Fintech545 HW1
Date: 01/27/2023
Name: Long Zhang

Problem 1

The main idea is calculate skewness and kurtosis function's p-value respectively and make a comparison with threshold which is set as 0.05.
Steps:
1.Sample 10 random normal values.
2.Calculate skewness and kurtosis via scipy.stats.skew() and scipy.stats.kurtosis().
3.Repeat step1 and step2 100 times.
4.Calculate p-values and compare them with 0.05. If p-value<0.05, then biased; If p-value>0.05 unbiased

Following the output:

```
skewness p value: 0.3752809359948356
skewness function unbiased
kurtosis p value: 4.501117922614715e-11
kurtosis function biased
```

Since skewness p-value>0.05 then it is unbiased and kurtosis p-value<0.05 then biased

Problem 2

For OLS fitting, import the data apply sm.OLS(y, x).fit() method in package statsmodels.api and collect the summary of OLS fitting:

Dep. Variable:	y	**R-squared:**	0.195
Model:	OLS	**Adj. R-squared:**	0.186
Method:	Least Squares	**F-statistic:**	23.68
Date:	Sat, 28 Jan 2023	**Prob (F-statistic):**	4.34e-06
Time:	05:06:41	**Log-Likelihood:**	-159.99
No. Observations:	100	**AIC:**	324.0
Df Residuals:	98	**BIC:**	329.2
Df Model:	1		
Covariance Type:	nonrobust		

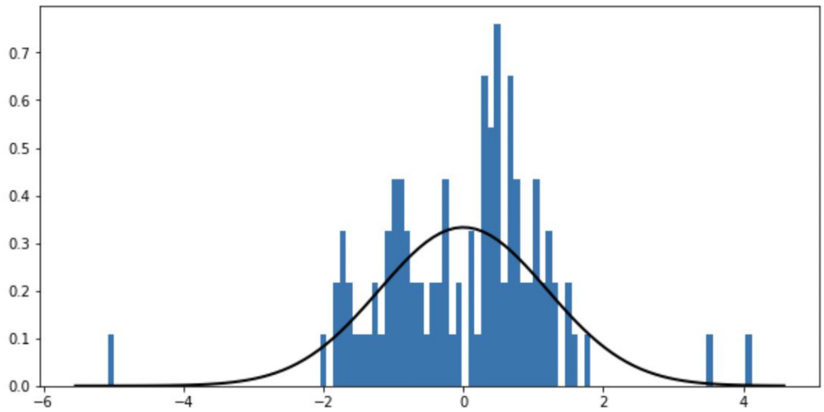
	coef	std err	t	P>|t|	[0.025	0.975]
const	0.1198	0.121	0.990	0.325	-0.120	0.360
x1	0.6052	0.124	4.867	0.000	0.358	0.852

Omnibus:	14.146	**Durbin-Watson:**	1.885
Prob(Omnibus):	0.001	**Jarque-Bera (JB):**	43.673
Skew:	-0.267	**Prob(JB):**	3.28e-10
Kurtosis:	6.193	**Cond. No.**	1.03

The error vectors calculated by sm.ols.resid is shown below:

```
[-0.83848479  0.83529586  1.02742825  1.3197107  -0.1523166  -0.3
8641696
  1.28474611  0.6785721  -0.23279104  0.68498605  0.90479441  1.0
3882326
  0.88188173  0.14094188  0.59443017  0.71760455  0.36758746 -0.3
89435
  4.12403686 -0.05680601  0.66842671 -0.98837595 -1.31557297  0.2
6537682
  0.41153462  0.7788615  -1.84465372  1.06907408  1.82068861 -0.9
8639189
 -0.75239421 -1.01950983  0.48915464 -1.6436499  -0.2732364   1.1
8787117
  0.97341581  0.13851152  0.41529646  1.12914889  0.31369632 -0.7
8483505
  0.2665901   0.50569968 -1.67738413  0.65902192 -0.25881239 -1.9
9793919
 -0.64026358  1.52109106 -0.92685988 -1.71158989  0.63461011  0.5
0398216
 -0.36865304  0.08488123 -1.05294004 -5.08389235 -0.59820773  1.1
6069069
  1.62901979  0.52427467 -0.04299272  0.57525757 -1.46693675  1.5
4281348
  0.25996545 -1.27897259  0.30440434 -0.98989937  0.2006473  -1.2
6898348
  0.68496909 -0.2821325  -1.11770849  0.73021764 -1.20161542  1.2
6304551
  0.46058222 -0.78173218  3.53168002  1.17877991  1.00198234 -0.5
1711683
  0.74937184 -0.89077524  0.47792631 -0.67167181 -0.47687562  0.3
3442978
 -1.77706214 -0.85417328 -1.52321668  0.51743109 -1.07169233 -1.5
9026377
 -1.69484815  0.43487766  0.40226118 -0.92231882]
```

