

Significance testing of model parameters

GARCH MODELS IN PYTHON



Chelsea Yang
Data Science Instructor

Do I need this parameter?

- Is it relevant
- KISS: keep it simple stupid

THE KISS PRINCIPLE | **KEEP
IT
SIMPLE,
STUPID**

- Always prefer a parsimonious model

Hypothesis test

- Null hypothesis (H_0): a claim to be verified
- H_0 : parameter value = 0
- If H_0 cannot be rejected, leave out the parameter

Statistical significance

- Quantify having the observed results by chance
- Common threshold: 5%

P-value

- The odds of the observed results could have happened by chance
- The lower the p-value, the more ridiculous the null hypothesis looks
- Reject the null hypothesis if $p\text{-value} < \text{significance level}$

P-value example

```
print(gm_result.summary())
```

```
=====
                        Mean Model
=====
      coef  std err          t      P>|t|      95.0% Conf. Int.
-----
mu      0.0772  1.445e-02     5.345  9.031e-08 [4.892e-02, 0.106]
=====
                        Volatility Model
=====
      coef  std err          t      P>|t|      95.0% Conf. Int.
-----
omega    0.0396  9.181e-03     4.312  1.619e-05 [2.159e-02, 5.758e-02]
alpha[1] 0.1680  2.690e-02     6.243  4.284e-10 [ 0.115, 0.221]
beta[1]   0.7865  2.722e-02    28.897  1.303e-183 [ 0.733, 0.840]
=====
```

```
print(gm_result.pvalues)
```

```
mu          9.031206e-08
omega       1.619415e-05
alpha[1]    4.283526e-10
beta[1]     1.302531e-183
Name: pvalues, dtype: float64
```

T-statistic

- T-statistic = estimated parameter / standard error
- The absolute value of the t-statistic is a distance measure
- If $|t\text{-statistic}| > 2$: keep the parameter in the GARCH model

T-statistic example

```
print(gm_result.summary())
```

```
=====
                    Mean Model
=====
              coef  std err          t      P>|t|     95.0% Conf. Int.
-----
mu              0.0772  1.445e-02      5.345  9.031e-08 [4.892e-02,  0.106]
=====
                    Volatility Model
=====
              coef  std err          t      P>|t|     95.0% Conf. Int.
-----
omega           0.0396  9.181e-03      4.312  1.619e-05 [2.159e-02,5.758e-02]
alpha[1]        0.1680  2.690e-02      6.243  4.284e-10 [ 0.115,  0.221]
beta[1]         0.7865  2.722e-02     28.897  .303e-183 [ 0.733,  0.840]
=====
```

```
print(gm_result.tvalues)
```

```
mu              5.345210
omega           4.311785
alpha[1]        6.243330
beta[1]         28.896991
Name: tvalues, dtype: float64
```

```
# Manual calculation
```

```
t = gm_result.std_err/gm_result.tvalues
```


Let's practice!

