FINTECH 545 Week02 Project Report

Yilun Wu(yw528)

Problem 1:

For this problem, I used *skew()* function for computing skewness and *kurtosis()* function for computing kurtosis in the *scipy.stats* package module in python. The two functions **are both statistically significantly biased for small samples**, yet they are not statistically significantly biased for large samples with size greater than 100.

In this test, the null hypothesis (H_0) is that the mean values of skewness and kurtosis computed by corresponding functions are equal to 0, and the alternative hypothesis (H_a) is that the mean values are not equal to 0. The test is performed as follows:

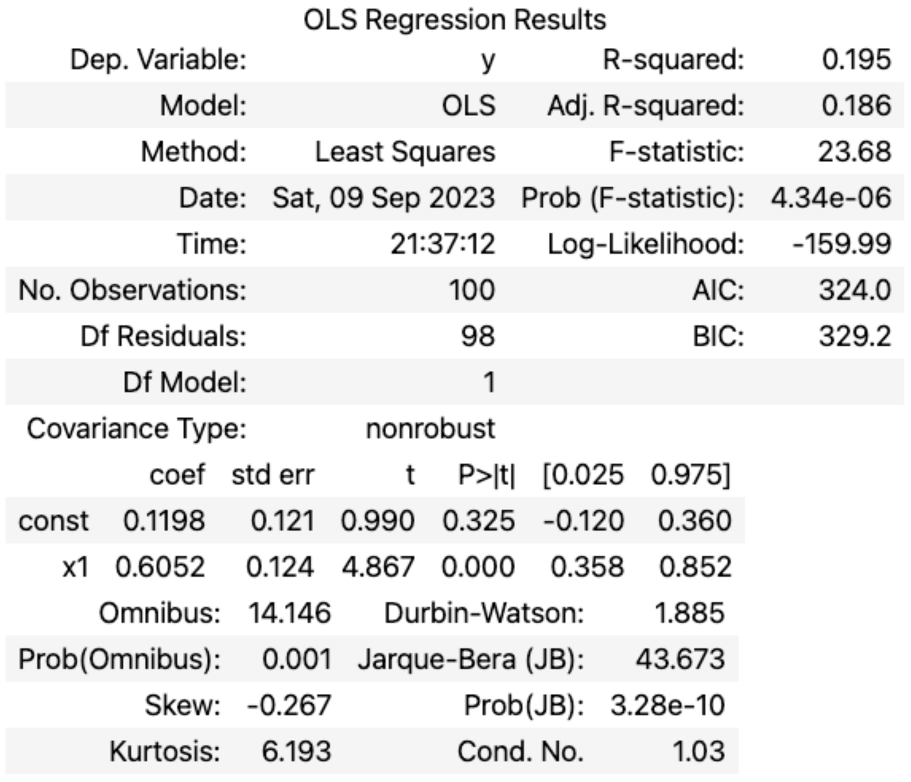
(1) Generate 100 random samples from standard normal distribution.
(2) Record the skewness and kurtosis computed by *skew()* function and *kurtosis()* function respectively for these samples. Note that for both functions, there are also
(3) Repeat the steps above to get the sample sets of skewness and kurtosis with a given sample size.
(4) For each part, if the sample number is smaller than 30, then use t-statistics from student's t-distribution to compute the p-value; otherwise, use z-score from standard normal distribution to compute the p-value. Check if the p-value is smaller than the significance level, which is set to α=0.05. If it is smaller than the significance level, then we reject the null hypothesis and have strong evidence to prove that this function provides a biased result.

The sample sizes used in this experiment are 10, 100, and 1000. Below is the table that demonstrates the experiment result:

Sample size	Skewness		Kurtosis	
	t-stats/z-score	p-value	t-stats/z-score	p-value
10	2.8216	0.02	-3.2033	0.0108
100	0.3267	0.7439	-0.3080	0.7581
1000	-0.4570	0.6476	-3.5229	0.0004

According to the table, we have for sample size equals to 10, both skewness and kurtosis have p-value < α=0.05, which indicates that we have sufficient evidence to reject H_0 and thus both are biased. However, when the sample sizes increase, the p-value becomes larger than the significance level α=0.05, and thus we fail to reject the null hypothesis and can only claimed that both are unbiased (except for the case that when sample size equals to 1000, the kurtosis has p-value < 0.05, which indicates that it is still biased).

