Time Series Econometrics

EBGN 594

Homework 1, Fall 2024

Due by midnight, Tuesday 9/10

There are two parts to this homework. You only need to complete **EITHER Part I or Part II**. (You are welcome to do both, but you don’t need to). Please be as brief as you possibly can. Communicate only what is most important. Do not hand in dozens of pages of code, output, and graphs. Don’t turn in your code unless you have something you need help figuring out or it helps communicate your answer. Please DO turn in a brief summary of results and interpretation. Save your code and writeup because you may build on this work as you develop a final project or prepare for take home exams.

**Part I**

The Law of Large Numbers and Central Limit Theorem say that estimates should converge to their true values as the sample size gets big (LLN), and sample estimates should be normally distributed around their true values across many independent data samples taken from the same underlying process or population (CLT).

In this problem you will show via simulation whether **failures** of assumptions of the Gauss-Markov theorem will cause sample regression coefficients to **fail to** converge to a normal distribution around their true values as the sample size or number of samples gets large.

You will combine some code from the example programs BiasEfficiencyGaussMarkov.R and LLN\_CLT\_SampleStats.R. Specifically: use the case of an omitted variable, Case 5b (lines 271 to 291) from BiasEfficiencyGaussMarkov.R, combined with the LLN and CLT simulation steps in lines 180 to 202 from LLN\_CLT\_SampleStats.R in order to show whether at least one coefficient converges to a normal distribution around it’s true value when that regressor is correlated with the residual because of an omitted variable. Case 5b shows that the estimate from a single sample is not that close to the true coefficient – now you will show whether this problem improves in larger samples or in many samples.

For a challenge, extend this exercise to Case 4 (serially correlated residuals, line 190) and/or Case 5c (serially correlated residuals correlated with lagged dependent variable regressor, line 295) from BiasEfficiencyGaussMarkov.R. The trick here is that because order of observations matters in time series, you can’t simulate a single giant population X from which to take many random samples – you will instead need to simulate many random samples (or larger and larger individual samples) from the arima.sim generator.

**Part II**

Please pick at least two equity and/or commodity prices whose relationship you want to investigate, read them into R using whatever method you like (e.g., “getSymbols” from the quantmod package, or “read.table” or “read.csv” if you have the data on file), and calculate their log returns. You will run a regression of one return series on the other(s). Often people pick a single equity or commodity vs. a market index like the S&P in order to estimate a CAPM-type model, or they pick an equity price from a commodities industry along with its corresponding commodity price, or two equities or two commodities from related industries. It’s up to you. Then do the following:

1. Build a table of summary statistics for the log returns of your two (or more) variables using the “basicStats” command from the fBasics package. You can follow lines 100-183 of the ReturnsDistributions.R example program.
2. Test for the Normality of each of the return series using a Jarque-Bera (JB) test. State the null and alternative hypotheses in terms of the JB test statistic, and report either the p-value, or the critical value relative to the test statistic from your data. For at least one of the return series, please also test for skewness and excess kurtosis separately using t-tests.
3. For at least one returns series (preferably the dependent variable in the regression you plan to run in (4)), characterize the autocorrelation in the series, i.e., the raw returns data. In other words, plot the ACF and the PACF and state any important observations in a bullet point or two. Report and interpret the results of the appropriate Portmanteau-type test (Box-Pierce or Ljung-Box), again stating the null and alternative hypotheses of the test and reporting the p-value or the critical value relative to the test statistic from your sample.
4. Pick one of your return series to be the dependent variable in a regression on your other return series as independent variable(s). Estimate the regression, and report the coefficients and their standard errors. Capture the residuals of the regression and report some evidence for serial correlation in the residuals (e.g., an ACF, PACF, or Portmanteau/Ljung-Box/Q-test). The key difference between this and (3) is that now you are testing autocorrelation in the regression residuals rather than in the returns data itself as you did in (3).

Some R hints:

You can modify my R codes from Canvas and use quantmod to read data from FRED or Yahoo! Finance. There is also another package called “tidyquant” that pulls financial time series from additional sources, including some metals commodity prices that are not on FRED. Handy tutorials are here:

[Tidy Quantitative Financial Analysis • tidyquant (business-science.github.io)](https://business-science.github.io/tidyquant/)

[Performance Analysis with tidyquant • tidyquant (business-science.github.io)](https://business-science.github.io/tidyquant/articles/TQ05-performance-analysis-with-tidyquant.html)

If you find data from another source, you can always download it, store it as a CSV file with variable names in the first row, and read it into R using something like the following:

dataset <- read.csv(file="C:/TheDataIWantToReadIn.csv", header=TRUE, sep=",")

header=TRUE indicates that the first row consists of variable names.

sep=”,” denotes that the file formatting separates columns using commas (hence, CSV=comma separated values).

However, if you read in price data and you don’t use quantmod, you can’t use the dailyReturn or monthlyReturn functions in R. You will have to calculate log returns yourself in R like so:

stock.rtrn = 100\*diff(log(pricevariable))