GARCH MODELS IN PYTHON

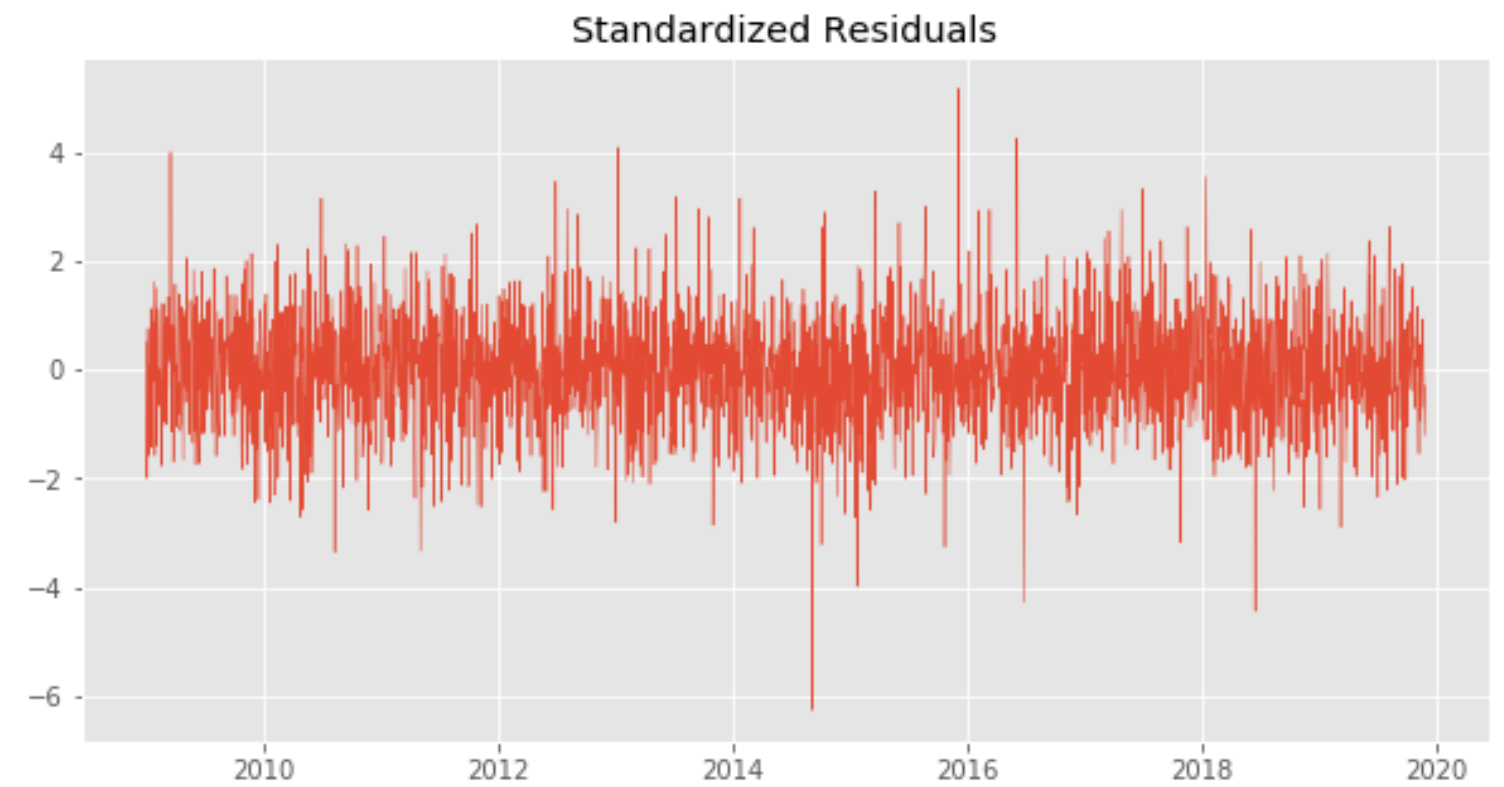
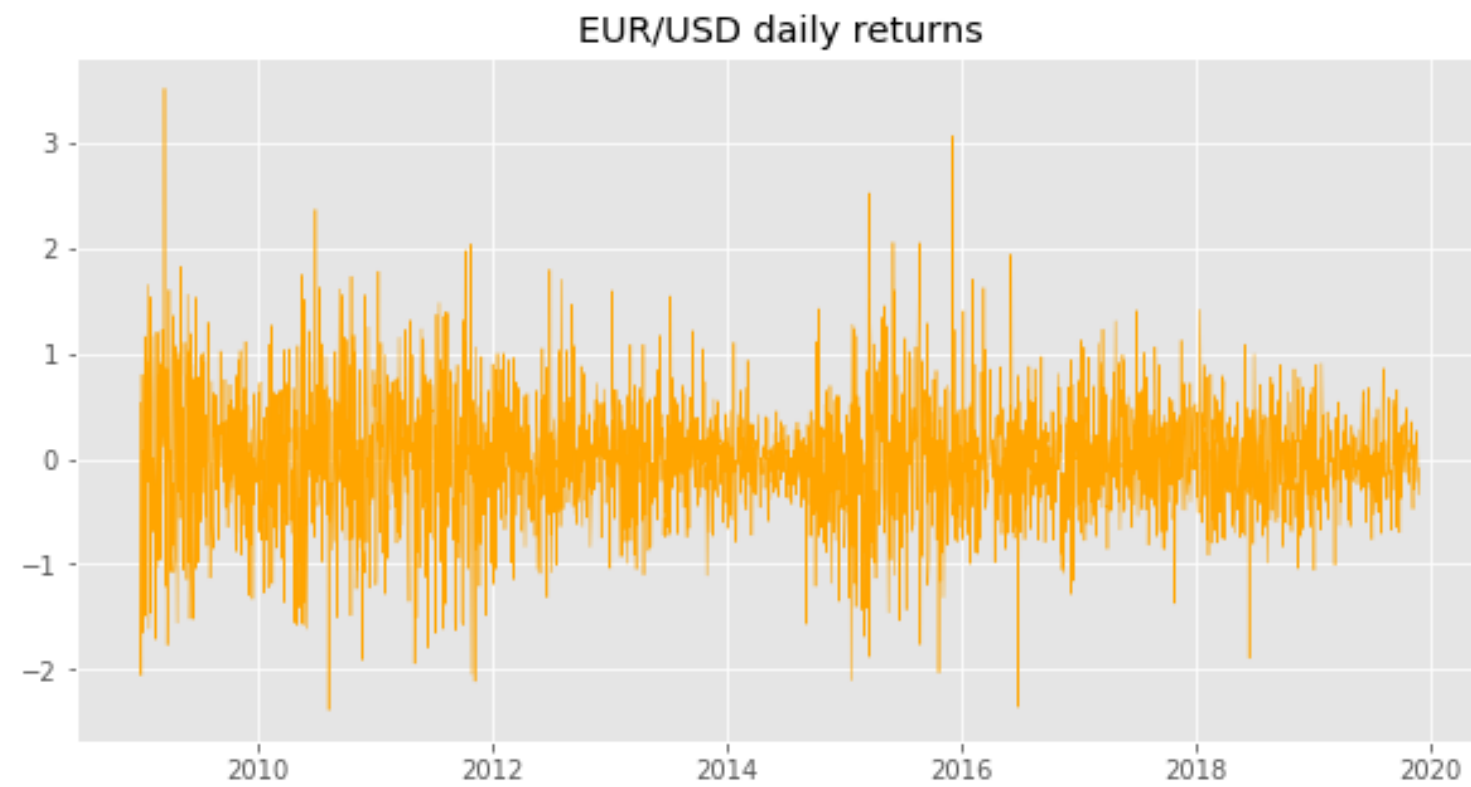
Validation of GARCH model assumptions