To check the goodness of fit, we plot the histogram and normal distribution curve to observe the shape of them. Moreover, calculate the 4 moments of error to precisely analyze the goodness of fit.




```
Mean of error: -1.2212453270876722e-17
Variance of error:1.1983941277418964
Skewness of error:-0.26726658552879606
Kurtosis of error:3.1931010009568777
```

As we can see, the two graph have relatively big differences. The bin graph arrive maximum at around 1 which is far from 0. Also the bin graph is asymmetric. Observing the 4 moments, all of them have great differences with normal distribution. Thus, it does not fit the normally distributed errors.

To compare the three model's goodness of fit, we using R square. Since we have already get the R square for OLS in the OLS regression summary, we now just need to compute two MLE simulation 's R square.

For calculating R square for MLE, apply the algorithm for R square:

$$SS_{total} = \sum_{i=1}^{n} \left(y_i - \bar{y}\right)^2$$

$$SS_{error} = \sum_{i=1}^{n} \left(\epsilon_i - \bar{\epsilon}\right)^2 = \sum_{i=1}^{n} \epsilon_i^2$$

$$SS_{model} = SS_{total} - SS_{error}$$

$$R^2 = \frac{SS_{model}}{SS_{total}} = 1 - \frac{SS_{error}}{SS_{total}}$$

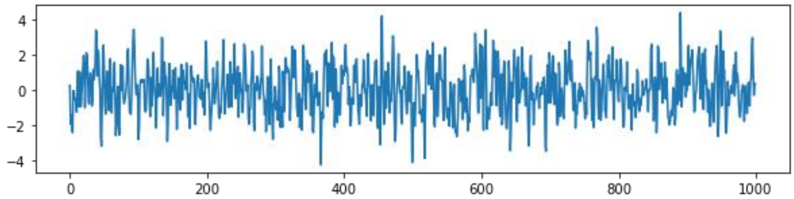
Find the prediction and error part by the MLE and compute the R square. Output shown below:

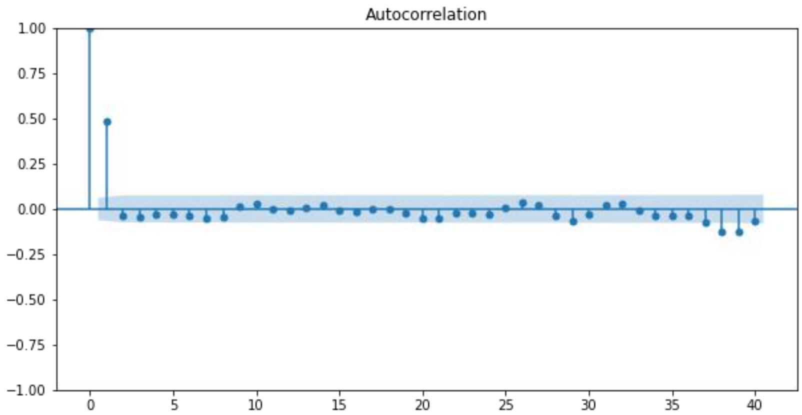
	OLS	MLE normal	MLE t
R square	0.195	0.1946	0.1935

