# Bayesian data analysis – exercise 6

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 Stan (6 points)

Replicate the computations for the bioassay example of section 3.7 (BDA3) using Stan.

### Information and hints

- See the Stan demos on how to use Stan from R or Python. The university Ubuntu desktops have the necessary libraries installed so there should be no need to install anything. To install Stan on your own laptop, see the instructions below.
- In R, install package `rstan`. Installation instructions on Linux, Mac and Windows can be found at `https://github.com/stan-dev/rstan/wiki/RStan-Getting-Started`. Additional useful packages are `loo`, `bayesplot` and `shinystan` (but you don't need these in this exercise).
- In Python, you can use Stan by importing the module `pystan`. To install the Python interface to your own laptop, follow instructions here `http://pystan.readthedocs.io/en/latest/getting_started.html`.
- Use the same Gaussian prior as in Exercises 4 and 5, 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 few Stan tips. You will need functions `multi_normal` and `binomial_logit` for implementing the prior and likelihood, respectively. In Stan code, it is easiest to declare a variable (say `theta`) which is a two-element vector so that the first value denotes $\alpha$ and latter one $\beta$. This is because the `multi_normal` function that you need for implementing the prior requires a vector as an input.
- You can use Stan's default settings for the number of chains, samples per chain and warm-up length.
- Use $\hat{R}$ to assess convergence. Note that Stan gives you these values automatically; in R or Python you can simply type `print(fit)`, where `fit` is the fit object returned by Stan's sampling function. Report the $\hat{R}$ values both for $\alpha$ and $\beta$ 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 the draws for $\alpha$ and $\beta$ (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 even when your algorithm works perfectly, the results will look slightly different (although fairly similar).
- Stan manual can be found at `http://mc-stan.org/documentation/`. From this website you can also find a lot of other useful material about Stan.