Assignment 4

Introduction to Bayesian Data Analysis 2025

Ana Sofia Acevedo (03805441)

Preamble

- **Points**: Assignment 4 comprises of 5 tasks, 2 points for Tasks 1.1-2.2 each and 4 points for Task 3 (12 in total). Full points are obtained for complete and correct answers—a proportion of the points are obtained for a proper approach or if only part of the task is solved.
- **Submission**: Hand in the assignment as a PDF `Markdown` report. The report should show the results, the code that produced the results, and additional text or comment. The report should appear clean and be uploaded on Moodle until Wednesday, July 7, 9:45 am.
- **Collaboration**: Reports can be handed in as team work (max. 2 people). When working in teams, declare this on page 2. However, each collaborator needs to hand in a report via Moodle, stating their name, student number (p. 1), and their machine specification (p. 2).
- **Permitted and Prohibited**: You may use materials from this class (e.g., slides, code on GitHub) as well as online forums such as Stack Overflow to write your code. However, you are not allowed to post questions from the assignment online or prompt them (including paraphrases) to LLMs/chatbots. All use of LLMs/chatbots is generally not allowed. Solutions may not be shared with other students from the class (except 1 potential collaborator).

`ulam` and `Quarto`

Running MCMC with `ulam()` takes some time and produces many messages. This can be be annoying when you repeatedly render the `Quarto` document and the goal is to have a clean report. Here are two tips to avoid an ugly document and refitting the model every time you re-render the document.

1) Write the ulam() model in a separate code chunk with the `echo`, `eval`, `output` settings as below to avoid printing the MCMC progress messages of `ulam()` in the PDF document.
2) Use the `file` argument in `ulam()` to fit the model only once and store the fitting results for reuse. In the code below, `ulam()` first looks into the folder `assignment_4` for a file `m1.rds`–the model fit object `m1`. If the file exists, it loads this file without fitting the model again. If the file does not exist yet, `ulam()` fits the model and stores the fitted object in the respective folder, so it can be reused later.

```
#| echo: true
#| eval: true
#| output: false
```

```
m1 <- ulam(
  model ,
  data= ... ,
  chains = ... ,
  cores = ... ,
  file = here("assignment_4", "m1")
)
```

Authorship Information

1. Declaration of Collaboration

☐ Yes (Collaborator name)
☒ No

2. Declaration of Authorship

☒ I certify that this assignment represents my own work. I have not used any unauthorized or unacknowledged aids as stated in the preamble, including free or commercial systems or services offered on the internet or text generating systems embedded into software. I did not copy code from someone else nor did I share my code with someone else.

3. System Information

☒ I confirm that I generated the submitted PDF report myself using R version 4.5.0 (2025-04-11 ucrt) and `Quarto/RMarkdown`.

Machine stamp: x86_64-w64-mingw32/x64

Timestamp: 2025-07-10 22:00:45 CEST

```
library(rethinking)
```

```
Loading required package: cmdstanr
```

```
This is cmdstanr version 0.9.0
```

```
- CmdStanR documentation and vignettes: mc-stan.org/cmdstanr
```

```
- CmdStan path: C:/Users/57314/.cmdstan/cmdstan-2.36.0
```

```
- CmdStan version: 2.36.0
```

```
Loading required package: posterior
```

```
This is posterior version 1.6.1
```

```

Attaching package: 'posterior'
```

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

```
    mad, sd, var
```

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

```
    %in%, match
```

```
Loading required package: parallel
```

```
rethinking (Version 2.42)
```

```

Attaching package: 'rethinking'
```

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

```
    rstudent
```

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.2     v tibble    3.2.1
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.0.4

-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
x purrr::map()    masks rethinking::map()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
library(here)
```

```
here() starts at C:/Users/57314/BayesIntro25_forked
```

```
library(ggplot2)
```

Task Set 1

For Tasks 1.1 and 1.2, consider the data generating processes for two groups of people below. Each of the `i=1,...,N` (`N=500`) persons belongs to either `Group 1` or `Group 2` and either succeeded in a task (`outcome=1`) or not (`outcome=0`).

```
N <- 500 # number of people
group <- rbinom(N, 1, prob=.5) # determine group membership
outcome <- rbinom(N, 1, prob=ifelse(group==0, .2, .5)) # determine outcome
sim <- data.frame(person=1:N, group, outcome)
```

Task 1.1

Model the outcome data with a Bernoulli likelihood using an *indicator* variable for group membership. Plot the posterior distribution of the *probability of success* for each group separately.

Answer

To use a Bernoulli likelihood, we work with the data for each individual.

