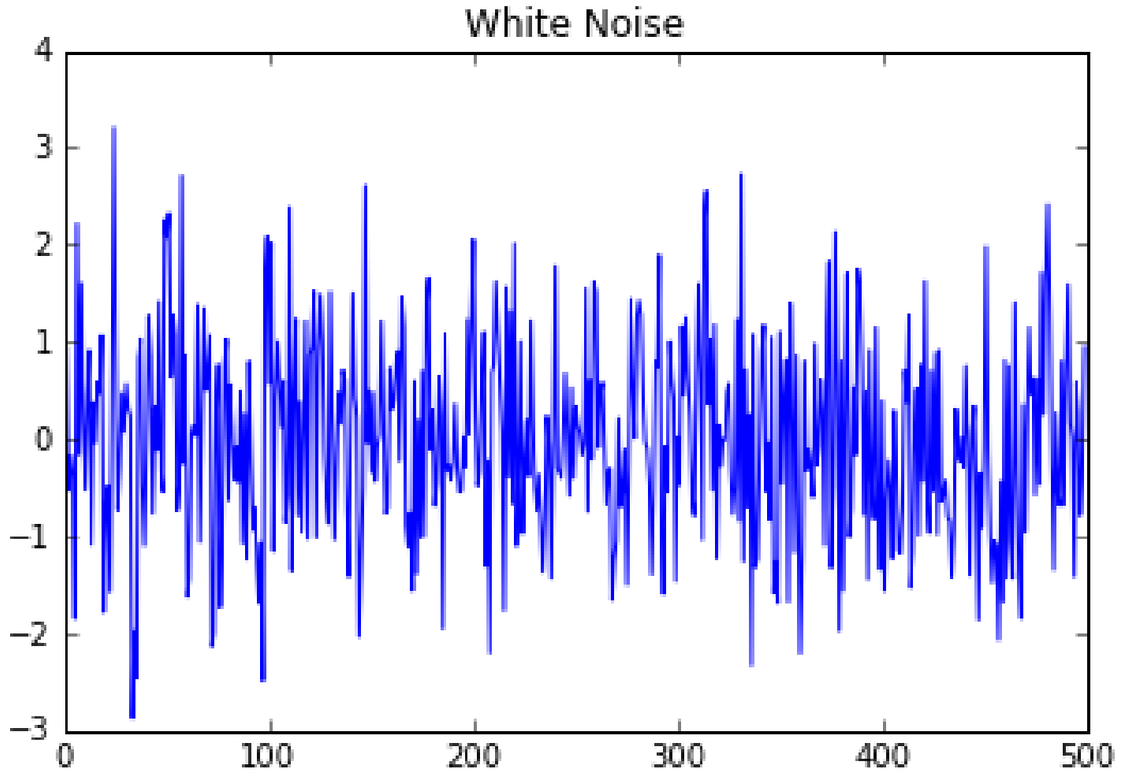
Simulating White Noise

- It's very easy to generate white noise

```
import numpy as np
noise = np.random.normal(loc=0, scale=1, size=500)
```

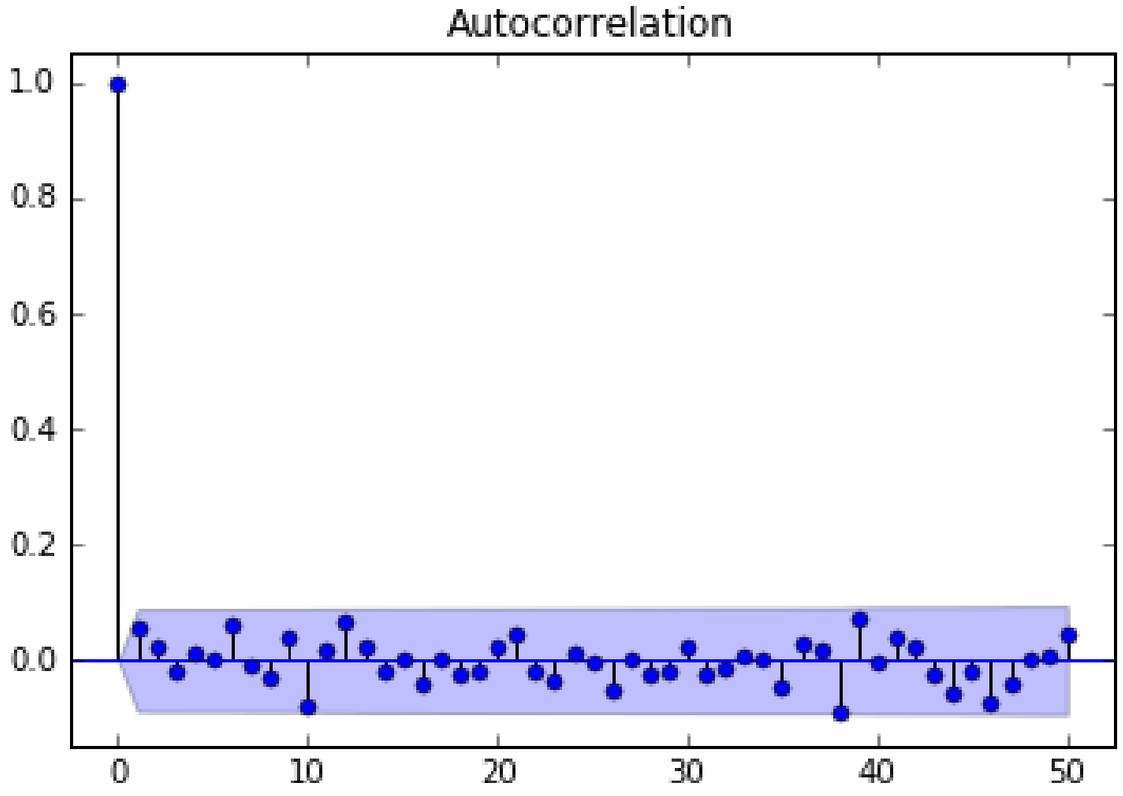
What Does White Noise Look Like?

```
plt.plot(noise)
```



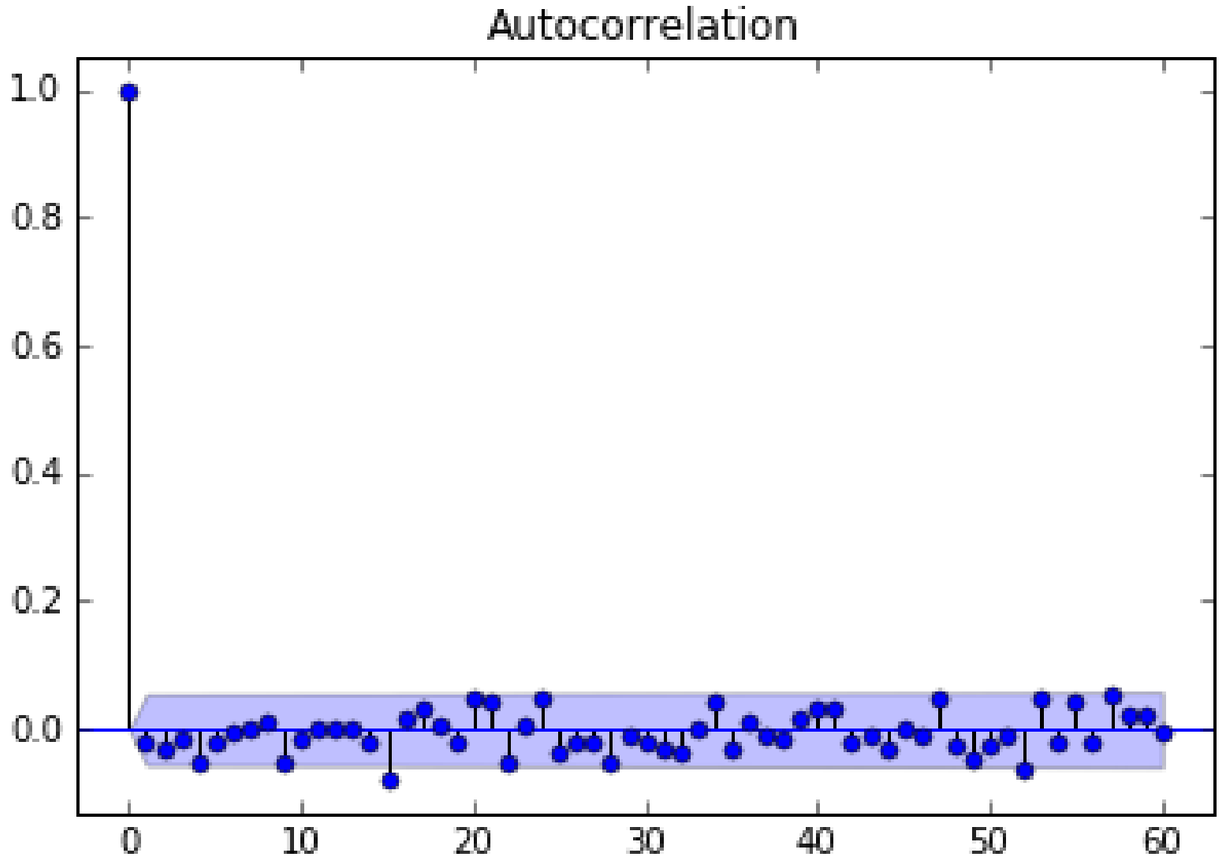
Autocorrelation of White Noise

```
plot_acf(noise, lags=50)
```



Stock Market Returns: Close to White Noise

- Autocorrelation Function for the S&P500



Let's practice!

