Bayesian data analysis – Assignment 5

General information

- The recommended tool in this course is R (with the IDE R-Studio). You can download R **here** and R-Studio **here**. There are tons of tutorials, videos and introductions to R and R-Studio online. You can find some initial hints **here**.
- You can write the report with your preferred software, but the outline of the report should follow the instruction in the R markdown template that can be found **here**.
- Report all results in a single, **anonymous** *.pdf -file and return it to **peergrade.io**.
- Many of the exercises can be checked using the R package `markmyassignment`. Information on how to install and use the package can be found **here**.
- The course has its own R package with data and functionality to simplify coding. To install the package just run the following:

  1. `install.packages("remotes")`
  2. `remotes::install_github("avehtari/BDA_course_Aalto", subdir = "rpackage")`

- Many of the exercises can be checked automatically using the R package `markmyassignment`. Information on how to install and use the package can be found **here**.
- Additional self study exercises and solutions for each chapter in BDA3 can be found **here**.
- We collect common questions regarding installation and technical problems in a course Frequently Asked Questions (FAQ). This can be found **here**.
- If you have any suggestions or improvements to the course material, please feel free to create an issue or submit a pull request to the public repository!!

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 **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: As many significant digits as justified by the Monte Carlo error and posterior accuracy.

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/exercises/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.

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 the density ratio function to compute 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 the metropolis algorithm using `density_ratio()` as a function called `metropolis_bioassay()`. Be sure to define your starting points and your jumping rule (proposal distribution). Run the simulations long enough for approximate convergence.

   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 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. Also plot the chains separately for alpha and beta, overlapping chains help to visually assess whether the chains have converged or not. However, in complex scenarios, visual assessment is not sufficient and $\hat{R}$ is a more robust indicator of convergence (of the metropolis chains).

3. 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 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).