```
m1_indicator <- ulam(
  alist(
    # likelihood
    outcome ~ dbern(p) ,
    logit(p) <- a + b*group ,

    # priors
    a ~ dnorm(0,1.5) ,
    b ~ dnorm(0,1.5)

  ) ,
  data=sim,
  chains = 4,
  cores = 4,
  #file = "my_scripts/my_models/assignment4_m1_indicator"
)
```

```
# Results analysis
precis(m1_indicator, depth = 2)
```

```
       mean         sd      5.5%     94.5%     rhat ess_bulk
a -1.370491 0.1672792 -1.648388 -1.119601 1.005764 504.3189
b  1.354536 0.2055853  1.033233  1.689900 1.004344 513.7667
```

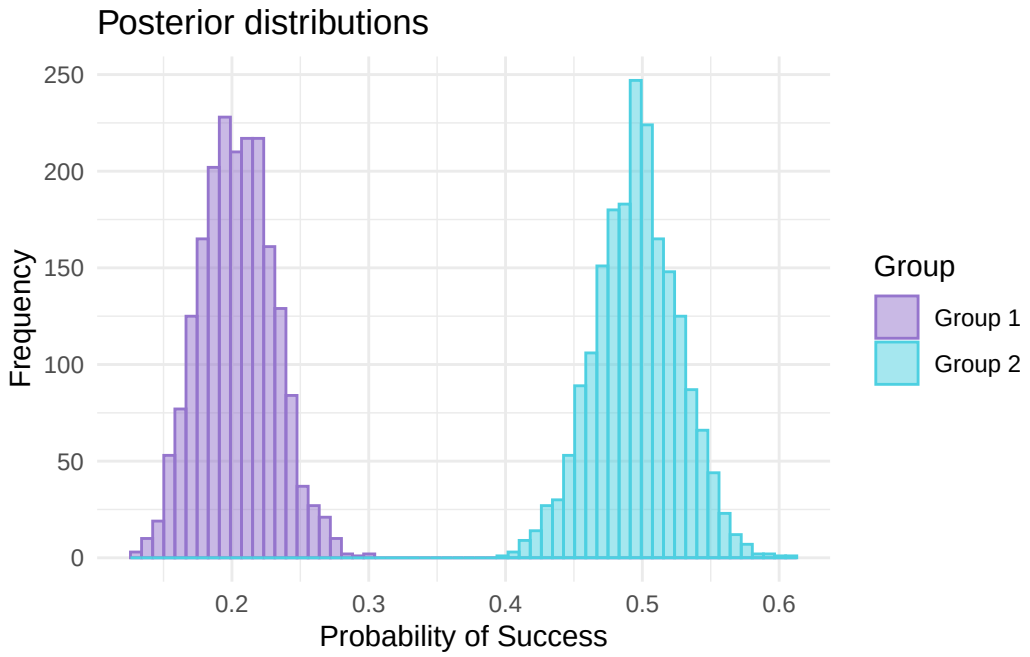
```
#alpha (mean probability for group 1)
inv_logit(coef(m1_indicator)[1])
```

```
        a
0.2025405
```

```
#alpha+beta (mean probability for group 2)
inv_logit(sum(coef(m1_indicator)))
```

```
[1] 0.4960112
```

```r
samp_m1_indicator <- extract.samples(m1_indicator)
m1_indicator.post <- data.frame(
  post_g1 = inv_logit(samp_m1_indicator$a),
  post_g2 = inv_logit(samp_m1_indicator$a + samp_m1_indicator$b)
)


ggplot() +
  geom_histogram(data = m1_indicator.post, aes(x = post_g1, fill = "Group 1"),
                 alpha = 0.5, color = "#9575CD", bins = 60) +
  geom_histogram(data = m1_indicator.post, aes(x = post_g2, fill = "Group 2"),
                 alpha = 0.5, color = "#4DD0E1", bins = 60) +
  scale_fill_manual(values = c("Group 1" = "#9575CD", "Group 2" = "#4DD0E1")) +
  labs(x = "Probability of Success",
       y = "Frequency",
       fill = "Group", title = "Posterior distributions") +
  theme_minimal()
```



Task 1.2

Model the outcome data with a Binomial likelihood using an *index* variable for group membership. Plot the posterior distribution of the *differences in the probability of success* between the

two groups—i.e., the 95% percentile interval should exclude 0 if there is a credible difference.

Answer

To use a Binomial likelihood, we work with the aggregated data, where the number of trials 'N' is included.