TIME SERIES ANALYSIS IN PYTHON

Random Walk

TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

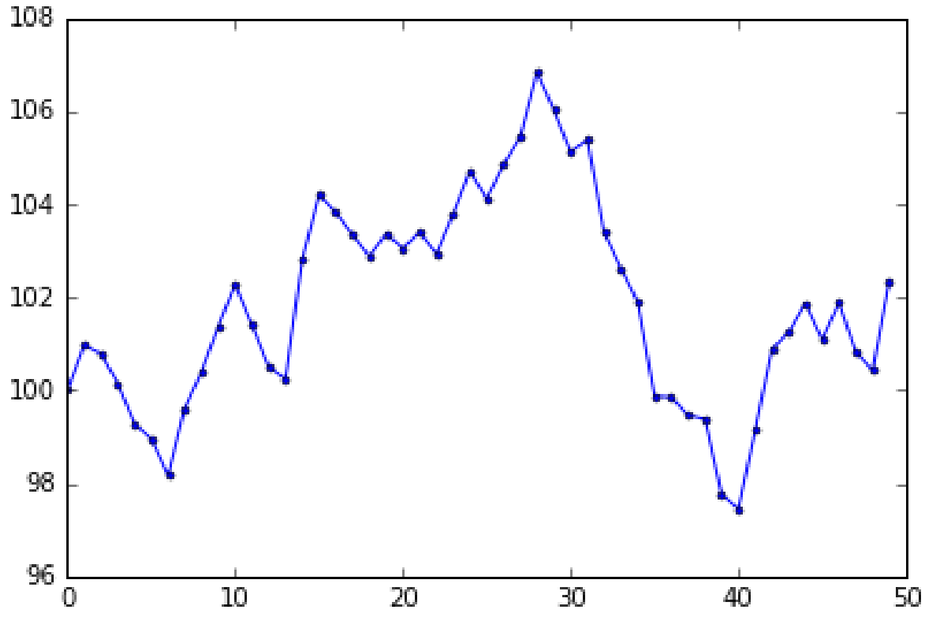
datacamp

What is a Random Walk?

- Today's Price = Yesterday's Price + Noise

$$P_t = P_{t-1} + \epsilon_t$$

- Plot of simulated data



What is a Random Walk?

- Today's Price = Yesterday's Price + Noise

$$P_t = P_{t-1} + \epsilon_t$$

- Change in price is white noise

$$P_t - P_{t-1} = \epsilon_t$$

- Can't forecast a random walk
- Best forecast for tomorrow's price is today's price

What is a Random Walk?

- Today's Price = Yesterday's Price + Noise

$$P_t = P_{t-1} + \epsilon_t$$

- Random walk with drift:

$$P_t = \mu + P_{t-1} + \epsilon_t$$

- Change in price is white noise with non-zero mean:

$$P_t - P_{t-1} = \mu + \epsilon_t$$

Statistical Test for Random Walk

- Random walk with drift

$$P_t = \mu + P_{t-1} + \epsilon_t$$

- Regression test for random walk

$$P_t = \alpha + \beta P_{t-1} + \epsilon_t$$

- Test: $H_0 : \beta = 1$ (random walk) $H_1 : \beta < 1$ (not random walk)

Statistical Test for Random Walk

- Regression test for random walk

$$P_t = \alpha + \beta P_{t-1} + \epsilon_t$$

- Equivalent to

$$P_t - P_{t-1} = \alpha + \beta P_{t-1} + \epsilon_t$$

- Test: $H_0 : \beta = 0$ (random walk) $H_1 : \beta < 0$ (not random walk)

Statistical Test for Random Walk

- Regression test for random walk

$$P_t - P_{t-1} = \alpha + \beta P_{t-1} + \epsilon_t$$

- Test: $H_0 : \beta = 0$ (random walk) $H_1 : \beta < 0$ (not random walk)
- This test is called the **Dickey-Fuller** test
- If you add more lagged changes on the right hand side, it's the **Augmented Dickey-Fuller** test

ADF Test in Python

- Import module from statsmodels

```
from statsmodels.tsa.stattools import adfulle
```

- Run Augmented Dickey-Test

```
adfuller(x)
```

Example: Is the S&P500 a Random Walk?

```
# Run Augmented Dickey-Fuller Test on SPX data
results = adfuller(df['SPX'])
```

```
# Print p-value
print(results[1])
```

```
0.782253808587
```

```
# Print full results
print(results)
```

```
(-0.91720490331127869,
 0.78225380858668414,
 0,
 1257,
 {'1%': -3.4355629707955395,
  '10%': -2.567995644141416,
  '5%': -2.8638420633876671},
 10161.888789598503)
```

Let's practice!

TIME SERIES ANALYSIS IN PYTHON

Stationarity

TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

What is Stationarity?

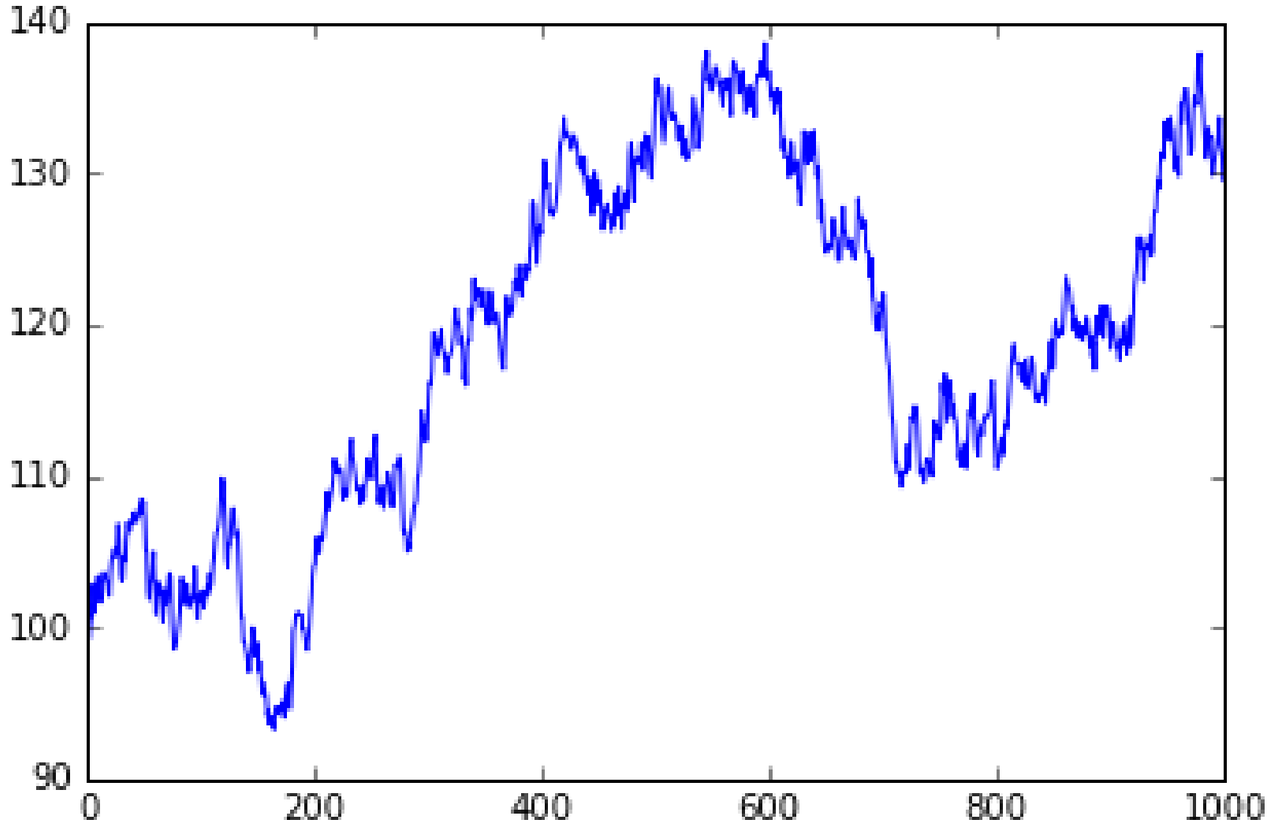
- **Strong stationarity**: entire distribution of data is time-invariant
- **Weak stationarity**: mean, variance and autocorrelation are time-invariant (i.e., for autocorrelation, corr$\left(X_t, X_{t-\tau}\right)$ is only a function of τ)

Why Do We Care?

