Assignment 5

Aki Vehtari et al.

1 General information

This assignment is related to Lecture 5 and Chapters 10 and 11.

The maximum amount of points from this assignment is 6.

We have prepared a *quarto template specific to this assignment (html, qmd, pdf)** to help you get started.

If you are not using JupyterHub (which has all the needed packages pre-installed), and want to make the assignment on your own computer, you may use a docker container that includes all the necessary software packages, too.

Tip

Reading instructions:

- **The reading instructions for BDA3 Chapter 10**.
- **The reading instructions for BDA3 Chapter 11**.

Grading instructions:
The grading will be done in peergrade. All grading questions and evaluations for this assignment are contained within this document in the collapsible **Rubric** blocks.

Reporting accuracy

For posterior statistics of interest, only report digits that are not completely random based on the Monte Carlo standard error (MCSE).
Example: If you estimate $E(\mu) \approx 1.234$ with $\text{MCSE}(E(\mu)) = 0.01$, then the true expectation is likely to be between 1.204 and 1.264, it makes sense to report $E(\mu) \approx 1.2$.
See Lecture video 4.1, the chapter notes, and a case study for more information.

Further information

- The recommended tool in this course is R (with the IDE RStudio).
- Instead of installing R and RStudio on you own computer, see **how to use R and RStudio remotely**.
- If you want to install R and RStudio locally, download R and RStudio.
- There are tons of tutorials, videos and introductions to R and RStudio online. You can find some initial hints from **RStudio Education**

pages.

- When working with R, we recommend writing the report using `quarto` and the provided template. The template includes the formatting instructions and how to include code and figures.
- Instead of `quarto`, you can use other software to make the PDF report, but the the same instructions for formatting should be used.
- Report all results in a single, **anonymous** *.pdf -file and submit it in **peergrade.io**.
- The course has its own R package `aaltobda` with data and functionality to simplify coding. The package is pre-installed in JupyterHub. To install the package on your own system, run the following code (upgrade="never" skips question about updating other packages):

```
install.packages("aaltobda", repos = c("https://avehtari.github.io/BDA_course_Aalto/", getOption("re
```

- Many of the exercises can be checked automatically using the R package `markmyassignment` (pre-installed in JupyterHub). Information on how to install and use the package can be found in the `markmyassignment` documentation. There is no need to include `markmyassignment` results in the report.
- Recommended additional self study exercises for each chapter in BDA3 are listed in the course web page. These will help to gain deeper understanding of the topic.
- Common questions and answers regarding installation and technical problems can be found in Frequently Asked Questions (FAQ).
- Deadlines for all assignments can be found on the course web page and in Peergrade. You can set email alerts for the deadlines in Peergrade settings.
- You are allowed to discuss assignments with your friends, but it is not allowed to copy solutions directly from other students or from internet.
- You can copy, e.g., plotting code from the course demos, but really try to solve the actual assignment problems with your own code and explanations.
- Do not share your answers publicly.
- Do not copy answers from the internet or from previous years. We compare the answers to the answers from previous years and to the answers from other students this year.
- Use of AI is allowed on the course, but the most of the work needs to by the student, and you need to report whether you used AI and in which way you used them (See points 5 and 6 in Aalto guidelines for use of AI in teaching).
- All suspected plagiarism will be reported and investigated. See more about the **Aalto University Code of Academic Integrity and Handling Violations Thereof**.
- Do not submit empty PDFs, almost empty PDFs, copy of the questions, nonsense generated by yourself or AI, as these are just harming the other students as they can't do peergrading for the empty or nonsense submissions. Violations of this rule will be reported and investigated in the same way was 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!

Rubric

- Can you open the PDF and it's not blank nor nonsense? If the pdf is blank, nonsense, or something like only a copy of the questions, 1) report it as problematic in Peergrade-interface to get another report to review, and 2) send a message to TAs.
- Is the report anonymous?