```
sim_wide <- list(
  group = 1:2,
  outcome = as.integer(tapply(sim$outcome, sim$group, sum)),
  total = as.integer(tapply(sim$outcome, sim$group, length))
)
view(sim_wide)
```

```
m1_index <- ulam(
  alist(
    # likelihood
    outcome ~ dbinom(total, p) ,
    logit(p) <- a[group]  ,

    # priors
    a[group] ~ dnorm(0,2)

  ) ,
  data=sim_wide,
  chains = 4,
  cores = 4,
  #file = "my_scripts/my_models/assignment4_m1_index"
)
```

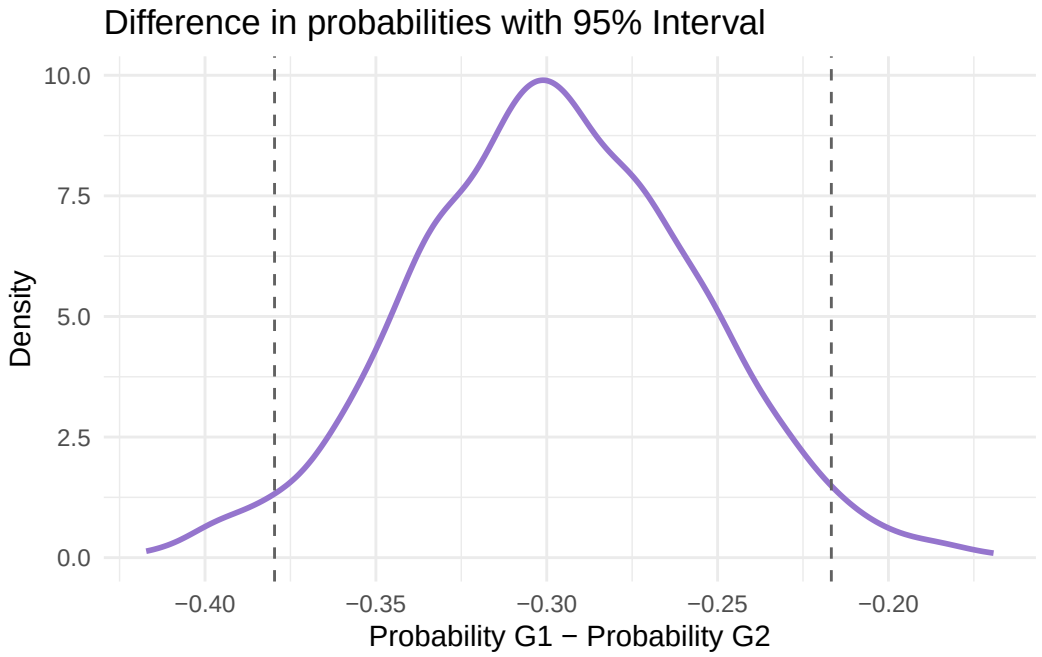
```
# Result analysis
precis(m1_index, depth = 2)
```

```
             mean        sd      5.5%      94.5%     rhat ess_bulk
a[1] -1.383926385 0.1606015 -1.641391 -1.1239039 1.000498 1366.268
a[2] -0.003629467 0.1276574 -0.209462  0.2032685 1.003732 1152.617
```

```
inv_logit(coef(m1_index))
```

```
     a[1]      a[2]
0.2003791 0.4990926
```

```r
m1_index.post <- extract.samples(m1_index)
m1_index.post <-  tibble(p1 = plogis(m1_index.post$a[,1]) ,
                         p2 = plogis(m1_index.post$a[,2]) ,
                         diff = p1 - p2)

ci <- quantile(m1_index.post$diff, probs=c(0.025, 0.975))

ggplot(m1_index.post, aes(diff)) +
  geom_density(linewidth = 1, color = "#9575CD") +
  geom_vline(xintercept = ci, linetype = "dashed", color = "#616161") +
  labs(x = "Probability G1 - Probability G2",
       y = "Density",
       title = "Difference in probabilities with 95% Interval") +
  theme_minimal()
```



Task Set 2

Load the data set `heart.csv` to solve the Task Set 2.