- If parameters vary with time, too many parameters to estimate
- Can only estimate a parsimonious model with a few parameters

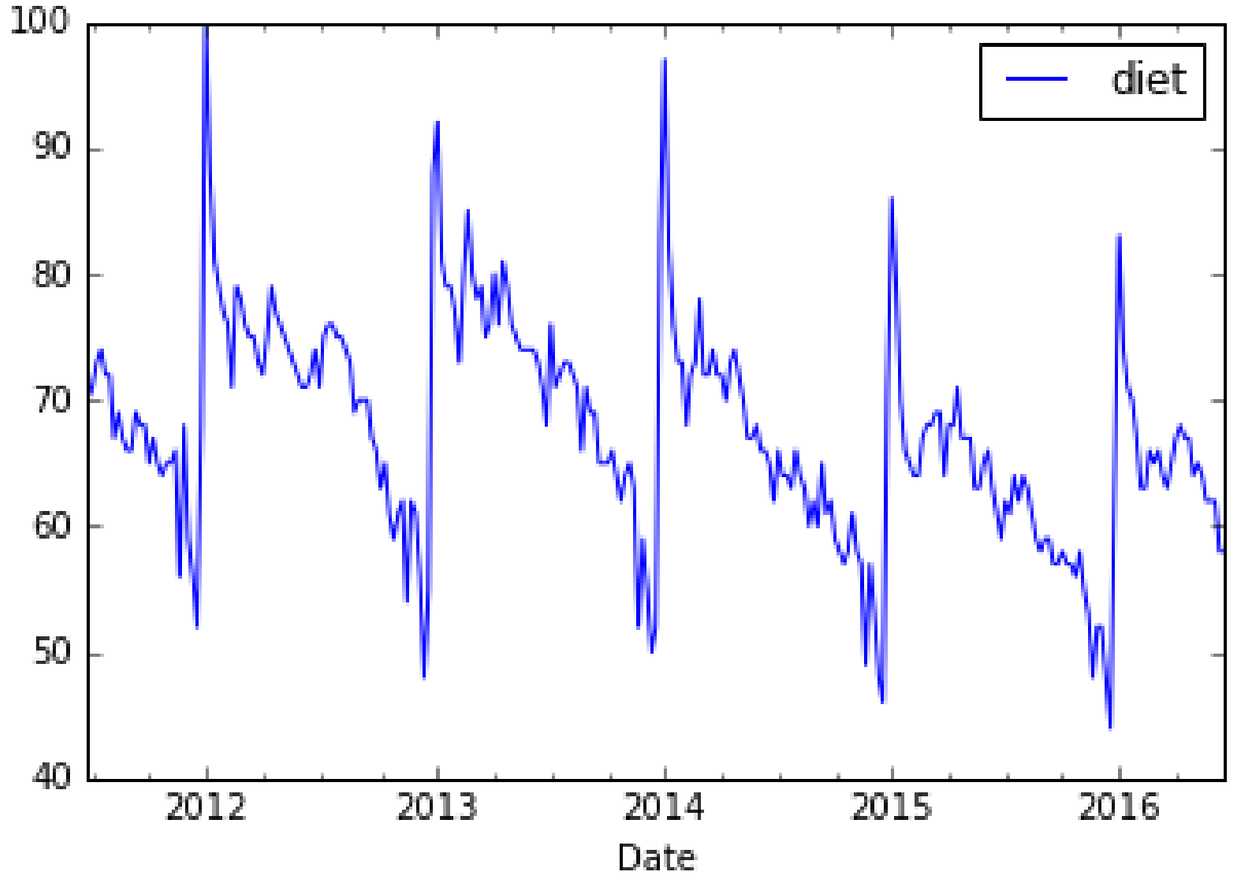
Examples of Nonstationary Series

- Random Walk



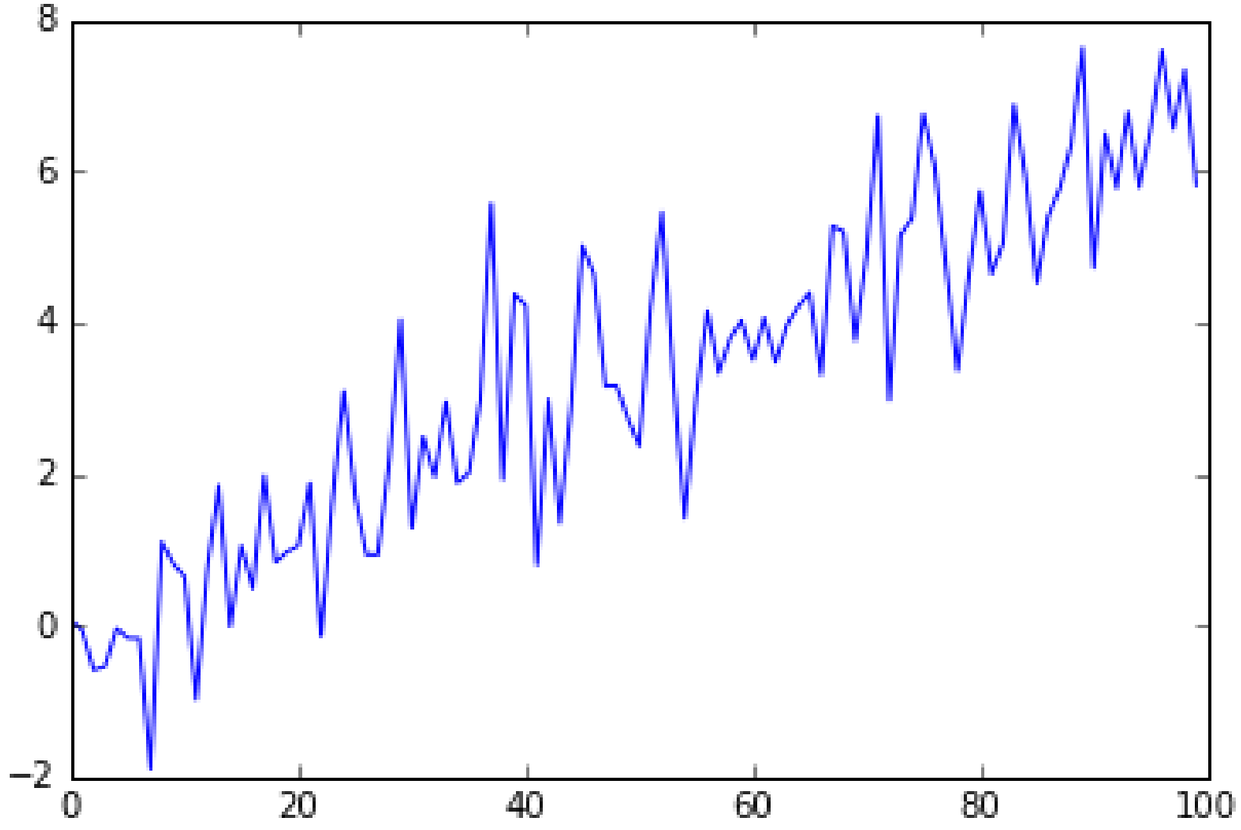
Examples of Nonstationary Series

- Seasonality in series



Examples of Nonstationary Series

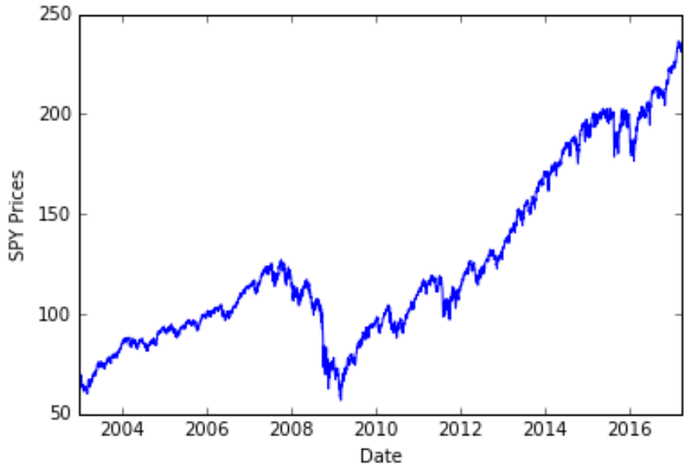
- Change in Mean or Standard Deviation over time



