Uppsala University

Bayesian Statistics and Data Analysis

Assignment 6

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 many tutorials, videos and introductions to R and R-Studio online. You can find some initial hints from **RStudio Education pages**.
- When working with R, we recommend writing the report using R markdown and the provided **R markdown template**. The template includes the formatting instructions and how to include code and figures.
- Instead of R markdown, you can use other software to make the PDF report, but you should use the same instructions for formatting. These instructions are also available in **the PDF produced from the R markdown template**.
- We supply a Google Colab notebook that you can also use for the assignments. We have included the installation of all necessary R packages; hence, this can be an alternative to using your own local computer. You can find the notebook **here**. You can also open the notebook in Colab **here**.
- Report all results in a single and *anonymous* pdf. Note that no other formats are allowed.
- The course has its own R package `bsda` with data and functionality to simplify coding. To install the package, just run the following (upgrade="never" skips question about updating other packages):

  1. `install.packages("remotes")`
  2. `remotes::install_github("MansMeg/BSDA", subdir = "rpackage", upgrade="never")`

- Many of the exercises can be checked automatically using the R package `markmyassignment`. you can find information on how to install and use the package **here**. There is no need to include `markmyassignment` results in the report.
- You can find common questions and answers regarding the installation and technical problems in Frequently Asked Questions (FAQ).
- You can find deadlines and information on how to turn in the assignments in Studium.
- You are allowed to discuss assignments with your friends, but it is not permitted to copy solutions directly from other students or the internet. Try to solve the actual assignment problems with your code and explanations. Do not share your answers publicly. We compare the answers with the "urkund" system. We will report all suspected plagiarism.
- If you have any suggestions or improvements to the course material, please post in the course chat feedback channel, create an issue, or submit a pull request to the public repository **here**.

- It is *mandatory* to include the following parts in all assignments (these are included already in the template):
    1. Time used for reading: How long time took the reading assignment (in hours)
    2. Time used for the assignment: How long time took the basic assignment (in hours)
    3. Good with assignment: Write one-two sentences of what you liked with the assignment/what we should keep for next year.
    4. Things to improve in the assignment: Write one-two sentences of what you think can be improved in the assignment. Can something be clarified further? Did you get stuck on stuff unrelated to the content of the assignment etc.
- You can find information on how each assignment will be graded and how points are assigned **here**. **Note!** This grading information can change during the course, for example, if we find errors or inconsistencies or do additions to the assignments. Please feel free to comment on these grading instructions, ideally before turning in your assignment, if you think something is missing or is incorrect.
- To pass (G) the assignment, you need 70% of the total points. To pass with distinction (VG), you need 90% of the total points. See the grading information on the point allocations for each assignment.
- On cheating: You are not allowed to show your assignments (text or code) to anyone. Only discuss the assignments with your fellow students. The student that show their assignment to anyone else could also be considered to cheat. Similarly, on zoom labs, only screen share when you are in a separate zoom room with teaching assistants. You are not allowed to use large language models, such as ChatGPT, to write assignments.
- All mathematics need to be done in digital form to simplify grading and commenting. Hence, it is not allowed to write math on paper and add an image in the assignment. If you have difficulties to write math in latex, see https://editor.codecogs.com/.

Information on this assignment

This assignment is related to Chapters 10 and 11.

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

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 $\text{MCSE}(E(\mu)) = 0.01$, you should report $E(\mu) = 1.2$.

When computing the $\hat{R}$ diagnostics, you only need to include two decimals.

1. Generalized linear model: Bioassay with Stan

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

1. Write down the model for the bioassay data in Stan syntax. For instructions in reporting your implementation, you can refer to parts 2 c) - g) in Assignment 5. Also report the posterior means of α and β. More information on the bioassay data can be found in Section 3.7 of the course book. To get access to data, use the following code:

   ```
   library(bsda)
   data("bioassay")
   ```

   Use the Gaussian prior

   $$\begin{bmatrix}\alpha \\ \beta\end{bmatrix} \sim \mathrm{N}\left(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0\right), \qquad \text{where} \quad \boldsymbol{\mu}_0 = \begin{bmatrix}0 \\ 10\end{bmatrix} \quad \text{and} \quad \boldsymbol{\Sigma}_0 = \begin{bmatrix}2^2 & 12 \\ 12 & 10^2\end{bmatrix}.$$

   Hint! You will need Stan functions `multi_normal` and `binomial_logit` for implementing the prior and observation model, 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 α and latter one β. This is because the `multi_normal` function that you need for implementing the prior requires a vector as an input.

2. Use $\widehat{R}$ 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 $\widehat{R}$ you used. In R the best choice is to use function `Rhat` from package `rstan` (see `?rstan::Rhat`). To check $\widehat{R}$ and other diagnostics, you can also call `print(fit)`, where `fit` is the fit object returned by Stan's sampling function. Report the $\widehat{R}$ values both for α and β and discuss the convergence of the chains. **Briefly explain in your own words how to interpret the obtained $\widehat{R}$ values.**

3. 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).