```
heart <- read.csv(here("my_scripts/my_assignments", "heart.csv"))
dat_heart <- data.frame(
  CHD = heart$TenYearCHD,
  # man is 1, woman is 2
  G = ifelse(heart$male==1, 1, 2),
  # 1 the person has diabetes, 2 they dont have
  D = ifelse(heart$diabetes==1, 1, 2),
  A = heart$age
)
```

Task 2.1

Run a Bayesian logistic regression model to estimate the risk of men and women with and without diabetes to develop a coronary heart disease (TenYearCHD). Does diabetes increase the risk for men or women?

Answer

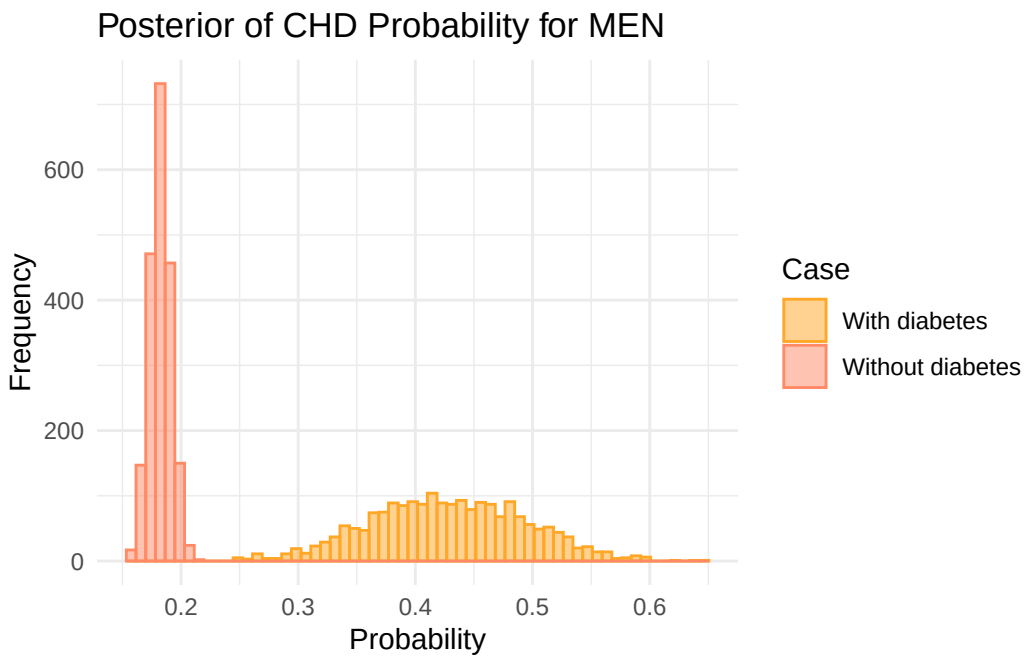
The dependent variable is coronary heart disease, and the independent variables are diabetes and gender.

```
m2_diabetes <- ulam(
  alist(
    # likelihood
    CHD ~ dbern(p) ,
    logit(p) <- a[D, G],