Transforming Nonstationary Series Into Stationary Series

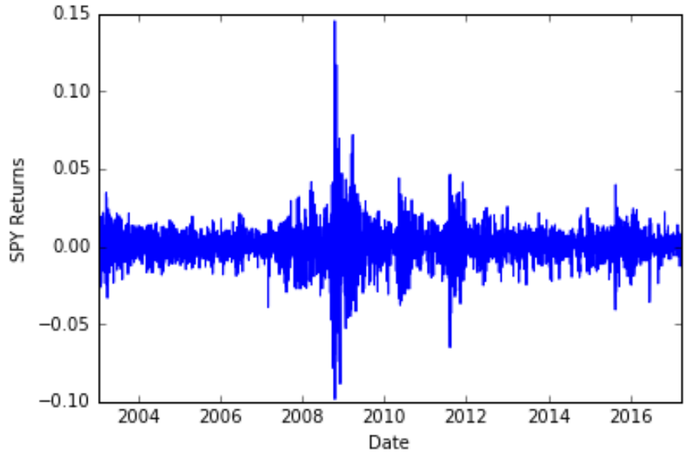
- Random Walk

```
plot.plot(SPY)
```



- First difference

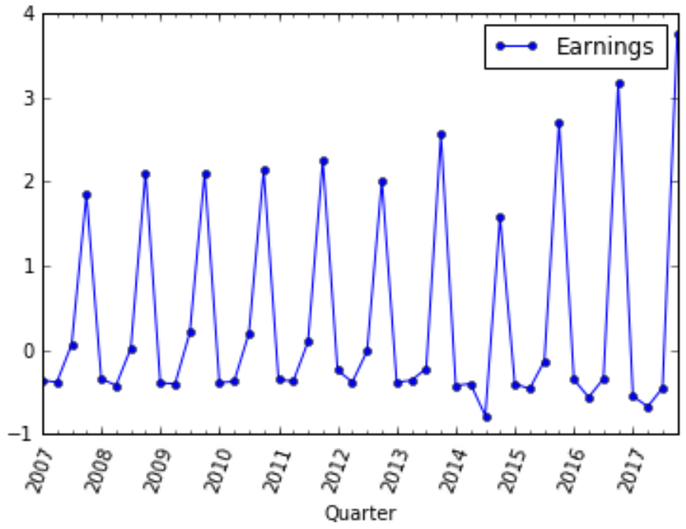
```
plot.plot(SPY.diff())
```



Transforming Nonstationary Series Into Stationary Series

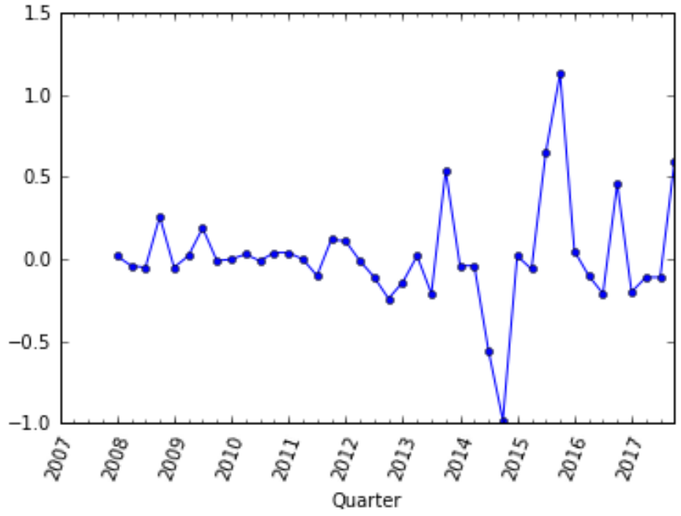
- Seasonality

```
plot.plot(HRB)
```



- Seasonal difference

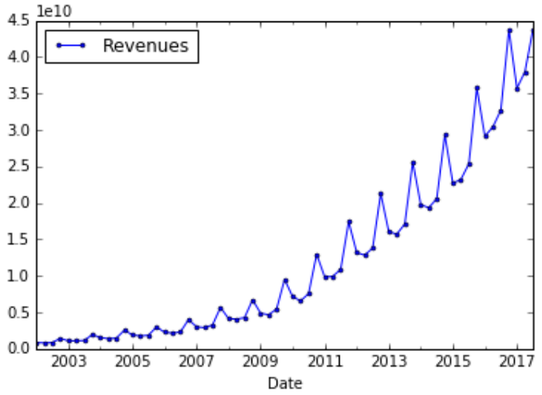
```
plot.plot(HRB.diff(4))
```



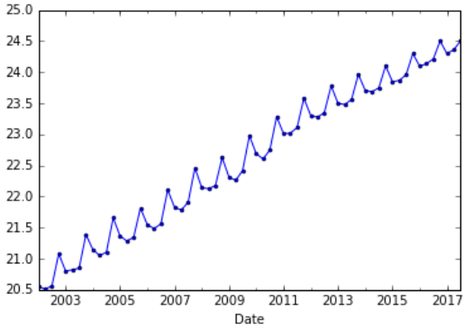
Transforming Nonstationary Series Into Stationary Series

- AMZN Quarterly Revenues

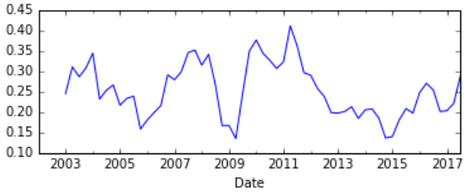
```
plt.plot(AMZN)
```




```
# Log of AMZN Revenues
plt.plot(np.log(AMZN))
```




```
# Log, then seasonal difference
plt.plot(np.log(AMZN).diff(4))
```



Let's practice!

TIME SERIES ANALYSIS IN PYTHON

Introducing an AR Model

TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

datacamp

Mathematical Description of AR(1) Model

$$R_t = \mu + \phi R_{t-1} + \epsilon_t$$

- Since only one lagged value on right hand side, this is called:
  - AR model of order 1, or
  - AR(1) model
- AR parameter is ϕ
- For stationarity, $-1 < \phi < 1$

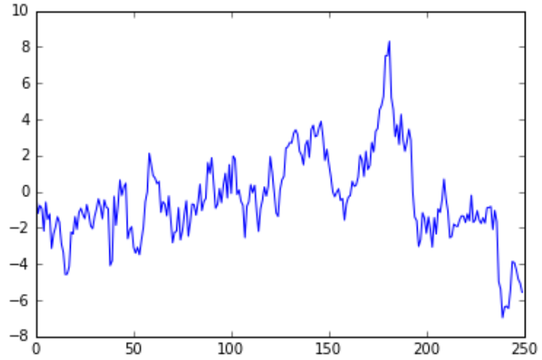
Interpretation of AR(1) Parameter

$$R_t = \mu + \phi R_{t-1} + \epsilon_t$$

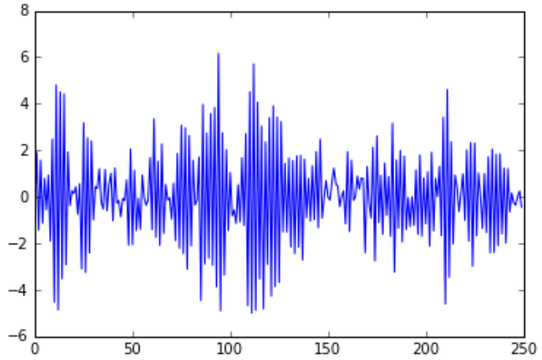
- Negative ϕ: Mean Reversion
- Positive ϕ: Momentum

Comparison of AR(1) Time Series

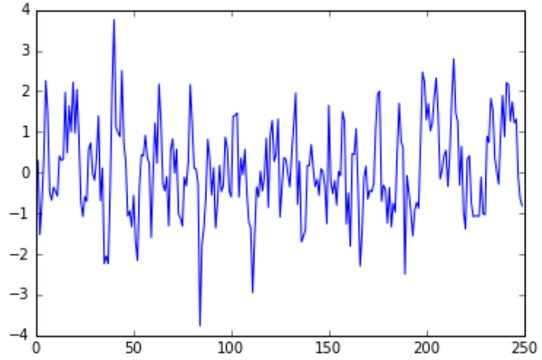
- $\phi = 0.9$



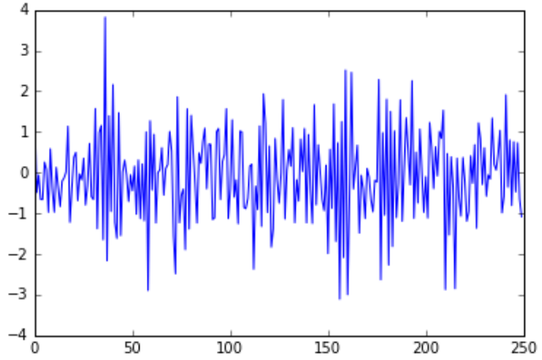
- $\phi = -0.9$



- $\phi = 0.5$

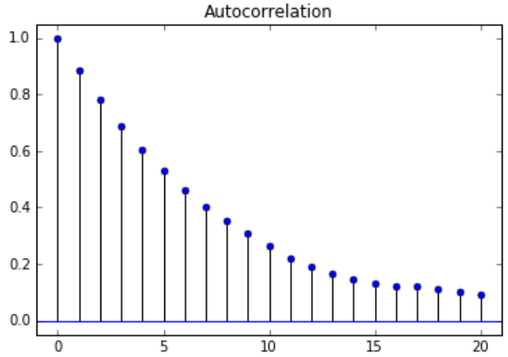


- $\phi = -0.5$

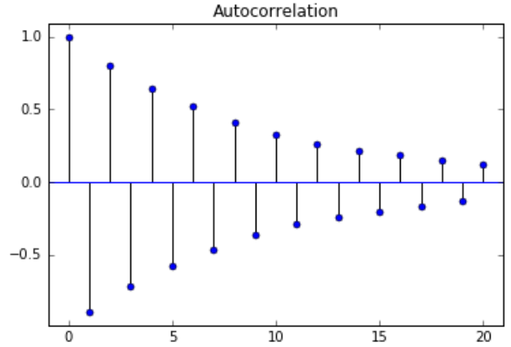


Comparison of AR(1) Autocorrelation Functions

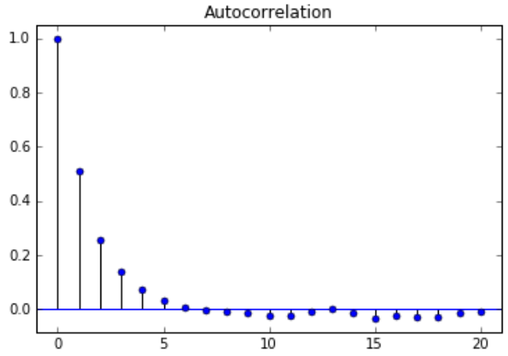
- $\phi = 0.9$



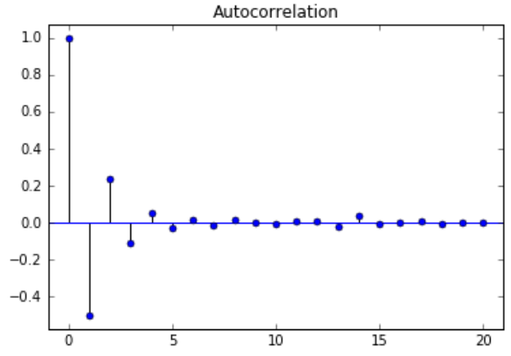
- $\phi = -0.9$



- $\phi = 0.5$



- $\phi = -0.5$



Higher Order AR Models

- AR(1)

$$R_t = \mu + \phi_1 R_{t-1} + \epsilon_t$$

- AR(2)

$$R_t = \mu + \phi_1 R_{t-1} + \phi_2 R_{t-2} + \epsilon_t$$

- AR(3)

$$R_t = \mu + \phi_1 R_{t-1} + \phi_2 R_{t-2} + \phi_3 R_{t-3} + \epsilon_t$$

- ...