⚠ Setup

This is the template for assignment 5. You can download the qmd-file or copy the code from this rendered document after clicking on `</> Code` in the top right corner.
Please replace the instructions in this template by your own text, explaining what you are doing in each exercise.
The following will set-up `markmyassignment` to check your functions at the end of the notebook:

```
if(!require(markmyassignment)){
    install.packages("markmyassignment")
    library(markmyassignment)
}
```

```
Loading required package: markmyassignment
```

```
assignment_path = paste("https://github.com/avehtari/BDA_course_Aalto/",
"blob/master/assignments/tests/assignment5.yml", sep="")
set_assignment(assignment_path)
```

```
Assignment set:
assignment5: Bayesian Data Analysis: Assignment 5
The assignment contain the following task:
- density_ratio
```

The following installs and loads the `aaltobda` package:

```
if(!require(aaltobda)){
    install.packages("aaltobda", repos = c("https://avehtari.github.io/BDA_course_Aalto/", getOption
    library(aaltobda)
}
```

```
Loading required package: aaltobda
```

The following installs and loads the `latex2exp` package, which allows us to use LaTeX in plots:

```
if(!require(latex2exp)){
    install.packages("latex2exp")
    library(latex2exp)
}
```

```
Loading required package: latex2exp
```

The following installs and loads the `posterior` package which imports the `rhat_basic()` function:

```
if(!require(posterior)){
    install.packages("posterior")
    library(posterior)
}
```

```
Loading required package: posterior
```

```
This is posterior version 1.6.0
```

```
Attaching package: 'posterior'
```

```
The following object is masked from 'package:aaltobda':

    mcse_quantile
```

```
The following objects are masked from 'package:stats':

    mad, sd, var
```

```
The following objects are masked from 'package:base':

    %in%, match
```

The following installs and loads the `ggplot2` package and the `bayesplot` package

```
if(!require(ggplot2)){
    install.packages("ggplot2")
    library(ggplot2)
}
```

```
Loading required package: ggplot2
```

```
if(!require(bayesplot)){
    install.packages("bayesplot")
    library(bayesplot)
}
```

```
Loading required package: bayesplot
```

```
This is bayesplot version 1.11.1
```

```
- Online documentation and vignettes at mc-stan.org/bayesplot
```

```
- bayesplot theme set to bayesplot::theme_default()
```

```
   * Does _not_ affect other ggplot2 plots
```

```
   * See ?bayesplot_theme_set for details on theme setting
```

```
Attaching package: 'bayesplot'
```

```
The following object is masked from 'package:posterior':

    rhat
```

2 Generalized linear model: Bioassay model with Metropolis algorithm

Metropolis algorithm: Replicate the computations for the bioassay example of BDA3 Section 3.7 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 in BDA3, and in **Chapter 3 notes**.

Subtask 2.a)

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

$$\begin{bmatrix}\alpha\\ \beta\end{bmatrix} \sim \mathrm{N}\left(\mu_0, \Sigma_0\right), \qquad \text{where} \quad \mu_0 = \begin{bmatrix}0\\ 10\end{bmatrix} \quad \text{and} \quad \Sigma_0 = \begin{bmatrix}2^2 & 12\\ 12 & 10^2\end{bmatrix}.$$

Tip

Compute with log-densities. Reasons are explained on BDA3 page 261 and Lecture video 4.1. Remember that $p_1/p_0 = \exp(\log(p_1) - \log(p_0))$. For your convenience we have provided functions that 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 sum of log-likelihood and log-prior. For evaluating the log of the Gaussian prior you can use the function `dmvnorm` from package `aaltobda`.
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 discussed 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.