    # priors
    matrix[D,G]:a ~ dnorm(0,1)
  ) ,
  data=dat_heart,
  chains = 4,
  cores = 4,
  #file = "my_scripts/my_models/assignment4_m2_diabetes"
)
```

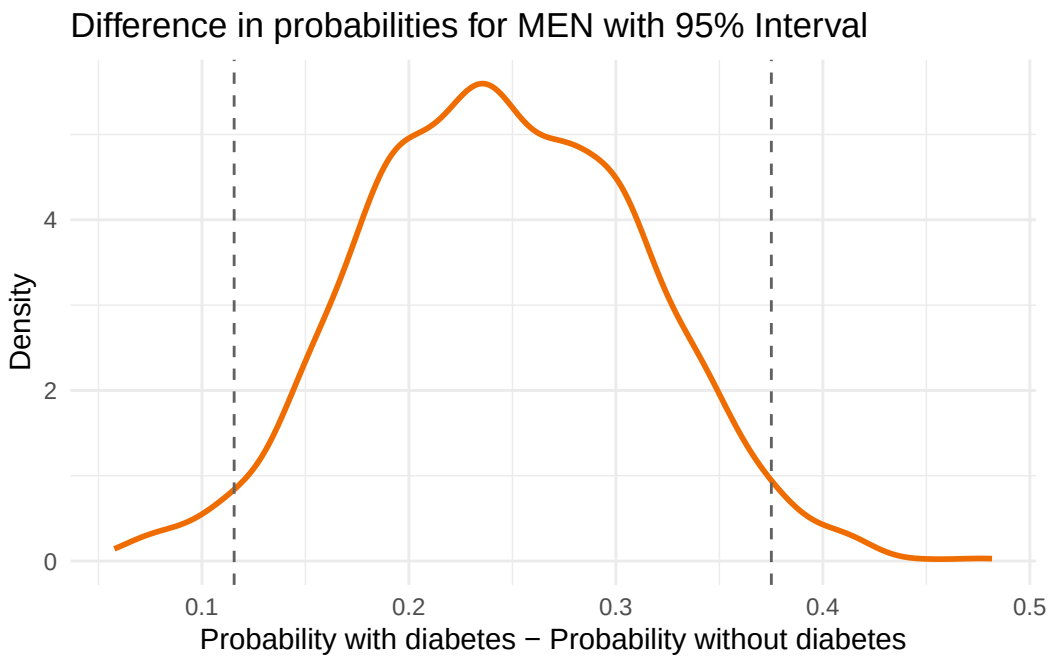
```r
# analyse the results here
#precis(m2_diabetes, depth=3)
df <- data.frame(case = c("D,M", "ND,M", "D,W",
                          "ND,W"),
                 prob = inv_logit(coef(m2_diabetes)))

df
```

```
       case      prob
a[1,1]  D,M 0.4262175
a[2,1] ND,M 0.1822155
a[1,2]  D,W 0.3258139
a[2,2] ND,W 0.1206085
```

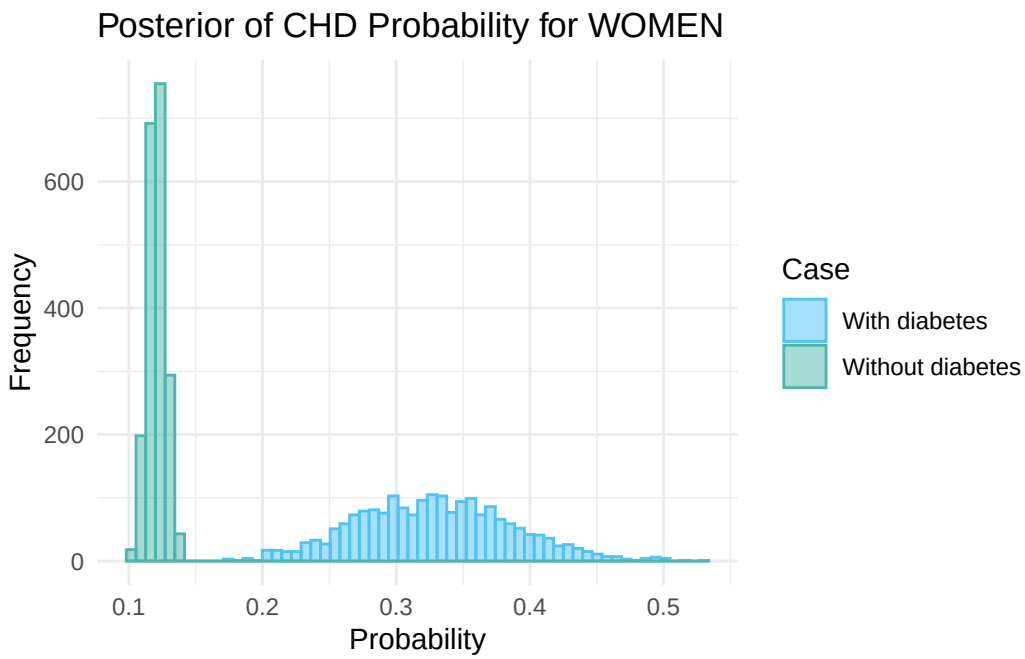
```r
#Extracting samples to then calculate the differences and check if there is a
#differential risk
m2_diabetes.post <- extract.samples(m2_diabetes)