Simulating an AR Process

```
from statsmodels.tsa.arima_process import ArmaProcess
ar = np.array([1, -0.9])
ma = np.array([1])
AR_object = ArmaProcess(ar, ma)
simulated_data = AR_object.generate_sample(nsample=1000)
plt.plot(simulated_data)
```

Let's practice!

TIME SERIES ANALYSIS IN PYTHON

Estimating and Forecasting an AR Model

TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

datacamp

Estimating an AR Model

- To estimate parameters from data (simulated)

```
from statsmodels.tsa.arima_model import ARMA
mod = ARMA(simulated_data, order=(1,0))
result = mod.fit()
```

Estimating an AR Model

- Full output (true $\mu = 0$ and $\phi = 0.9$)

```
print(result.summary())
```

```
                              ARMA Model Results
==============================================================================
Dep. Variable:                      y   No. Observations:                 5000
Model:                     ARMA(1, 0)   Log Likelihood               -7178.386
Method:                       css-mle   S.D. of innovations              1.017
Date:                Fri, 01 Dec 2017   AIC                          14362.772
Time:                        15:34:50   BIC                          14382.324
Sample:                             0   HQIC                         14369.625

==============================================================================
                 coef    std err          z      P>|z|      [95.0% Conf. Int.]
------------------------------------------------------------------------------
const         -0.0361      0.152     -0.238      0.812        -0.333     0.261
ar.L1.y        0.9054      0.006    151.020      0.000         0.894     0.917
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
AR.1            1.1045           +0.0000j            1.1045            0.0000
-----------------------------------------------------------------------------
```

Estimating an AR Model

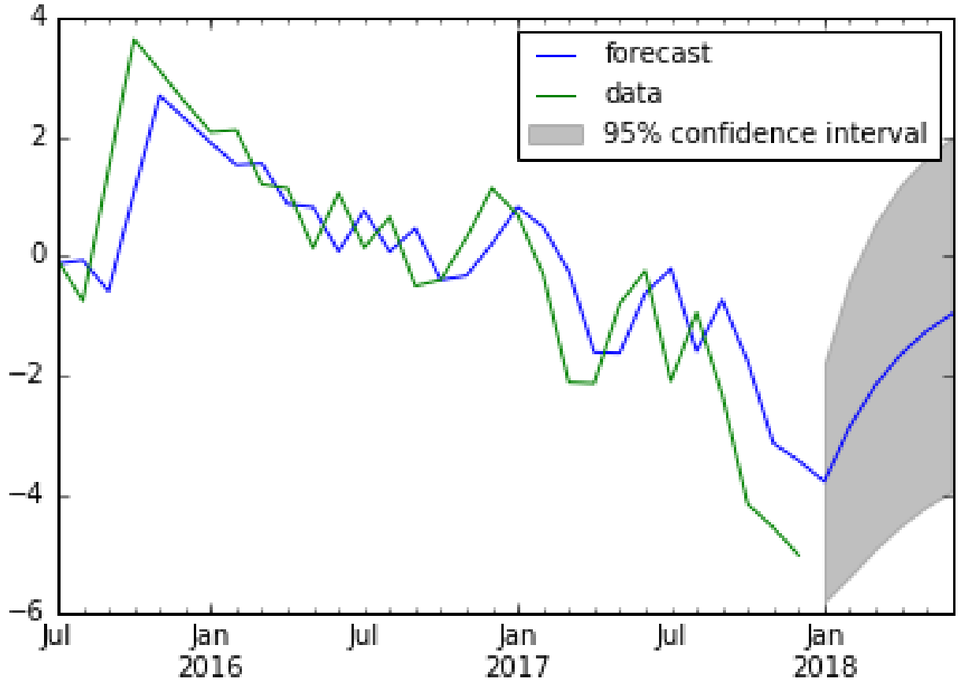
- Only the estimates of μ and ϕ (true $\mu = 0$ and $\phi = 0.9$)

```
print(result.params)
```

```
array([-0.03605989,  0.90535667])
```

Forecasting an AR Model

```
from statsmodels.tsa.arima_model import ARMA
mod = ARMA(simulated_data, order=(1,0))
res = mod.fit()
res.plot_predict(start='2016-07-01', end='2017-06-01')
plt.show()
```



Let's practice!

TIME SERIES ANALYSIS IN PYTHON

Choosing the Right Model

TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

Identifying the Order of an AR Model

- The order of an AR(p) model will usually be unknown
- Two techniques to determine order
  - Partial Autocorrelation Function
  - Information criteria

Partial Autocorrelation Function (PACF)

$$R_t = \phi_{0,1} + \phi_{1,1} R_{t-1} + \epsilon_{1t}$$

$$R_t = \phi_{0,2} + \phi_{1,2} R_{t-1} + \phi_{2,2} R_{t-2} + \epsilon_{2t}$$

$$R_t = \phi_{0,3} + \phi_{1,3} R_{t-1} + \phi_{2,3} R_{t-2} + \phi_{3,3} R_{t-3} + \epsilon_{3t}$$

$$R_t = \phi_{0,4} + \phi_{1,4} R_{t-1} + \phi_{2,4} R_{t-2} + \phi_{3,4} R_{t-3} + \phi_{4,4} R_{t-4} + \epsilon_{4t}$$

$$\vdots$$

Plot PACF in Python

- Same as ACF, but use `plot_pacf` instead of `plt_acf`
- Import module

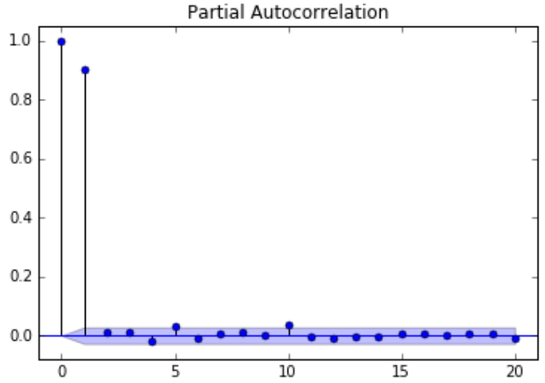
```
from statsmodels.graphics.tsaplots import plot_pacf
```

- Plot the PACF

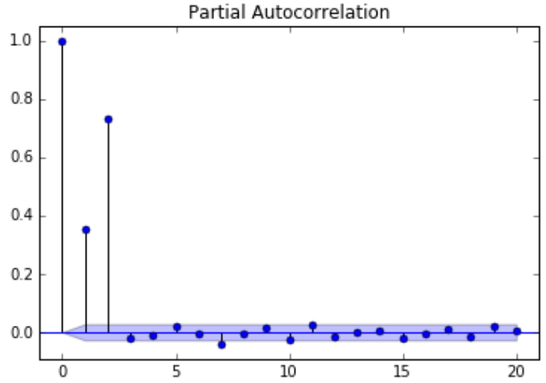
```
plot_pacf(x, lags= 20, alpha=0.05)
```

Comparison of PACF for Different AR Models

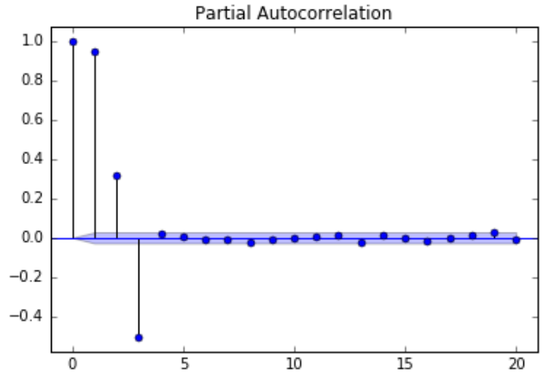
- AR(1)



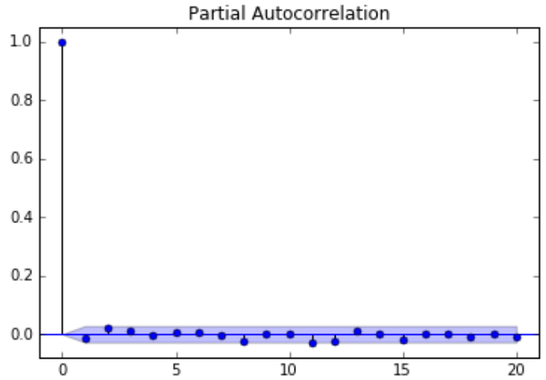
- AR(2)



- AR(3)