2. Linear model: drowning data with Stan

The provided data `drowning` in the `bsda` package contains the number of people who died from drowning each year in Finland 1980–2019. A statistician is going to fit a linear model with Gaussian residual model to these data using time as the predictor and number of drownings as the target variable (see the related linear model example for the Kilpisjärvi-temperature data in the example Stan codes). She has two objective questions:

i) What is the trend of the number of people drowning per year? (We would plot the histogram of the slope of the linear model.)

ii) What is the prediction for the year 2020? (We would plot the histogram of the posterior predictive distribution for the number of people drowning at $\tilde{x} = 2020$.)

To access the data, use:

```
library(bsda)
data("drowning")
```

Corresponding Stan code is provided in Listing 1. However, it is not entirely correct for the problem. First, there are *three mistakes*. Second, there are no priors defined for the parameters. In Stan, this corresponds to using uniform priors.

Your tasks are the following:

a) Find the three mistakes in the code and fix them. Report the original mistakes and your fixes clearly in your report. Include the *full* corrected Stan code in your report.

Hint: You may find some of the mistakes in the code using Stan syntax checker. If you copy the Stan code to a file ending .stan and open it in RStudio (you can also choose from RStudio menu File→New File→Stan file to create a new Stan file), the editor will show you some syntax errors. More syntax errors might be detected by clicking 'Check' in the bar just above the Stan file in the RStudio editor. Note that some of the errors in the presented Stan code may not be syntax errors.

b) Determine a suitable weakly-informative prior normal$(0, \sigma_\beta)$ for the slope `beta`. It is very unlikely that the mean number of drownings changes more than 50 % in one year. The approximate historical mean yearly number of drownings is 138. Hence, set σ_β so that the following holds for the prior probability for `beta`: $\Pr(-69 <$ `beta` $< 69) = 0.99$. Determine suitable value for σ_β and report the approximate numerical value for it.

c) Using the obtained σ_β, add the desired prior in the Stan code. In the report, in a separate section, indicate clearly how you carried out your prior implementation, e.g. "Added line . . . in block . . . ".

d) In a similar way, add a proprt and weakly informative prior for the intercept `alpha` and explain how you chose the prior.

Hint! Example resulting plots for the problem, with the fixes and the desired prior applied, are shown in Figure 1. If you want, you can use these plots as a reference for testing if your modified Stan code produces similar results. However, running the inference and comparing the plots is not required.

Note! The example/test plots and results are based on data up to 2016. You should report your result for the whole period 2019.

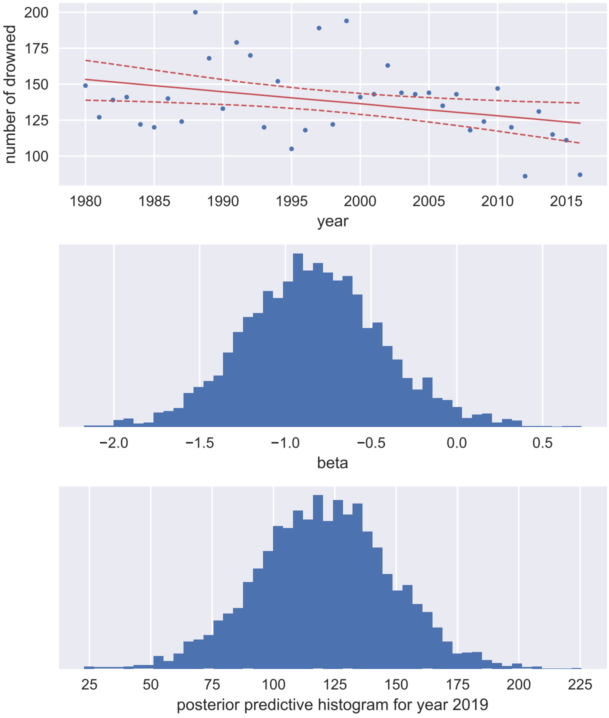

Figure 1: Example plots for the results obtained for the problem in the Question 1 with data until 2016. In the first subplot, the red lines indicate the resulting 5 %, 50 %, and 95 % posterior quantiles for the transformed parameter `mu` at each year.

Listing 1: Broken Stan code for question 1

```
data {
    int<lower=0> N;  // number of data points
    vector[N] x;     // observation year
    vector[N] y;     // observation number of drowned
    real xpred;      // prediction year
}
parameters {
    real alpha;
    real beta;
    real<upper=0> sigma;
}
transformed parameters {
    vector[N] mu = alpha + beta*x;
}
model {
    y ~ normal(mu, sigma)
}
generated quantities {
    real ypred = normal_rng(mu, sigma);
}
```