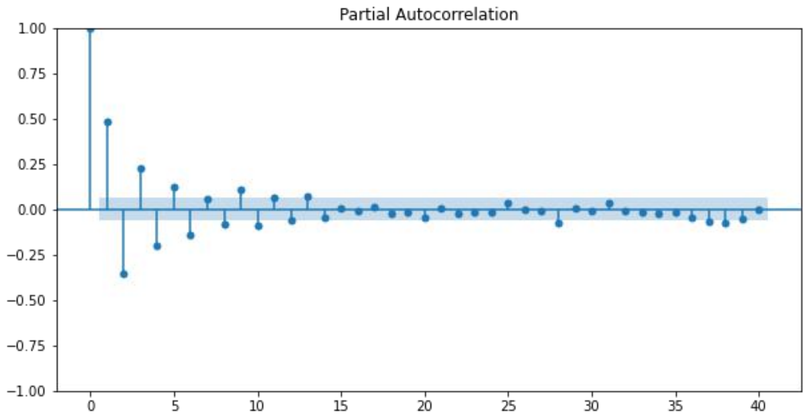
Apparently, MLE with t distribution has the smallest R square value. Thus the MLE t distribution fit the best.

Problem 3

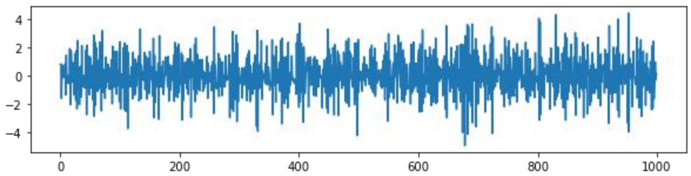
AR(1)

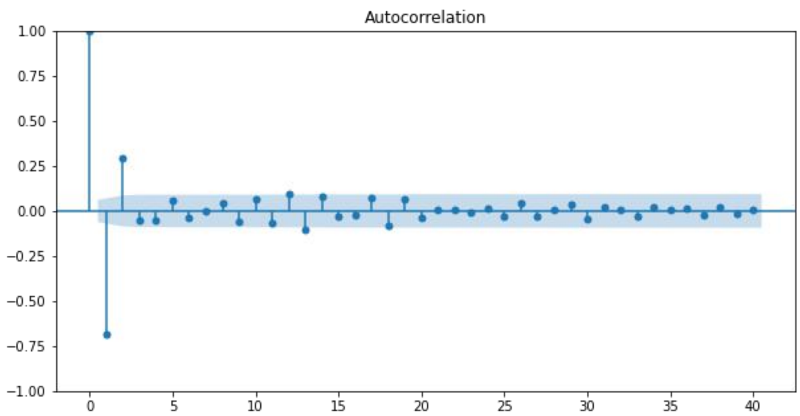


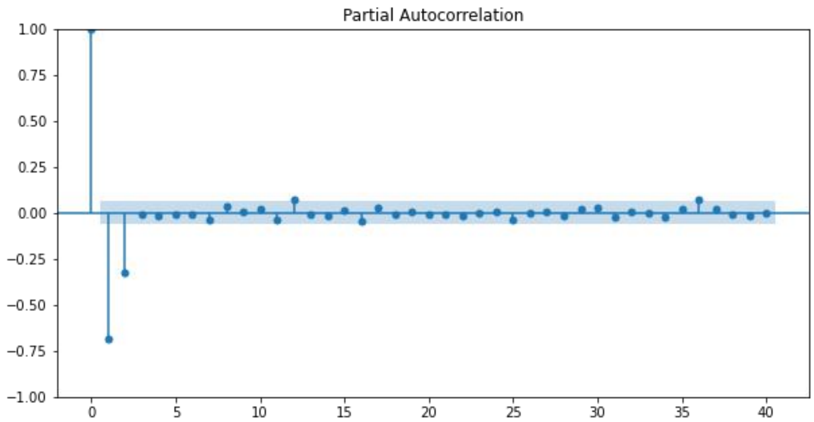

Autocorrelation


Partial Autocorrelation

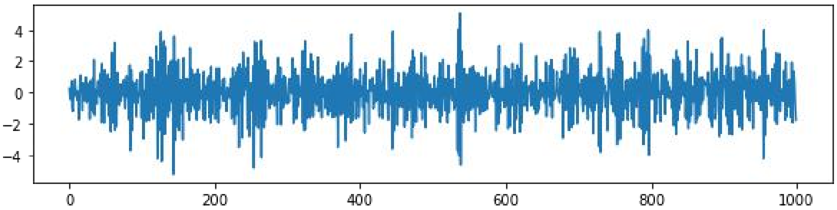
AR(2)