- White Noise



Information Criteria

- Information criteria: adjusts goodness-of-fit for number of parameters
- Two popular adjusted goodness-of-fit measures
  - AIC (Akaike Information Criterion)
  - BIC (Bayesian Information Criterion)

Information Criteria

- Estimation output

```
                              ARMA Model Results
==============================================================================
Dep. Variable:                      y   No. Observations:                 2500
Model:                     ARMA(2, 0)   Log Likelihood               -3536.481
Method:                       css-mle   S.D. of innovations              0.996
Date:                Fri, 29 Dec 2017   AIC                           7080.963
Time:                        22:53:24   BIC                           7104.259
Sample:                             0   HQIC                          7089.420

==============================================================================
                 coef    std err          z      P>|z|      [95.0% Conf. Int.]
------------------------------------------------------------------------------
const          0.0054      0.010      0.517      0.605        -0.015     0.026
ar.L1.y       -0.6130      0.019    -32.243      0.000        -0.650    -0.576
ar.L2.y       -0.3109      0.019    -16.351      0.000        -0.348    -0.274
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
AR.1           -0.9859           -1.4982j            1.7935           -0.3426
AR.2           -0.9859           +1.4982j            1.7935            0.3426
-----------------------------------------------------------------------------
```

Getting Information Criteria From `statsmodels`

- You learned earlier how to fit an AR model

```
from statsmodels.tsa.arima_model import ARMA
mod = ARMA(simulated_data, order=(1,0))
result = mod.fit()
```

- And to get full output

```
result.summary()
```

- Or just the parameters

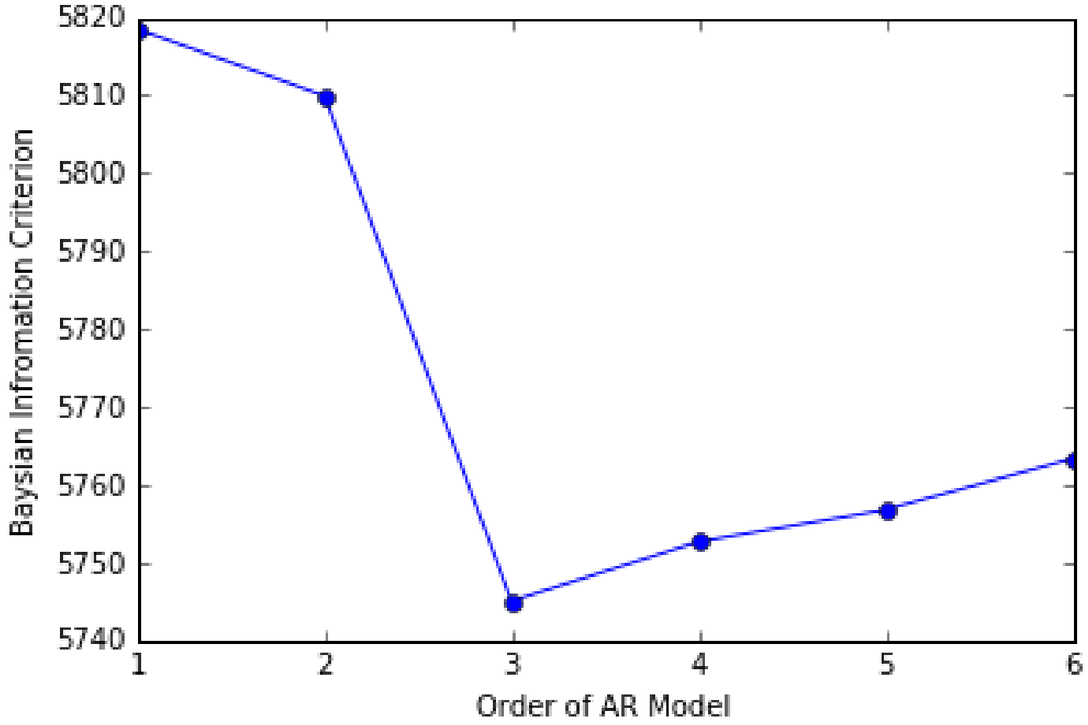
```
result.params
```

- To get the AIC and BIC

```
result.aic
result.bic
```

Information Criteria

- Fit a simulated AR(3) to different AR(p) models
- Choose p with the lowest BIC



Let's practice!

TIME SERIES ANALYSIS IN PYTHON

Describe Model

TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

datacamp

Mathematical Description of MA(1) Model

$$R_t \ equals \ \mu \ + \ \epsilon_t \ + \ \theta \, \epsilon_{t-1}$$

- Since only one lagged error on right hand side, this is called:
  - MA model of order 1, or
  - MA(1) model
- MA parameter is θ
- Stationary for all values of θ

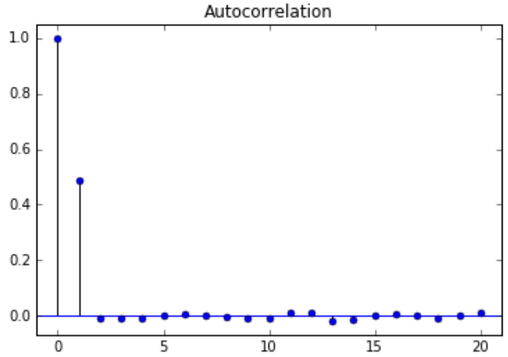
Interpretation of MA(1) Parameter

$$R_t = \mu + \epsilon_t + \theta\,\epsilon_{t-1}$$

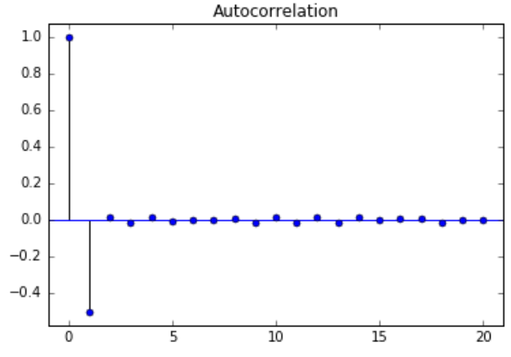
- Negative θ: One-Period Mean Reversion
- Positive θ: One-Period Momentum
- Note: One-period autocorrelation is $\theta/(1+\theta^2)$, not θ