Write your answers/code here!

```
# Useful functions: runif, rnorm
# bioassaylp, dmvnorm (from aaltobda)

data("bioassay")
# Start by implementing a function called `density_ratio` to
# compute the density ratio function, $r$ in Eq. (11.1) in BDA3:
density_ratio <- function(alpha_propose, alpha_previous, beta_propose, beta_previous, x, y, n){
    # Do computation here, and return as below.
    # Below are the correct return values for two different calls of this function:

    # 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
    1.305179

    # 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
    0.7661784
```

```
    ### {.content-hidden when-profile="public"}
    prior_mean = c(0, 10)
    prior_sigma = cbind(c(4, 12), c(12, 100))
    exp(
        bioassaylp(alpha_propose, beta_propose, x, y, n)
        - bioassaylp(alpha_previous, beta_previous, x, y, n)
        + dmvnorm(c(alpha_propose, beta_propose), prior_mean, prior_sigma, TRUE)
        - dmvnorm(c(alpha_previous, beta_previous), prior_mean, prior_sigma, TRUE)
    )
    ###
}
# Then implement a function called `metropolis_bioassay()` which
# implements the Metropolis algorithm using the `density_ratio()`:
metropolis_bioassay <- function(alpha_initial, beta_initial, alpha_sigma, beta_sigma, no_draws, x, y,
    # Do computation here, and return as below.
    # Below are "wrong" values (unlikely to actually occur)
    # in the "correct" format (such that they work with the plotting functions further down).
    data.frame(
        alpha=c(alpha_initial, alpha_initial+alpha_sigma, alpha_initial-alpha_sigma),
        beta=c(beta_initial, beta_initial+beta_sigma, beta_initial-beta_sigma)
    )
    ### {.content-hidden when-profile="public"}
    alpha_previous = alpha_initial
    beta_previous = beta_initial
    alpha_rv = c()
    beta_rv = c()
    for(draw in 1:no_draws){
        alpha_propose = rnorm(1, alpha_previous, alpha_sigma)
        beta_propose = rnorm(1, beta_previous, beta_sigma)
        if(runif(1) < density_ratio(alpha_propose, alpha_previous, beta_propose, beta_previous, x, y,
            alpha_previous = alpha_propose
            beta_previous = beta_propose
        }
        alpha_rv = c(alpha_rv, alpha_previous)
        beta_rv = c(beta_rv, beta_previous)
    }
    data.frame(alpha=alpha_rv, beta=beta_rv)
    ###
}
df = metropolis_bioassay(0, 0, 1, 1, 1000, bioassay$x, bioassay$y, bioassay$n)
```

Subtask 2.b)

Include in the report the following:

1. Describe in your own words in one paragraph the basic idea of the Metropolis algorithm (see BDA3 Section 11.2, and Lecture video 5.1).
2. The proposal distribution (related to *jumping rule*) you used. Describe briefly in words how you chose the final proposal distribution you used for the reported results.
3. The initial points of your Metropolis chains (or the explicit mechanism for generating them).
4. Report the chain length or the number of iterations for each chain.

Run the simulations long enough for approximate convergence (see BDA Section 11.4, and Lecture 5.2).

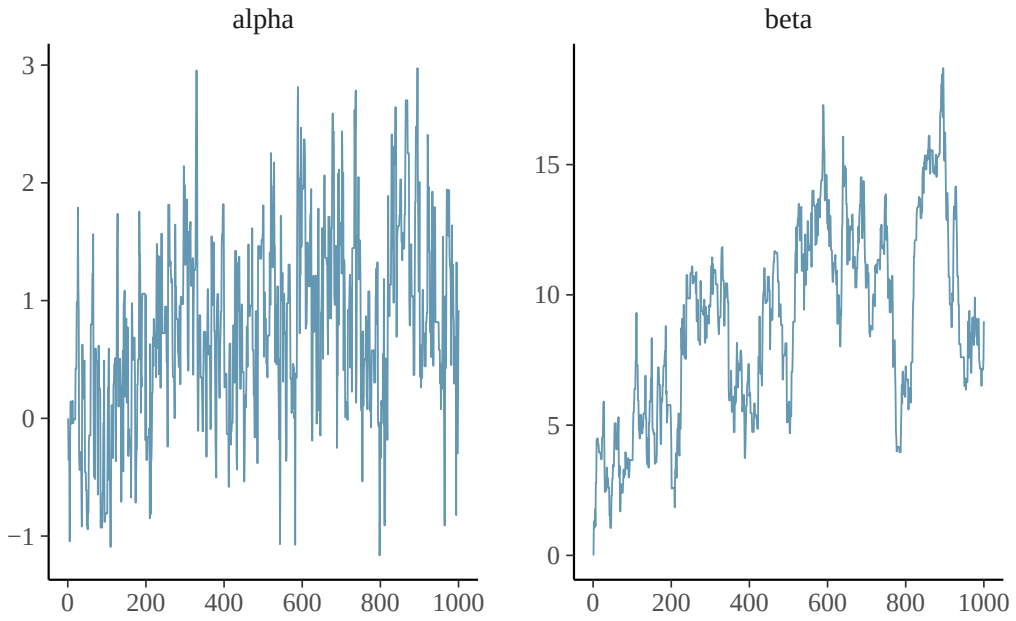
5. Report the warm-up length (see BDA Section 11.4, and Lecture 5.2).
6. The number of Metropolis chains used. It is important that multiple Metropolis chains are run for evaluating convergence (see BDA Section 11.4, and Lecture 5.2).
7. Plot all chains for α in a single line-plot. Overlapping the chains in this way helps in visually assessing whether chains have converged or not.
8. Do the same for β.

