Homework 10

36-467, Fall 2018

Due at 6 pm on Wednesday, 14 November 2018

AGENDA: Getting familiar with model-based bootstrapping, simulation-based tests, and model checking.

In this assignment, we will continue to work with the data set on the historical population of lynxes in Canada, introduced in Lecture 18.

1. (5) Plot the data. Roughly how far apart are the peaks in the curve? Call this the **inter-peak interval**.
2. (5) Plot the autocovariance function. This should start out positive, go negative, and then become positive again. At roughly what lag is the peak of this second positive region? What does this correspond to in the plot of the time series?
3. (5) Fit an AR(1), and AR(2), and an AR(3) to this data. (Include an intercept.) Display their coefficients in a table. Does the coefficient on $X(t-1)$ stay more or less the same? Should it?
4. (10) Explain how the following code differs from the `sim.ar1` function we have written before:

   ```
   sim.ar1.new <- function(mdl, x.start, n) {
       x <- vector(length=n)
       x[1] <- x.start
       epsilon <- sample(mdl$resid[-1], size=n-1, replace=TRUE)
       for (t in 2:n) {
           x[t] <- mdl$x.intercept + mdl$ar[1]*x[t-1] + epsilon[t-1]
       }
       return(x)
   }
   ```

   In particular, explain where this new simulator gets the distribution it uses for the innovations. Also explain what kind of input it needs in its `mdl` argument.
5. *General hint:* Lecture 18.

(a) (5) Simulate your estimated AR(1) model 100 times, and use this model-based bootstrap to give standard errors for its parameters.

(b) (8) Repeat this for your estimated AR(2). *Hint:* You will need to modify some code.

(c) (7) Repeat again, but for your estimated AR(3). *Hint:* You will need to modify some code again.

6. (a) (5) Simulate your estimated AR(2) 100 times. Fit an AR(3) to each simulation run, and record the coefficient estimated for $X(t-3)$. Store these in a vector `b3.star`. Report the summary statistics for `b3.star` and for `abs(b3.star)`.

   (b) (10) Store the coefficient on $X(t-3)$ you estimated when you fit an AR(3) to the data in a variable `b.hat`. Is `mean(abs(b3.hat) >= abs(b3.star))` an (approximate) p-value? If so, what hypothesis is being tested, against what alternative? If not, why not?

7. (10) Simulate your estimated AR(3) 100 times, and fit an AR(3) to each simulation run. Find a 95% confidence interval for the coefficient on $X(t-3)$ using these simulated values. Does the confidence interval include 0? Does this constitute a test of whether the coefficient is 0? If so, what is the size (false positive rate) of the test? If not, why not?

8. (a) (5) Simulate your estimated AR(1), AR(2) and AR(3) models once each, and plot their three autocovariance functions. Compare them (in words) to the ACF you got from the data in Problem 2.

   (b) (5) Simulate your estimated AR(1) 100 times, and, on each simulation, calculate the autocovariance at a lag corresponding to the inter-peak interval. Plot a histogram of these autocovariances, and add a vertical line indicating the autocovariance you calculated from the data in Problem 2.

   (c) (5) Repeat this for your estimated AR(2) model.

   (d) (5) Repeat this for your estimated AR(3) model.

Note: If you think you can re-use simulations from one problem to the next, explain why, and do so.

RUBRIC (10): The text is laid out cleanly, with clear divisions between problems and sub-problems. The writing itself is well-organized, free of grammatical and other mechanical errors, and easy to follow. Questions which ask for a plot or table are answered with both the figure itself and the command (or commands) use to make the plot. Plots are carefully labeled, with informative and legible titles, axis labels, and (if called for) sub-titles and legends; they are placed near the text of the corresponding problem. All quantitative and mathematical claims are supported by appropriate derivations, included in the text, or calculations in code. Numerical results are reported to appropriate precision. Code is properly integrated with a tool like R Markdown or knitr, and both the knitted file and the source file are submitted. The code is indented, commented, and uses meaningful names. All code is relevant, without dangling or useless commands. All parts of all problems are answered with coherent sentences, and raw computer code or output are only shown when explicitly asked for.