Programming assignment No. 3

General hints:

- Although the submission of this assignment is not mandatory, you are advised and encouraged to submit your solution, as you can earn bonus points (to be added to your final exam score).
- The solutions can be submitted until Thursday, December 15 to ralf.brueggemann@uni-konstanz.de. Late submissions will not be considered.
- Use Python[*] for the following programming problems. Program from scratch and don't use 'packages' or 'functions' by other people. Programming on your own helps you to understand the relevant concepts. The knowledge you acquire by solving the assignment will be helpful in both the take home exam and the final exam.
- Name your code file with the assignment number and your student number (not your name). Example: For programming assignment 2 (PA2) and if your student number is 007007, call the file PA2_007007.py
- Policy with regard to academic dishonesty: Students who wish to work together on assignment material may do so. Please indicate the student numbers of all fellow students that have worked in your group. **Each student must submit her or his individual code/solution file**.
- Write your code general enough such that it can be easily reused. Use comments to document your code properly.
- Avoid loops as much as possible and use matrix language techniques whenever possible.

1) Read Appendix D in Lütkepohl (2005). Write a function that implements a residual bootstrap for a VAR(p) with intercept and returns the bootstrap standard errors of the VAR coefficients in B.[†] The function should take a $T + p \times K$ matrix of observations on y_t, the lag length p, and the number of bootstrap replications R as input.
2) Use the VAR(2) from 2) on programming assignment 2 and your function from 1) with $R = 499$ bootstrap replications. Report the bootstrap standard errors of the VAR coefficients and compare them to the asymptotic standard errors from the Python VAR package/or your VAR estimation function.
3) Write a Python function that simulates time series data from a K-dimensional VAR(2) process $y_t = A_1 y_{t-1} + A_2 y_{t-2} + u_t$, where the innovations u_t are drawn from a multivariate normal distribution with mean zero and covariance matrix Σ_u. Use $y_{-1} = y_0 = \mathbf{0}$ as starting values, where $\mathbf{0}$ is a $K \times 1$ vector of zeros, generate time series of length $T + 50$ and discard the first 50 observations, such that you have available time series of total length equal to T. Your function should take A_1, A_2, Σ_u and T as an input and should return a $T \times K$ matrix of observations on y_t.

[*]We recommend using the Anaconda distribution with Spyder as an IDE, see ILIAS for documentation.
[†]Use the standard deviation over the R bootstrap VAR estimates as a bootstrap standard error.

4) Write a Python function that computes the h-step ahead point forecasts $y_T(h)$ and the corresponding MSE matrix $\hat{\Sigma}_y(h)$ based on a VAR(p) with intercept. The inputs to the function should be a $K \times T$ matrix of observations, the lag order p, and the forecast horizon h. As an output, the function should return the h-step ahead forecasts and the corresponding MSE matrix.[‡]

5) Use your function in 3) and generate time series data of length $T = 100$ according to a bivariate VAR(2) process with the following parameters:

$$A_1 = \begin{bmatrix} 0.4 & 0.25 \\ 0 & 0.5 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0.2 & 0.4 \\ 0 & 0 \end{bmatrix}, \quad \Sigma_u = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}.$$

Use the function in 4) to compute h-step ahead forecasts and the corresponding MSE matrix based on a VAR(2) process with intercept for $h = 1$ and $h = 4$. Use these estimates to set up a 95% interval forecasts assuming that the process y_t is Gaussian.[§]

References

Lütkepohl, H. (2005). *New Introduction to Multiple Time Series Analysis*, Berlin: Springer-Verlag.

[‡]Consider using equation 3.5.10 from Lütkepohl (2005) for the point forecasts.

[§]You may want to compare your results with those of the Python VAR package function `forecast_interval`.