Bayesian data analysis – Assignment 5

Information on this assignment

This exercise is related to Chapters 10 and 11. The maximum amount of points from this assignment is 6.

Reading instructions: Chapter 10 and 11 in BDA3, see reading instructions **here** and **here**.

Grading instructions: The grading will be done in peergrade. All grading questions and evaluations for assignment 5 can be found **here**

Reporting accuracy: For posterior statistics of interest, only report digits for which the Monte Carlo standard error (MCSE) is zero. *Example:* If you estimate $E(\mu) = 1.234$ with MCSE$(E(\mu)) = 0.01$, you should report $E(\mu) = 1.2$.

To use markmyassignment for this assignment, run the following code in R:

```
> library(markmyassignment)
> exercise_path <-
    "https://github.com/avehtari/BDA_course_Aalto/blob/master/assignments/tests/ex5.yml"
> set_assignment(exercise_path)
> # To check your code/functions, just run
> mark_my_assignment()
```

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. The Metropolis algorithm is described in BDA3 Chapter 11.2. More information on the bioassay data can be found in Section 3.7 of the course book and in chapter 3 reading instructions **here**.

1. Implement the Metropolis algorithm as an R function for the bioassay data. Use the Gaussian prior as in Assignment 4, that is

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

   a) Start by implementing a function called `density_ratio` to compute the density ratio function, r in Eq. (11.1) in BDA3. Below is an example on how the function should work. You can test the function using `markmyassignment`.

```
> library(aaltobda)
> data("bioassay")

> density_ratio(alpha_propose = 1.89, alpha_previous = 0.374,
                beta_propose = 24.76, beta_previous = 20.04,
                x = bioassay$x, y = bioassay$y, n = bioassay$n)

[1] 1.187524

> density_ratio(alpha_propose = 0.374, alpha_previous = 1.89,
                beta_propose = 20.04, beta_previous = 24.76,
                x = bioassay$x, y = bioassay$y, n = bioassay$n)

[1] 0.8420882
```

   Hint! 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 will evaluate the log-likelihood for given α and β (see `bioassaylp()` in the `aaltobda` package). Notice that you still need to add the prior yourself and remember the unnormalized log posterior is simply the log-likelihood plus log-prior. For evaluating the log of the Gaussian prior you can use function `dmvnorm` from package `aaltobda`. It can be worthwhile to look up your implementation of `p_log_posterior()` that you implemented in Assignment 4.

   b) Now implement a function called `metropolis_bioassay()` which implements the metropolis algorithm using the `density_ratio()`.

   Hint! Use a simple (normal) proposal distribution. Example proposals are $\alpha^* \sim N(\alpha_{t-1}, \sigma = 1)$ and $\beta^* \sim N(\beta_{t-1}, \sigma = 5)$. There is no need to try to find optimal proposal but test some different values for the jump scale (σ). 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.

2. Include in the report the following:

a) The jumping rule (other name for the *proposal distribution*).

b) The starting points of your metropolis chains (or the mechanism for generating them).

c) Report the chain length or the number of draws from each chain. Run the simulations long enough for approximate convergence.

d) Report the warm-up length.

e) The number of metropolis chains used. It is a important that multiple metropolis chains be run for evaluating convergence.

f) Plot all chains for alpha (line plot) in a single plot, overlapping chains help in visually assessing whether chains have converged or not.

g) Do the same for beta.

3. In complex scenarios, visual assessment is not sufficient and $\hat{R}$ is a more robust indicator of convergence of the metropolis chains. Use $\hat{R}$ for convergence analysis. You can either use Eq. (11.4) in BDA3 or the later version that can be found **here**. You should specify which $\hat{R}$ you used. In R the best choice is to use function `Rhat` from package `rstan`. Remember to remove the warm-up samples before computing $\hat{R}$. Report also the $\hat{R}$ values for α and β separately, and discuss the convergence of the chains. **This means that you should briefly explain how to interpret the obtained $\hat{R}$ values.**

4. Plot the draws for α and β (scatter plot) and include this plot in your 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 a uniform prior, so even when your algorithm works perfectly, the results will look slightly different (although fairly similar).