GARCH MODELS IN PYTHON



Chelsea Yang
Data Science Instructor

Visual check



Autocorrelation

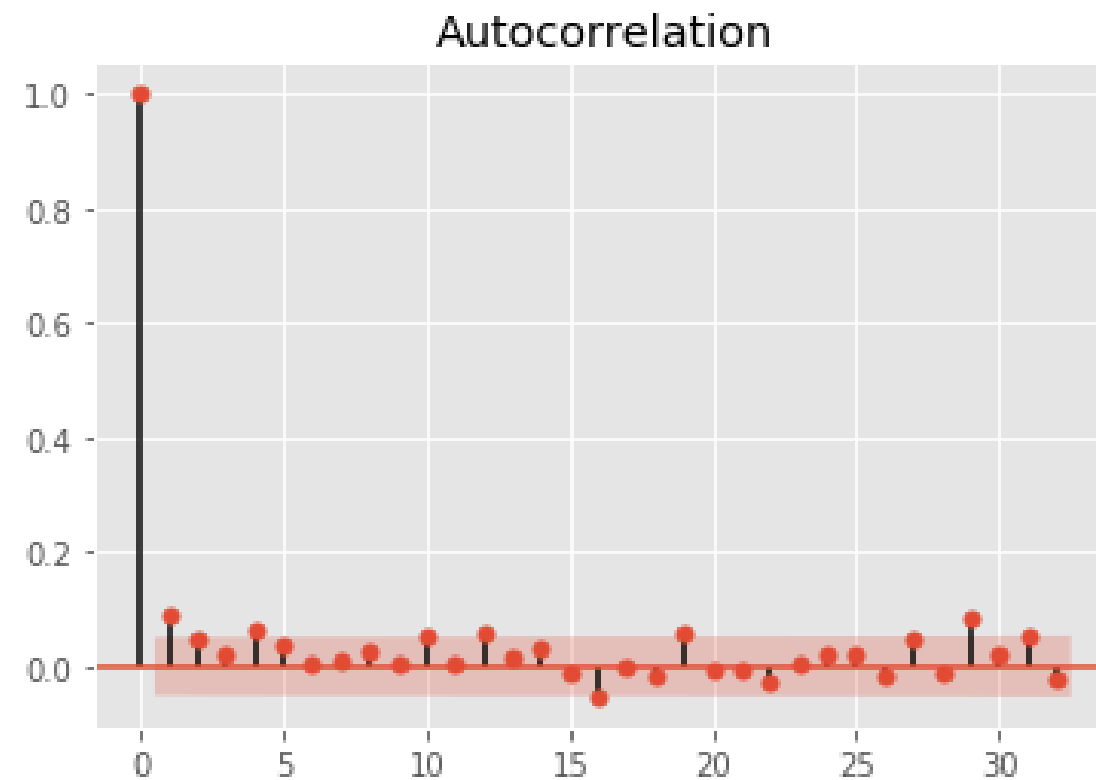
- Describe the correlation of a variable with itself given a time lag
- Existence of autocorrelation in the standardized residuals indicates the model may not be sound