Write your answers/code here!

Have a look at `bayesplot` trace plot examples and tune your plot if wanted/needed. Don't forget to include a title/caption/description.

The below example plot only includes a single chain, but your report should include a plot with multiple chains overlayed!

```
# Useful functions: mcmc_trace (from bayesplot)
mcmc_trace(df, pars=c("alpha", "beta"))
```



Subtask 2.c)

In complex scenarios, visual assessment is not sufficient and $\widehat{R}$ is a more robust indicator of convergence of the Markov chains. Use $\widehat{R}$ for convergence analysis. You can either use Eq. (11.4) in BDA3 or the more recent version described in the article *Rank-normalization, folding, and localization: An improved $\widehat{R}$ for assessing convergence of MCMC*. You should specify which $\widehat{R}$ you used. In R the best choice is to use function `rhat_basic()` from the package `posterior` (this function implements the version described in the above mentioned article). Remember to remove the warm-up sample before computing $\widehat{R}$. Report the $\widehat{R}$ values for α and β separately. Report the values for the proposal distribution you finally used.

1. Describe briefly in your own words the basic idea of $\widehat{R}$ and how to to interpret the obtained $\widehat{R}$ values.
2. Tell whether you obtained good $\widehat{R}$ with first try, or whether you needed to run more iterations or how did you modify the proposal distribution.

Write your answers/code here!

```
# Useful functions: rhat_basic (from posterior)
```

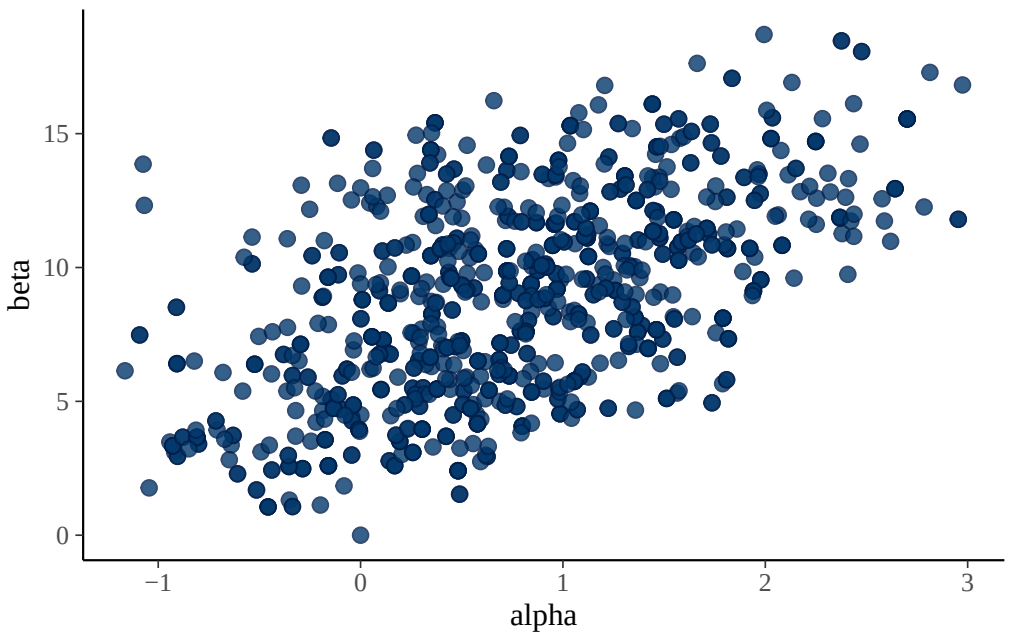
Subtask 2.d)

Plot the draws for α and β (scatter plot) and include this plot in your report. You can compare the results to BDA3 Figure 3.3b to verify that your code gives sensible results. Notice though that the results in Figure 3.3b are generated from the posterior with a uniform prior, so even when if your algorithm works perfectly, the results will look slightly different (although fairly similar).

