Washington University Spring 2010
Department of Economics James Morley

Economics 5161

Homework #3
Due on Tuesday 4/13

1. Autoregressive Processes (20 points)

In this exercise, you will simulate data for different Gaussian AR processes in GAUSS or R. In GAUSS, there are different ways to simulate data from an AR process, but I would recommend using a basic "do" loop in order to make what you are doing as transparent as possible.

Here is the general equation for Gaussian AR(p) processes:

$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t$, where $\varepsilon_t \sim iidN(0,\sigma_\varepsilon^2)$.

Here are the cases that I want you to consider. For simplicity, set $c = 0$ and $\sigma_\varepsilon^2 = 1$.

1. AR(1) with $\phi_1 = 0.5$.
2. AR(1) with $\phi_1 = 0.95$.
3. AR(2) with $\phi_1 = 0.75$ and $\phi_2 = 0.2$.
4. AR(2) with $\phi_1 = 1.35$ and $\phi_2 = -0.7$.
5. AR(3) with $\phi_1 = 1.35$, $\phi_2 = -0.8$, and $\phi_3 = 0.2$.

i. Before simulating data, I want you to determine the true autocorrelation function (ACF) and partial autocorrelation function (PACF) for each of the five cases. Recall that an autocorrelation is $\rho_j = \frac{\gamma_j}{\gamma_0}$, where $\gamma_j = \text{cov}(y_t, y_{t-j})$ and an ACF is a plot of ρ_j against j. Do this for $j = 1,...,12$. For the PACF, the calculation for a partial autocorrelation can be found on p. 138 of the class reader, which is from Section 4.8 of Hamilton (1994). As discussed there, the partial autocorrelation α_m is the population coefficient on the mth lag in a linear projection of the time series on m lags and can be calculated for each m as follows:

$$\begin{bmatrix} \alpha_1^{(m)} \\ \alpha_2^{(m)} \\ \vdots \\ \alpha_m^{(m)} \end{bmatrix} = \begin{bmatrix} \gamma_0 & \gamma_1 & \cdots & \gamma_{m-1} \\ \gamma_1 & \gamma_0 & \cdots & \gamma_{m-2} \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_{m-1} & \gamma_{m-2} & \cdots & \gamma_0 \end{bmatrix}^{-1} \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_m \end{bmatrix}$$

That is, $\alpha_m = \alpha_m^{(m)}$. Note that this calculation is not as hard as it looks for this assignment because $\alpha_m = 0$ for $m > p$. Also, note that $\alpha_p = \phi_p$, making most of the calculations that you have to do quite trivial. Finally, note that for each lag m, you need to set up the matrix calculation above and find $\alpha_m = \alpha_m^{(m)}$. E.g., for, say, an AR(2), you need to do it for *m*=1 and then again for *m*=2. The PACF, then, is a plot of α_m against *m*. Again, plot from 1 to 12 lags. You should have eight panels (an ACF and PACF for each of the four processes). Also, show your workings for the AR(2) and AR(3) cases. (2 pages)

ii. Now, for each process, simulate a series of sample size = 100 and sample size = 500. Plot these series in eight panels. Discuss the visual differences in the persistence of the series. Report the code that you used to simulate 500 observations from the fifth process. (2 pages)

iii. For the simulated series of sample size = 100, calculate the sample ACF and sample PACF and report these in ten panels. Note that the sample PACF is straightforward to calculate, with $\hat{\alpha}_m$ being the estimate of the last coefficient in an OLS regression of the series on a constant and *m* lags (see p. 62 or p. 138 of the reading package for more discussion). Discuss how well the sample analogues match up with the corresponding true ACFs and PACFs that you calculated in part i. Then do the same for the simulated series of sample size = 500. (2 pages)

2. Monte Carlo Analysis (20 points)

In this exercise, you will conduct some very basic Monte Carlo analysis to see the properties of the least squares estimator for an AR(1) coefficient and the spurious regression phenomenon.

A Monte Carlo experiment involves simulating repeated samples of data from a known Data Generating Process (DGP) and determining characteristics of, say, estimators or test statistics for the given DGP. Note that the characteristics may be highly dependent on the particular parameter assumptions for the DGP, thus one should be careful about extrapolating what the behaviour would be for other parameter assumptions or other DGPs. On the other hand, doing so is the basis of so-called "bootstrap" analysis that we will consider later in the course.

For the Monte Carlo experiments, you need to determine the following things:

- Number of Monte Carlo samples (which I like to call "nsim"). Let's choose nsim=1,000. There will still be some "Monte Carlo" error, although it will, in most experiments, be small given such a large number of simulations. Monte Carlo error means that the characteristics that you find will only be accurate to a certain number of decimal places. Of course, the cost of a larger number of simulations is computational time. This cost has diminished massively in recent years. If you want, you can compare your results for 1,000 simulations to, say, only 100 or 500 simulations in order to get a sense of the Monte Carlo error.
- DGP: You need to code up how to simulate data for a given DGP with given parameters. You've already done this in the previous exercise. The difference now is that you will loop around this simulation to generate 1,000 samples instead of just one.
- You need to code up how to construct an estimate or test statistic given a particular sample. Again, you've done this in the previous homeworks. The only difference is that you are going to do this within the loop for each simulation. That is, you will repeat the calculation for each simulated sample and store the result. The code for the calculation is the same as if you had real data. Meanwhile, the code for storing the results is straightforward. Outside of the loop for each simulation, set up storage space. For example, suppose you want to store a parameter estimate "b". You can define a storage space called "b_mat={};" outside of the loop. Then, inside the loop, every time you calculate b, you can add it to the storage space as follows: "b_mat=b_mat|b;". Then, after you've gone through all 1,000 simulations, you will have 1,000 draws of b from its distribution stored in b_mat. Thus, it should be straightforward to determine features of this distribution (e.g., use the average draw (i.e., "meanc(b_mat);") to determine the mean of the distribution or the standard deviation of draws (i.e., "stdc(b_mat);"). This will allow you to see if the estimator is unbiased.

i. For the first two DGPs in question 1 (i.e., the two AR(1) processes, with coefficients 0.5 and 0.95, respectively), determine the distribution of the OLS estimator for the autoregressive coefficient when the sample size is 100 and then when it is 500. That is, you will have four cases to consider: two AR(1) DGPs and two sample sizes. Report the Monte Carlo estimates of mean and standard deviation for the OLS estimator for each of the four cases. What do you notice about the behaviour of the estimator in each case? Compare across cases. Also, report the code you used for the Monte Carlo experiment for the last case. (2 pages)

ii. Again consider the first two DGPs and sample sizes of 100 and 500. However, now in each case, we are going to consider two independent AR processes with the same coefficients. That is, for each simulation, draw two samples, one of which will be treated as a realization for series 1 and the other of which will be treated as a realization for series 2. Then, run a regression of series 1 on a constant and series 2. Construct a t-statistic for the hypothesis that the coefficient on series 2 is zero. To do this, you will need to calculate the OLS

standard error. Store the t-statistic for each simulation. After running 1,000 simulations, sort the stored t-statistics. What are the 2.5^{th} and 97.5^{th} percentiles of the draws of the t-statistics for the Monte Carlo experiment? Repeat this for the two DGPs and the two sample sizes (i.e., four cases). How do the results compare to the traditional assumption that a t-statistic has a Student-t distribution? I.e., what would happen if you used a Student-t distribution to test the hypothesis that the two series are related? Would you reject more or less than 5% of the time using a 5% critical value for a two-tailed test? Compare across sample sizes. Again, report the code for the Monte Carlo experiment for the last case. (2 pages)