To detect autocorrelation:

- ACF plot
- Ljung-Box

ACF plot

- ACF: AutoCorrelation Function
- ACF Plot: visual representation of the autocorrelation by lags



Red area in the plot indicates the confidence level ($\alpha = 5\%$)

ACF plot in Python

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

```
plot_acf(my_data, alpha = 0.05)
```

Ljung-Box test

- Test whether any of a group of autocorrelations of a time series are different from zero
- H_0 : the data is independently distributed
- $P\text{-value} < 5\%$: the model is not sound

Ljung-Box test Python

```
# Import the Python module  
from statsmodels.stats.diagnostic import acorr_ljungbox
```

```
# Perform the Ljung-Box test  
lb_test = acorr_ljungbox(std_resid , lags = 10)
```

```
# Check p-values  
print('P-values are: ', lb_test[1])
```


Let's practice!

GARCH MODELS IN PYTHON

Goodness of fit measures

GARCH MODELS IN PYTHON



Chelsea Yang
Data Science Instructor

Goodness of fit

Can model do a good job explaining the data?

1. Maximum likelihood
2. Information criteria

Maximum likelihood

- Maximize the probability of getting the data observed under the assumed model
- Prefer models with larger likelihood values

Log-likelihood in Python

- Typically used in log form: log-likelihood

```

                        Constant Mean - GARCH Model Results
=====
Dep. Variable:          Return    R-squared:                -0.001
Mean Model:            Constant Mean  Adj. R-squared:           -0.001
Vol Model:             GARCH        Log-Likelihood:          -3966.27
Distribution:          Standardized Student's t    AIC:                    7942.53
Method:                Maximum Likelihood       BIC:                    7969.04
                                                                No. Observations:       1483
                                                                Df Residuals:           1478
                                                                Df Model:                5
Date:                  Thu, Jan 09 2020
Time:                  00:21:27
```

```
print(gm_result.loglikelihood)
```

Overfitting

- Fit in-sample data well, but perform poorly on out-of-sample predictions
- Usually due to the model is overly complex

Information criteria

- Measure the trade-off between goodness of fit and model complexity
- Likelihood + penalty for model complexity
- AIC: Akaike's Information Criterion
- BIC: Bayesian Information Criterion

Prefer models with the lower information criterion score

AIC vs. BIC

- Generally they agree with each other
- BIC penalizes model complexity more severely

AIC/BIC in Python

```

                                Constant Mean - GARCH Model Results
=====
Dep. Variable:                  Return    R-squared:                  -0.001
Mean Model:                    Constant Mean  Adj. R-squared:             -0.001
Vol Model:                     GARCH        Log-Likelihood:            -3966.27
Distribution:                  Standardized Student's t
Method:                        Maximum Likelihood
                                AIC:                  7942.53
                                BIC:                  7969.04
                                No. Observations:      1483
Date:                          Thu, Jan 09 2020      Df Residuals:              1478
Time:                          00:21:27             Df Model:                  5
                                Mean Model

```

```
print(gm_result.aic)
print(gm_result.bic)
```

Let's practice!

GARCH MODELS IN PYTHON

GARCH model backtesting

GARCH MODELS IN PYTHON



Chelsea Yang
Data Science Instructor

Backtesting

- An approach to evaluate model forecasting capability
- Compare the model predictions with the actual historical data

In-sample vs. out-of-sample

- In-sample: model fitting
- Out-of-sample: backtesting

MAE

Mean Absolute Error

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$$

MSE

Mean Squared Error

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

Calculate MAE, MSE in Python

```
from sklearn.metrics import mean_absolute_error, mean_squared_error
```

```
# Call function to calculate MAE  
mae = mean_absolute_error(observation, forecast)
```

```
# Call function to calculate MSE  
mse = mean_squared_error(observation, forecast)
```


Let's practice!

GARCH MODELS IN PYTHON