Comparison of MA(1) Autocorrelation Functions

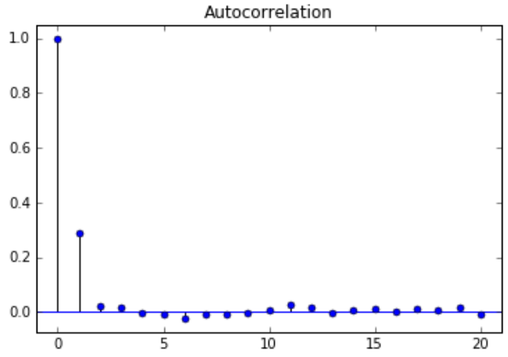
- $\theta = 0.9$



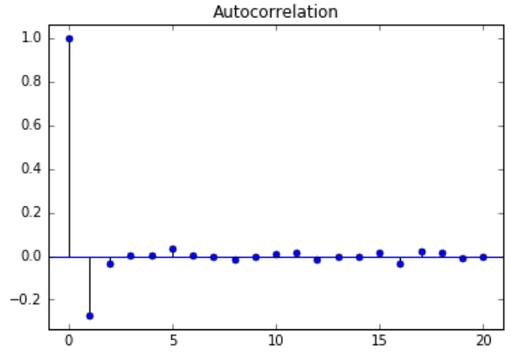
- $\theta = -0.9$



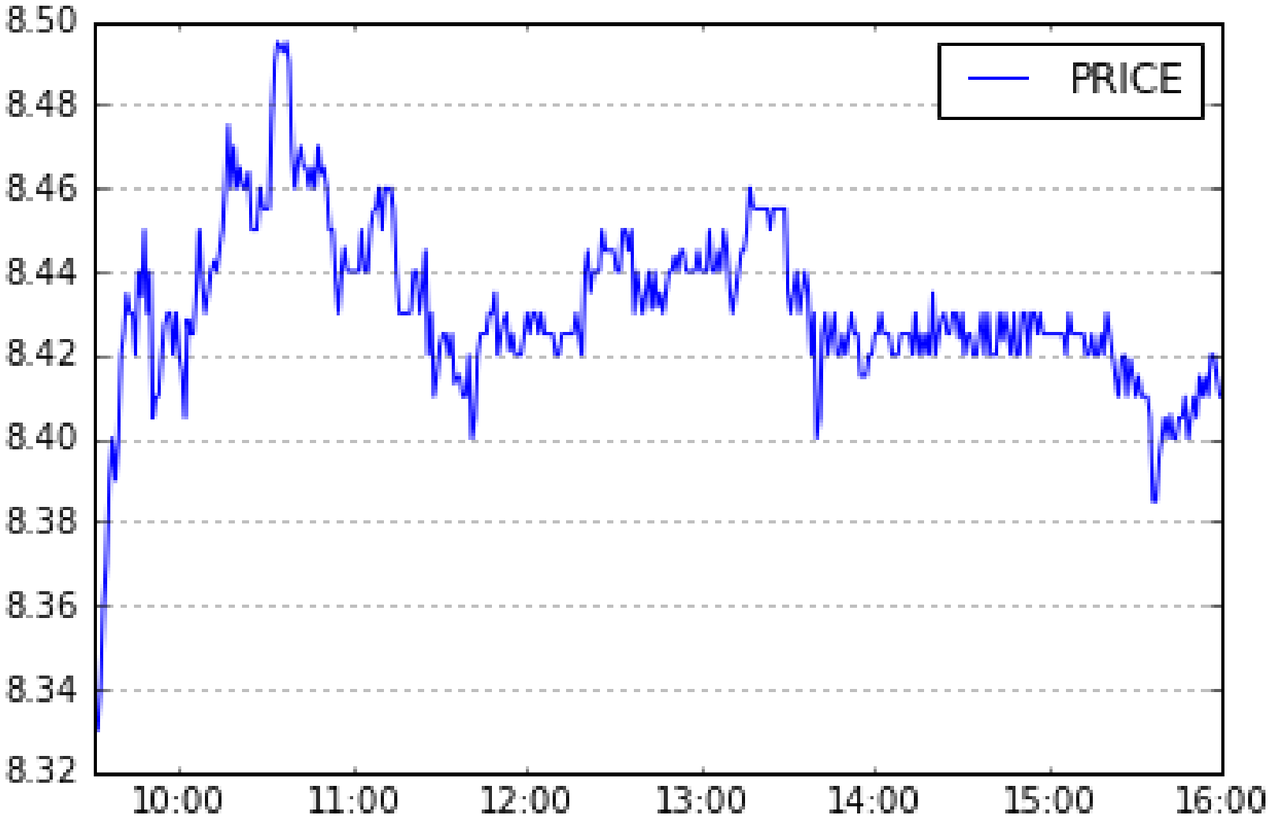
- $\theta = 0.5$



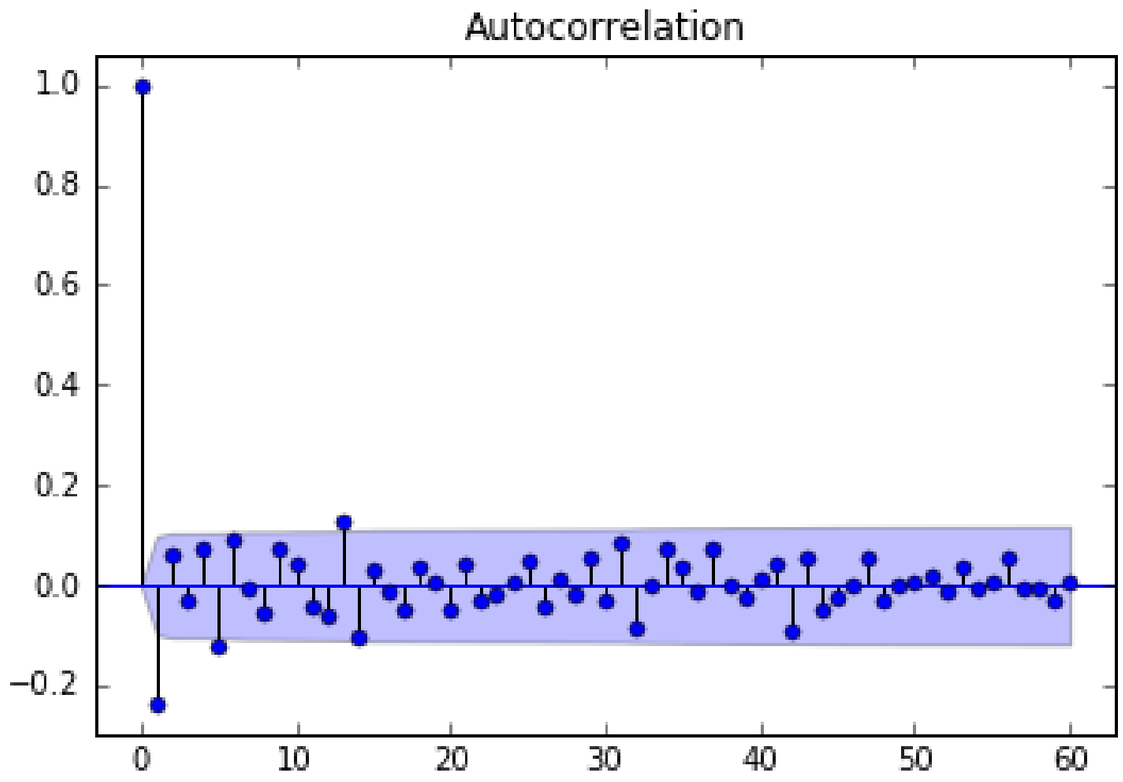
- $\theta = -0.5$



Example of MA(1) Process: Intraday Stock Returns



Autocorrelation Function of Intraday Stock Returns



Higher Order MA Models

- MA(1)

$$R_t = \mu + \epsilon_t - \theta_1 \, \epsilon_{t-1}$$

- MA(2)

$$R_t = \mu + \epsilon_t - \theta_1 \, \epsilon_{t-1} - \theta_2 \, \epsilon_{t-2}$$

- MA(3)

$$R_t = \mu + \epsilon_t - \theta_1 \, \epsilon_{t-1} - \theta_2 \, \epsilon_{t-2} - \theta_3 \, \epsilon_{t-3}$$

- ...

Simulating an MA Process

```
from statsmodels.tsa.arima_process import ArmaProcess
ar = np.array([1])
ma = np.array([1, 0.5])
AR_object = ArmaProcess(ar, ma)
simulated_data = AR_object.generate_sample(nsample=1000)
plt.plot(simulated_data)
```

Let's practice!

TIME SERIES ANALYSIS IN PYTHON

Estimation and Forecasting an MA Model

TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

datacamp

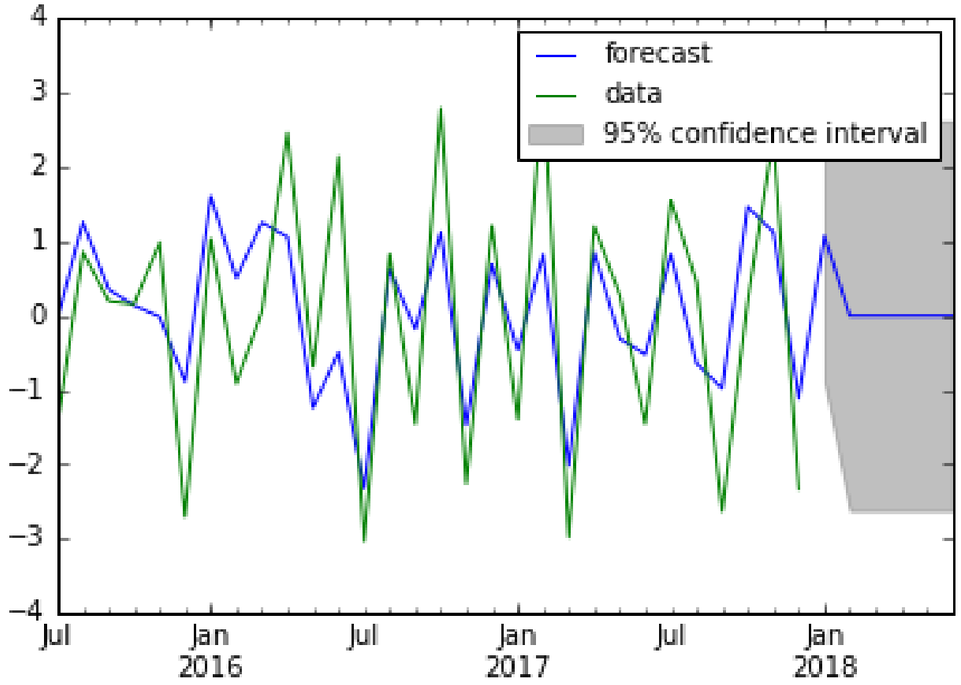
Estimating an MA Model

- Same as estimating an AR model (except `order=(0,1)`)

```
from statsmodels.tsa.arima_model import ARMA
mod = ARMA(simulated_data, order=(0,1))
result = mod.fit()
```

Forecasting an MA Model

```
from statsmodels.tsa.arima_model import ARMA
mod = ARMA(simulated_data, order=(0,1))
res = mod.fit()
res.plot_predict(start='2016-07-01', end='2017-06-01')
plt.show()
```



Let's practice!

TIME SERIES ANALYSIS IN PYTHON

ARMA models

TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

datacamp

ARMA Model

- ARMA(1,1) model:

$$R_t = \mu + \phi R_{t-1} + \epsilon_t + \theta \epsilon_{t-1}$$

Converting Between ARMA, AR, and MA Models

- Converting AR(1) into an MA(infinity)

$$R_t = \mu + \phi R_{t-1} + \epsilon_t$$

$$R_t = \mu + \phi(\mu + \phi R_{t-2} + \epsilon_{t-1}) + \epsilon_t$$

$$\vdots$$

$$R_t = \frac{\mu}{1-\phi} + \epsilon_t + \phi\epsilon_{t-1} - \phi^2\epsilon_{t-2} + \phi^3\epsilon_{t-3} + \ldots$$

Let's practice!

TIME SERIES ANALYSIS IN PYTHON

Cointegration Models

TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

datacamp

What is Cointegration?