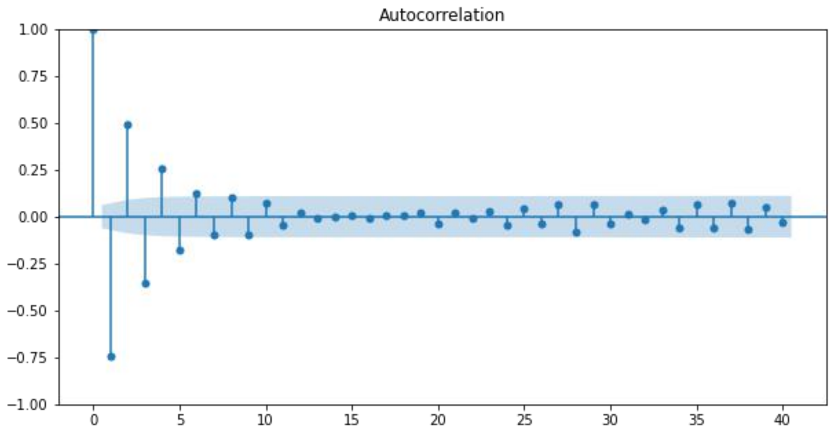


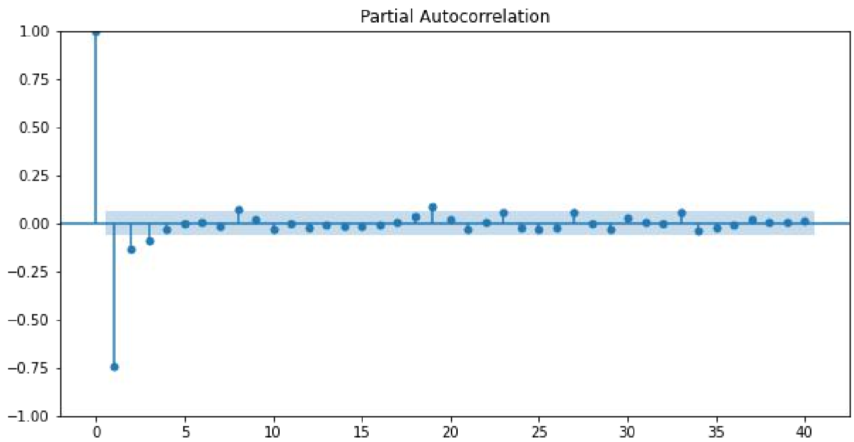

Autocorrelation


Partial Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35
40

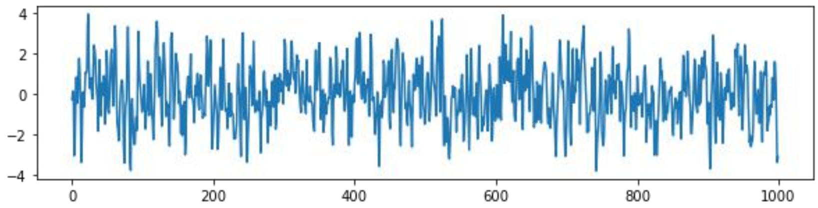
AR(3)