Problem 2:
The fitted model using OLS on the data in problem2.csv and the error vector are shown both below and in the code file (code.ipynb):

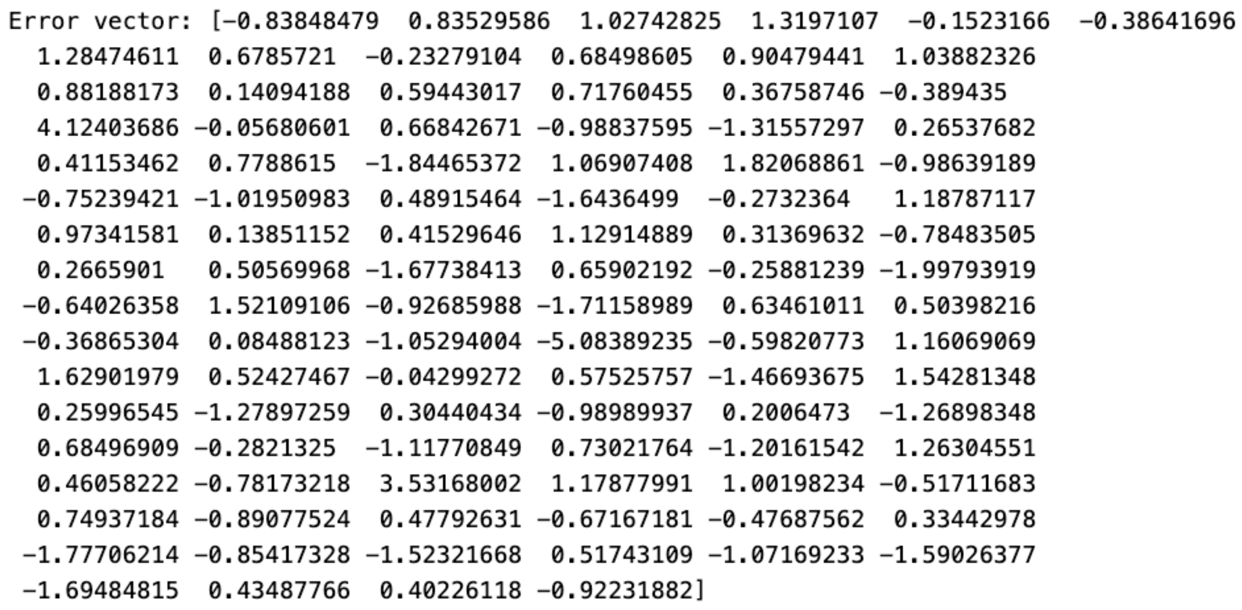


OLS Regression Results

Dep. Variable:	y	R-squared:	0.195
Model:	OLS	Adj. R-squared:	0.186
Method:	Least Squares	F-statistic:	23.68
Date:	Sat, 09 Sep 2023	Prob (F-statistic):	4.34e-06
Time:	21:37:12	Log-Likelihood:	-159.99
No. Observations:	100	AIC:	324.0
Df Residuals:	98	BIC:	329.2
Df Model:	1		
Covariance Type:	nonrobust		

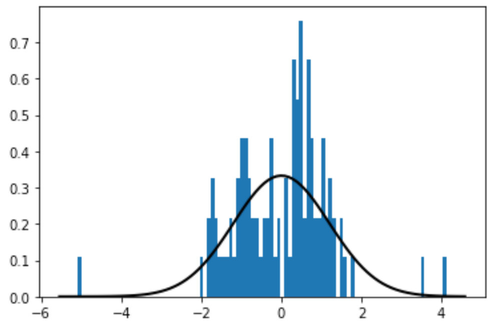
	coef	std err	t	P>|t|	[0.025	0.975]
const	0.1198	0.121	0.990	0.325	-0.120	0.360
x1	0.6052	0.124	4.867	0.000	0.358	0.852