Write your answers/code here!

Have a look at `bayesplot` scatter plot examples and tune your plot if wanted/needed. Don't forget to include a title/caption/description.

```
# Useful functions: mcmc_scatter (from bayesplot)
mcmc_scatter(df, pars=c("alpha", "beta"))
```



Rubric

- Is the implementation of `density_ratio` function included ?
  - No
  - Yes
- Is the implementation of `metropolis_bioassay` function included ?
  - No

  - Yes

- 2 a) Is the brief description of Metropolis-Hastings algorithm included (and it's not complete nonsense)? Provide also a brief comment on how clear you think that description is (and potentially mention errors if you see them).

  - No
  - Yes

- 2 b) Is the proposal/jumping distribution reported?

  - No
  - Yes

- 2 c) Are the starting points or the mechanism how they are generated reported?

  - No
  - Yes

- 2 d) Is the number of draws per chain reported?

  - No
  - Yes

- 2 e) Is the warm-up length reported?

  - No
  - Yes

- 2 f) Is the number of chains reported?

  - No
  - Yes

- 2 g) and 2 h) Are line plots of the chains included? (Separate plots for alpha and beta)

  - No plots are included
  - Yes, but both plots are in a single figure, or the plots are scatter plots (scatter plots aren't useful for visual convergence evaluation).
  - Yes, but only a plot for alpha or beta is included.
  - Yes, separate line plots for both alpha and beta are included.

- Is there a discussion on the convergence of the chains?

  - No discussion on convergence.
  - Yes, but the discussion is not convincing.
  - Yes, discussed in the report.

- Is it mentioned which implementation of Rhat is used? Two possible ways to compute R-hat would be:

  1.
  2.

  It is OK as long as it is mentioned (or evident from the code) which of the above is used.

  - No

  - Yes

- Is the brief description of Rhat included (and it's not complete nonsense)? Provide also a brief comment on how clear you think that description is (and potentially mention errors if you see them).
  - No
  - Yes

- Are the Rhat-values for alpha and beta reported?
  - No
  - Yes, but incorrectly computed
  - Yes, but computed separately for each chain
  - Yes, but only for alpha or beta
  - Yes, single values both for alpha and beta

- Is the interpretation of R-hat values correct ()?
  - No interpretation or discussion about the R-hat values, or conclusions clearly wrong
  - Interpretation somewhat correct
  - Interpretation correct

- Does the report contain a scatter plot about the draws? Do the results look reasonable, that is, roughly like in the Figure below ?
  - No plot included
  - Plot included, but the results do not look like in the figure above
  - Plot included, and the results look roughly like in the figure above

markmyassignment

The following will check the functions for which `markmyassignment` has been set up:

```
mark_my_assignment()
```

```
v | F W  S  OK | Context

/ |          0 | task-1-subtask-1-tests
/ |          0 | density_ratio()
v |          6 | density_ratio()

== Results =====================================================================
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 6 ]
Everything's correct!
```

3 Overall quality of the report

Rubric

- Does the report include comment on whether AI was used, and if AI was used, explanation on how it was used?
  - No
  - Yes
- Does the report follow the formatting instructions?
  - Not at all
  - Little
  - Mostly
  - Yes
- In case the report doesn't fully follow the general and formatting instructions, specify the instructions that have not been followed. If applicable, specify the page of the report, where this difference is visible. This will help the other student to improve their reports so that they are easier to read and review. If applicable, specify the page of the report, where this difference in formatting is visible.
- Please also provide feedback on the presentation (e.g. text, layout, flow of the responses, figures, figure captions). Part of the course is practicing making data analysis reports. By providing feedback on the report presentation, other students can learn what they can improve or what they already did well. You should be able to provide constructive or positive feedback for all non-empty and non-nonsense reports. If you think the report is perfect, and you can't come up with any suggestions how to improve, you can provide feedback on what you liked and why you think some part of the report is better than yours.