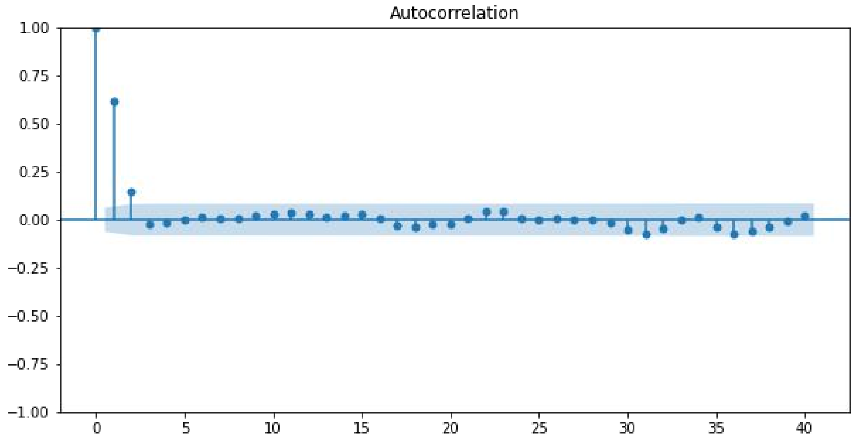

4
2
0
−2
−4
0
200
400
600
800
1000

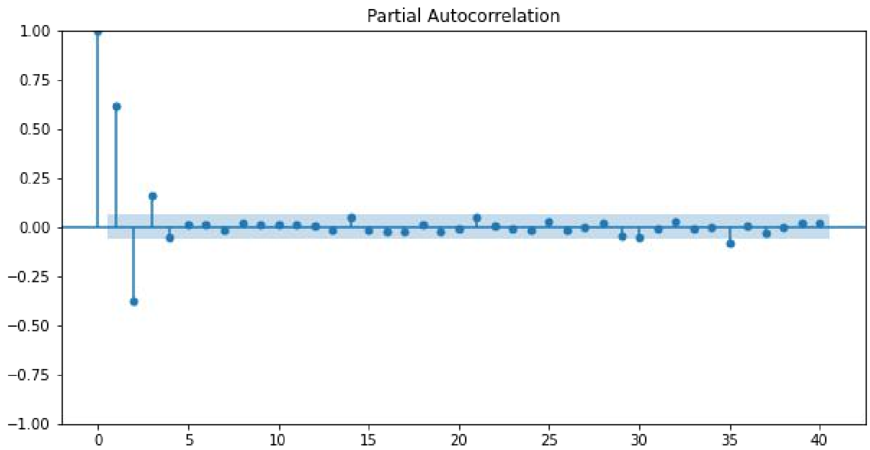

Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35
40


Partial Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35
40

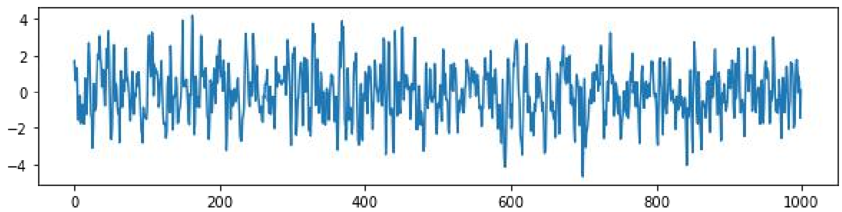
MA(1)