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



```
Error vector: [-0.83848479  0.83529586  1.02742825  1.3197107  -0.1523166  -0.38641696
  1.28474611  0.6785721  -0.23279104  0.68498605  0.90479441  1.03882326
  0.88188173  0.14094188  0.59443017  0.71760455  0.36758746 -0.389435
  4.12403686 -0.05680601  0.66842671 -0.98837595 -1.31557297  0.26537682
  0.41153462  0.7788615  -1.84465372  1.06907408  1.82068861 -0.98639189
 -0.75239421 -1.01950983  0.48915464 -1.6436499  -0.2732364   1.18787117
  0.97341581  0.13851152  0.41529646  1.12914889  0.31369632 -0.78483505
  0.2665901   0.50569968 -1.67738413  0.65902192 -0.25881239 -1.99793919
 -0.64026358  1.52109106 -0.92685988 -1.71158989  0.63461011  0.50398216
 -0.36865304  0.08488123 -1.05294004 -5.08389235 -0.59820773  1.16069069
  1.62901979  0.52427467 -0.04299272  0.57525757 -1.46693675  1.54281348
  0.25996545 -1.27897259  0.30440434 -0.98989937  0.2006473  -1.26898348
  0.68496909 -0.2821325  -1.11770849  0.73021764 -1.20161542  1.26304551
  0.46058222 -0.78173218  3.53168002  1.17877991  1.00198234 -0.51711683
  0.74937184 -0.89077524  0.47792631 -0.67167181 -0.47687562  0.33442978
 -1.77706214 -0.85417328 -1.52321668  0.51743109 -1.07169233 -1.59026377
 -1.69484815  0.43487766  0.40226118 -0.92231882]
```

And here is the distribution of the error vector with relevant statistics:



```
Mean: -1.2212453270876722e-17
Variance: 1.4361484854062607
Skewness: -0.26726658552879606
Kurtosis: 3.1931010009568777
```

From the visualized graph above, it seems like **the errors do not quite fit the assumption of normality well**, as its shape deviates from the normal distribution heavily.

In this part, I also used the Shapiro-Wilk test, with the null hypothesis is that the errors are normally distributed, to further check the normality of the error vector. The resulting test statistic is 0.9384 and the p-value is 0.00015, which is smaller than the significance level α=0.05. Hence, we have statistically significant evidence to reject the null hypothesis, and so **the error vector is not normally distributed**.

For fitting the data using MLE for both normality assumption and t-distribution assumption , I used the following steps:

(1) Construct the negative log-likelihood function based on the assumption of distribution.
(2) Use *minimize()* function from *scipy.optimize* package module to compute the minimized negative log-likelihood, and the corresponding parameters.

The detailed implementation is in code.ipynb.

To determine which one is the best fit, I used AIC and BIC for each fitted model. The result is shown on the table below:

Assumption	AIC	BIC
OLS	324.0	329.2
MLE under normality	325.9841933783253	333.7997039362896
MLE under t-distribution	319.03057455157347	329.4512552955258

According to the table, we have **using MLE using the assumption of a T distribution of the errors is the best fit among all models**, since it has a smaller AIC value comparing to the other two and a smaller BIC value comparing to fitted MLE model under normality and a close BIC value comparing to the OLS model.

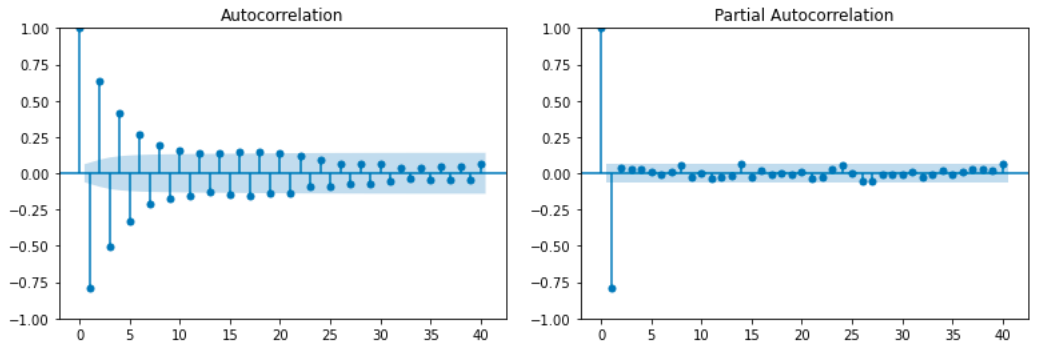
The fitted parameters for each model are listed below:

Model	k (coefficient)	b (intercept)
OLS	0.6052	0.1198
MLE under normality	0.60520485	0.11983628
MLE under t-distribution	0.55893058	0.06998071

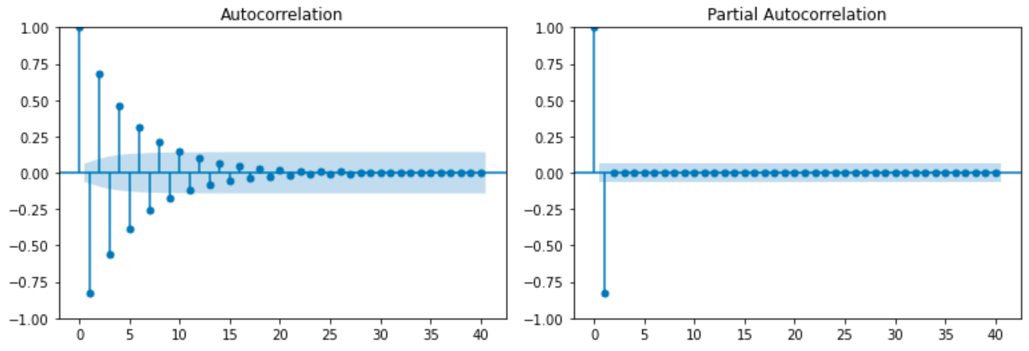
According to the parameters listed, we can figure out that the OLS model and the fitted MLE model under normality assumption has approximately the same parameters, yet the fitted MLE model under t-distribution assumption has different parameters; this indicates that **breaking the normality assumption will result in a different fitted mode**l, and in this case **this different model is a better model for fitting the data**.

Problem 3:

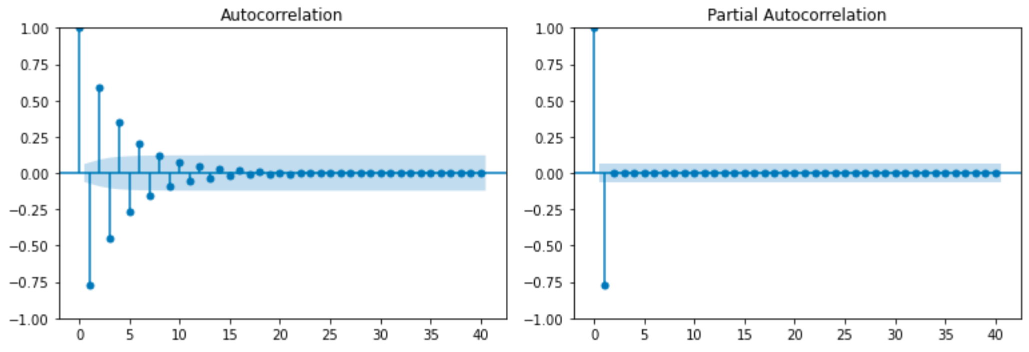
AR(1) (Coefficient = [0.8]):