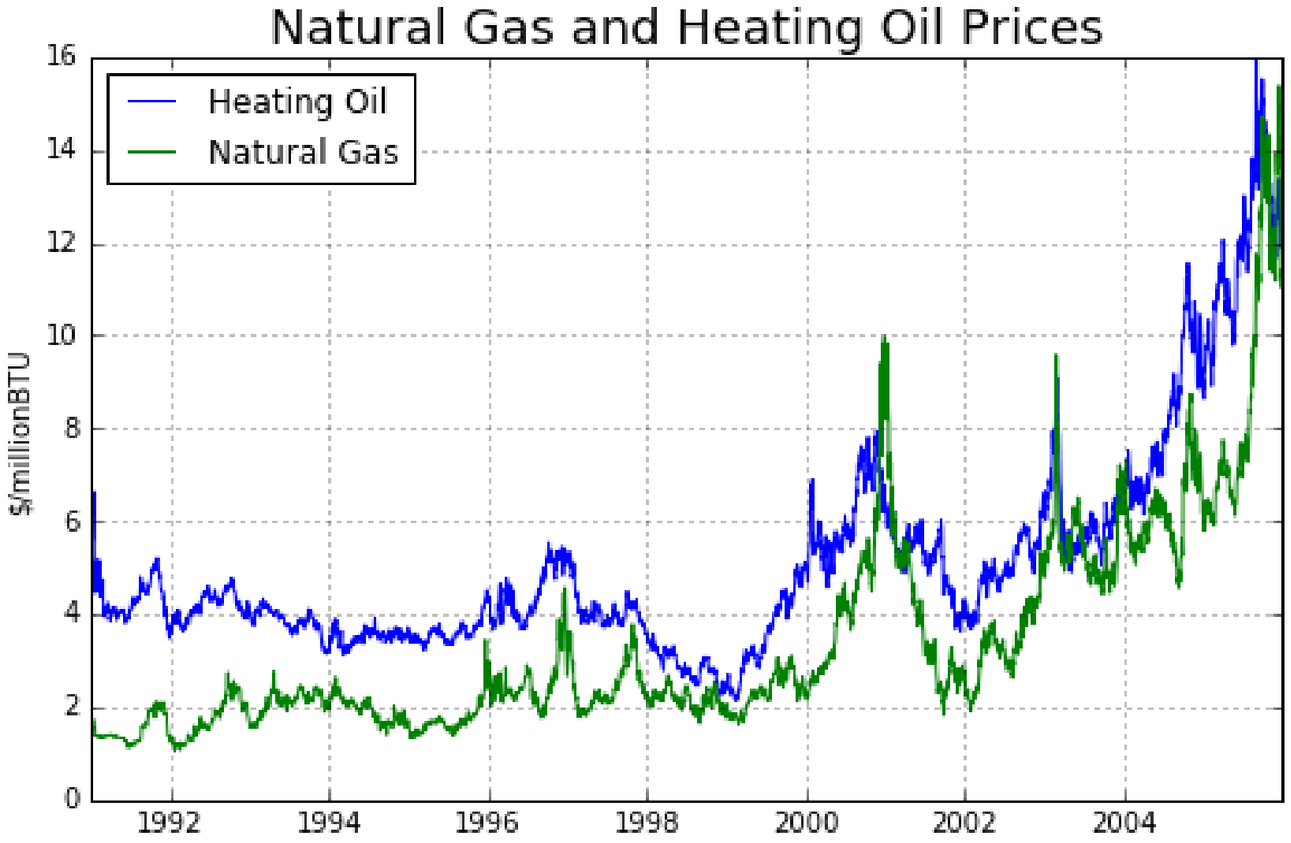
- Two series, P_t and Q_t can be random walks
- But the linear combination $P_t - c\,Q_t$ may not be a random walk!
- If that's true
  - $P_t - c\,Q_t$ is forecastable
  - P_t and Q_t are said to be cointegrated

Analogy: Dog on a Leash

- $P_t =$ Owner
- $Q_t =$ Dog
- Both series look like a random walk
- Difference, or distance between them, looks mean reverting
  - If dog falls too far behind, it gets pulled forward
  - If dog gets too far ahead, it gets pulled back

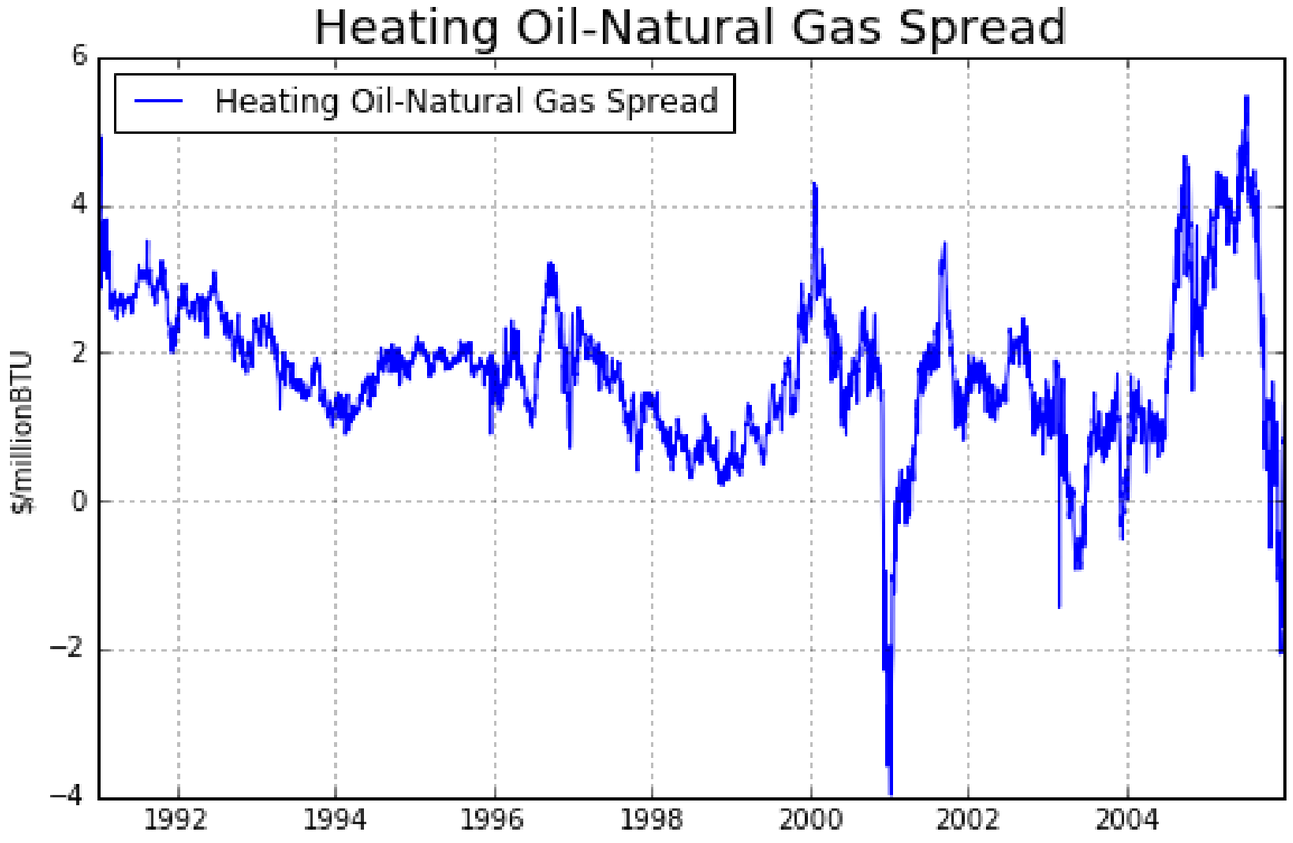
Example: Heating Oil and Natural Gas

- Heating Oil and Natural Gas both look like random walks...



Example: Heating Oil and Natural Gas

- But the spread (difference) is mean reverting



What Types of Series are Cointegrated?

- Economic substitutes
  - Heating Oil and Natural Gas
  - Platinum and Palladium
  - Corn and Wheat
  - Corn and Sugar
  - ...
  - Bitcoin and Ethereum?
- How about competitors?
  - Coke and Pepsi?
  - Apple and Blackberry? No! Leash broke and dog ran away

Two Steps to Test for Cointegration

- Regress P_t on Q_t and get slope c
- Run Augmented Dickey-Fuller test on $P_t - c\,Q_t$ to test for random walk
- Alternatively, can use `coint` function in statsmodels that combines both steps

```
from statsmodels.tsa.stattools import coint
coint(P,Q)
```

Let's practice!

TIME SERIES ANALYSIS IN PYTHON

Case Study: Climate Change

TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

datacamp

Analyzing Temperature Data

- Temperature data:
  - New York City from 1870-2016
  - Downloaded from National Oceanic and Atmospheric Administration (NOAA)
- Convert index to datetime object
- Plot data

Analyzing Temperature Data

- Test for Random Walk
- Take first differences
- Compute ACF and PACF
- Fit a few AR, MA, and ARMA models
- Use Information Criterion to choose best model
- Forecast temperature over next 30 years

Let's practice!

TIME SERIES ANALYSIS IN PYTHON

Congratulations

TIME SERIES ANALYSIS IN PYTHON

Rob Reider
Adjunct Professor, NYU-Courant
Consultant, Quantopian

Advanced Topics

- GARCH Models
- Nonlinear Models
- Multivariate Time Series Models
- Regime Switching Models
- State Space Models and Kalman Filtering
- ...

Keep practicing!

TIME SERIES ANALYSIS IN PYTHON