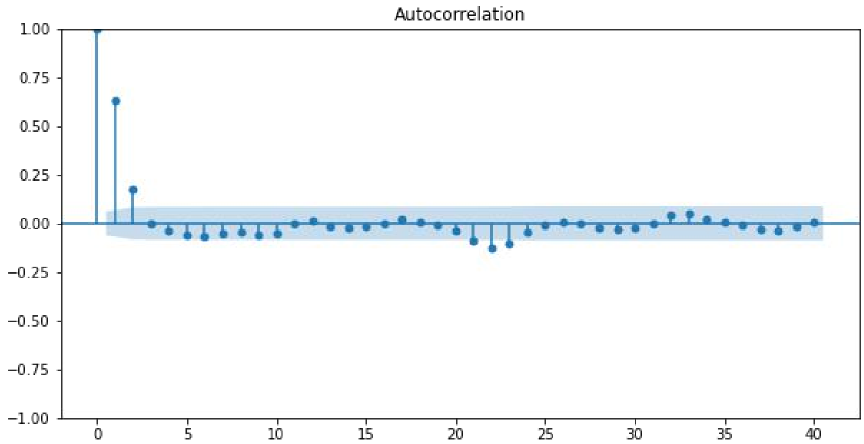

4
2
0
−2
−4
0
200
400
600
800
1000

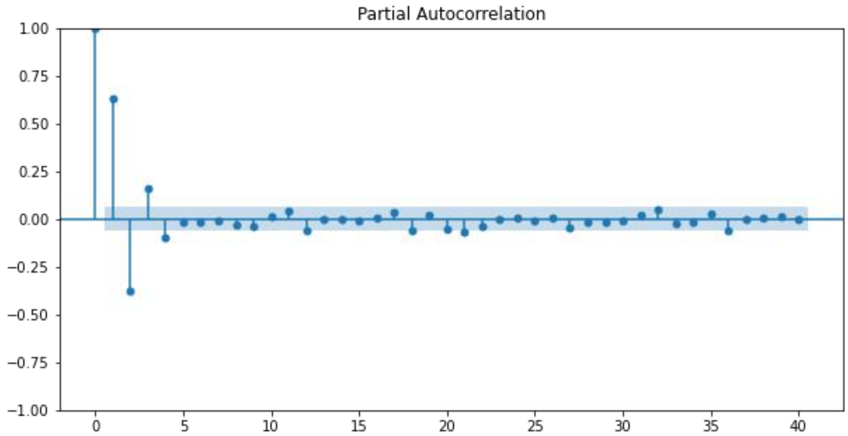

Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35
40


Partial Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35
40

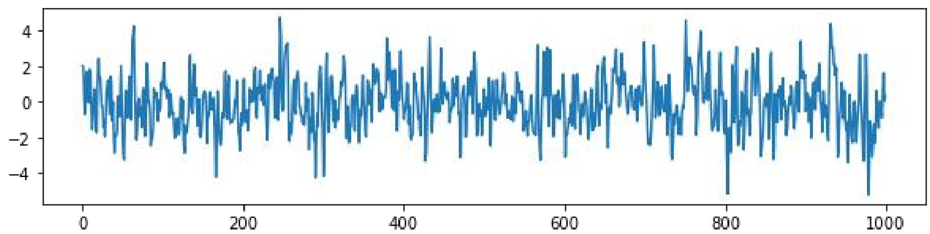
MA(2)

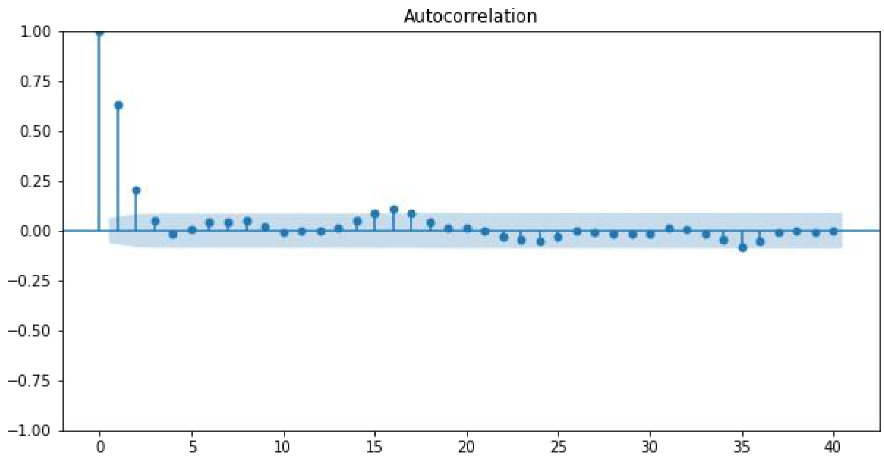

4
2
0
−2
−4
0
200
400
600
800
1000

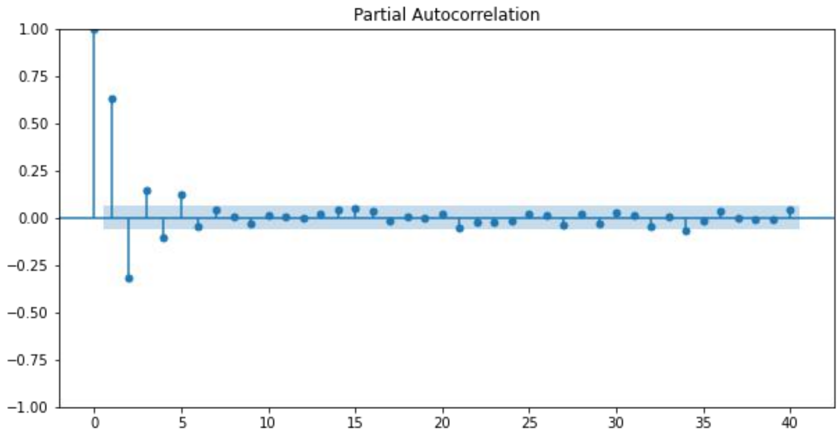

Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35
40


Partial Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35
40

MA(3)


4
2
0
−2
−4
0
200
400
600
800
1000


Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35
40



By observing the graph, for AR model, if tail off at ACF and cut off at PACF, we can determine the order p, AR(p). for MA model, if tail off at PACF and cut off at ACF, we can determine the order q, MA(q)