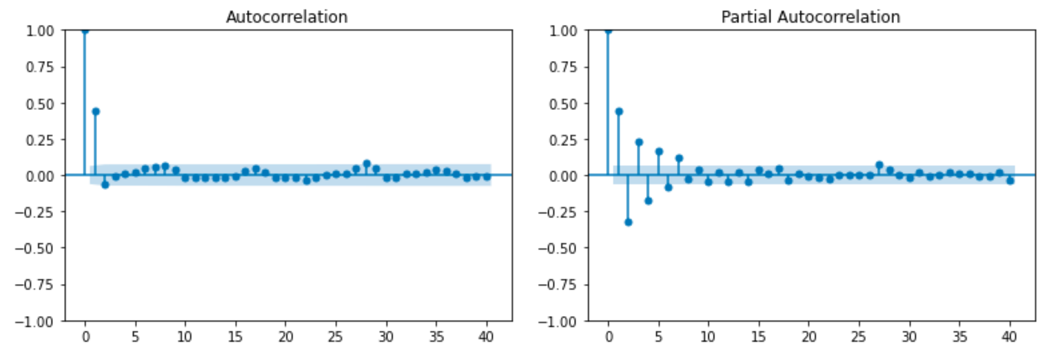
AR(2) (Coefficient = [0.8, -0.5]):



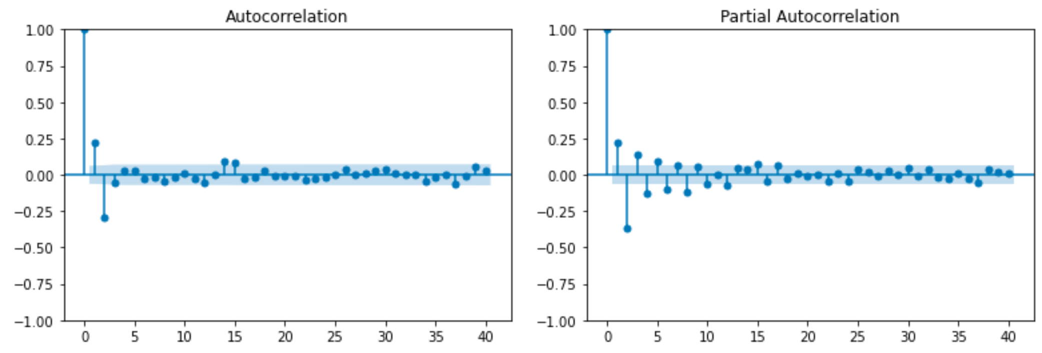
AR(3) (Coefficient = [0.8, -0.5, 0.2]):



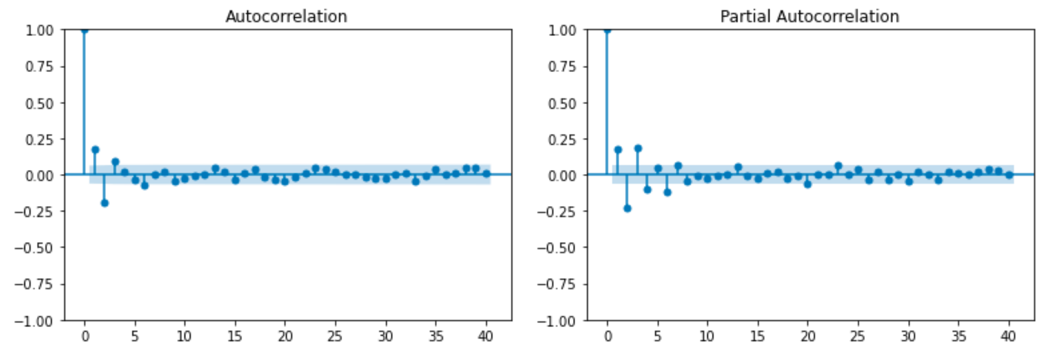
MA(1) (Coefficient = [0.8]):



MA (2) (Coefficient = [0.8, -0.5]):



MA (3) (Coefficient = [0.8, -0.5, 0.2]):



According to the ACF and PACF graphs shown above, we can figure out that an AR process has an autocorrelation trend that converges to 0 gradually and a partial autocorrelation trend that converges rapidly throughout the process, especially between the first two lags and the remaining lags; while for an MA process, it has its autocorrelation trend that converges quickly with the partial autocorrelation trend that converges a bit more slowly. Therefore, based on the ACF graphs and PACF graphs, here is a general way to distinguish AR and MA process: Figure out which one has a more rapid converging trend on the graph—**if ACF has a trend that converges quickly comparing to that of PACF, then it is MA; if PACF has a rapidly converging trend than that of ACF, then it is AR**.