Bayesian data analysis – exercise 5

This assignment is related to Chapters 10 and 11.

The maximum amount of points from this assignment is 6. In addition to the correctness of the answers, the overall quality and clearness of the report is evaluated.

Report all results to a single, **anonymous** *.pdf -file and return it to peergrade.io. Include also source code to the report (either as an attachment or as a part of the answer). By anonymity it is meant that the report should not contain your name or student number.

1. Generalized linear model: Bioassay with Metropolis (6 points)

Metropolis algorithm: Replicate the computations for the bioassay example of section 3.7 (BDA3) using the Metropolis algorithm. Be sure to define your starting points and your jumping rule (proposal distribution). Run the simulations long enough for approximate convergence.

Information and hints

- Use the Gaussian prior as in Exercise 4, that is

$$\begin{bmatrix}\alpha\\ \beta\end{bmatrix} \sim \mathrm{N}\left(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0\right) \qquad \text{where} \quad \boldsymbol{\mu}_0 = \begin{bmatrix}0\\ 10\end{bmatrix} \quad \text{and} \quad \boldsymbol{\Sigma}_0 = \begin{bmatrix}2^2 & 10\\ 10 & 10^2\end{bmatrix}$$

- Use a simple proposal distribution, there is no need to try to find optimal proposal but remember to report the one you used. Efficient proposals are dicussed in BDA3 p. 295–297 (not part of the course). In real-life a pre-run could be made with an automatic adaptive control to adapt the proposal distribution.
- Metropolis is a simple algorithm, you do not need many lines of code for it.
- Compute with log-densities. Reasons are explained on page 261 (BDA3). Remember that $p_1/p_0 = \exp(\log(p_1) - \log(p_0))$. For your convenience we have provided functions for R and Python that will evaluate the log-likelihood for given α and β (see `bioassaylp.R` and `bioassaylp.py`). Notice that you still need to add the prior yourself. Remember the unnormalized log posterior is simply the log-likelihood plus log-prior. For evaluating the log of the Gaussian prior, in R you can use function `dmvnorm` from package `mvtnorm`, or in Python use function `scipy.stats.multivariate_normal.logpdf`.
- Use $\hat{R}$ for convergence analysis (Eq. (11.4) in BDA3). In R the easiest choice is to use function `R.hat` from package `asbio`. In Python you can use the provided function in the `psrf.py` file. Remember to remove the warm-up samples before computing $\hat{R}$ (although notice that function `R.hat` does this automatically, you simply need to specify the burn-in length).

- Include in the report the proposal distribution, number of chains used, the starting points (or the mechanism for generating them), the number of draws generated from each chain, and the warm-up length. Report also the $\hat{R}$ values for α and β and draw conclusions about the convergence of the chains. **This means that you should briefly explain how to interpret the obtained $\hat{R}$ values**.
- Plot also the draws for α and β (scatter plot) and include this plot in you report. You can compare the results to Figure 3.3b in BDA3 to verify that your code gives sensible results. Notice though that the results in Figure 3.3b are generated from posterior with uniform prior, so you even when your algorithm works perfectly, the results will look slightly different (although fairly similar).