# For MEN
m2_diabetes_men.post <-  tibble(p1 = plogis(m2_diabetes.post$a[,1,1]) ,
                     p2 = plogis(m2_diabetes.post$a[,2,1]) ,
                     diff = p1 - p2)
ci_men <- quantile(m2_diabetes_men.post$diff, probs=c(0.025, 0.975))

ggplot() +
  geom_histogram(data = m2_diabetes_men.post,
                 aes(x = p1, fill = "With diabetes"),
                 alpha = 0.5, color = "#FFA726", bins = 60) +
  geom_histogram(data = m2_diabetes_men.post,
                 aes(x = p2, fill = "Without diabetes"),
                 alpha = 0.5, color = "#FF8A65", bins = 60) +
  scale_fill_manual(values = c("With diabetes" = "#FFA726",
                               "Without diabetes" = "#FF8A65")) +
  labs(
    x = "Probability",
    y = "Frequency",
    fill = "Case",
    title = "Posterior of CHD Probability for MEN"
  ) +
  theme_minimal()
```



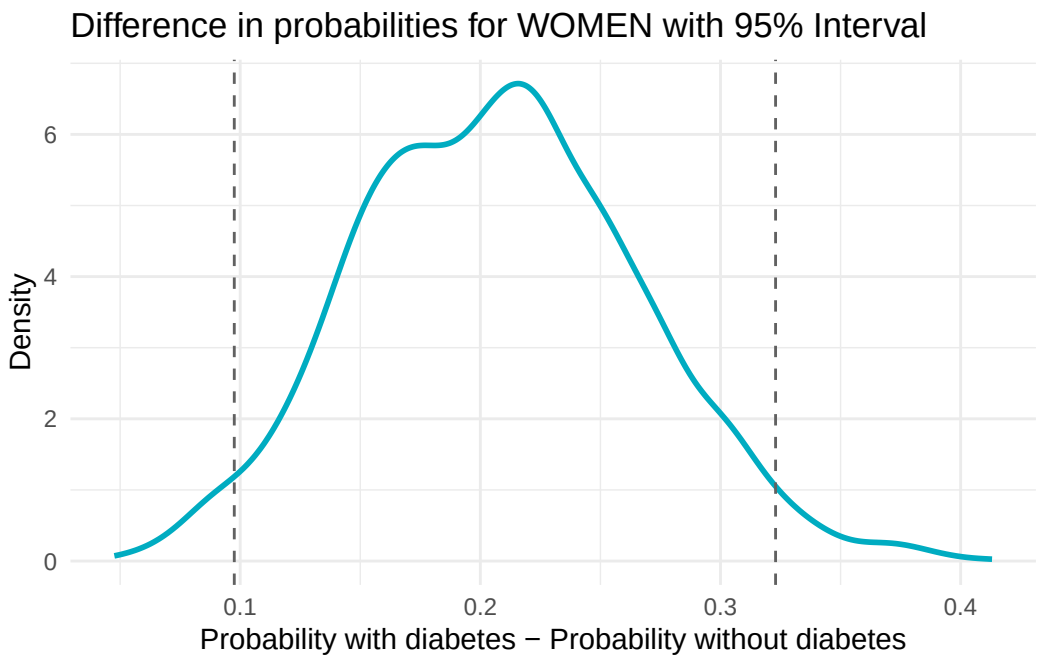

```
ggplot(m2_diabetes_men.post, aes(diff)) +
  geom_density(linewidth = 1, color = "#EF6C00") +
  geom_vline(xintercept = ci_men, linetype = "dashed", color = "#616161") +
  labs(x = "Probability with diabetes - Probability without diabetes",
       y = "Density",
       title = "Difference in probabilities for MEN with 95% Interval") +
  theme_minimal()
```




```
# For WOMEN
m2_diabetes_women.post <-  tibble(p1 = plogis(m2_diabetes.post$a[,1,2]) ,
                     p2 = plogis(m2_diabetes.post$a[,2,2]) ,
                     diff = p1 - p2)
ci_women <- quantile(m2_diabetes_women.post$diff, probs=c(0.025, 0.975))

ggplot() +
  geom_histogram(data = m2_diabetes_women.post, aes(x = p1,
                 fill = "With diabetes"),
                 alpha = 0.5, color = "#4FC3F7", bins = 60) +
  geom_histogram(data = m2_diabetes_women.post, aes(x = p2,
                 fill = "Without diabetes"),
                 alpha = 0.5, color = "#4DB6AC", bins = 60) +
  scale_fill_manual(values = c("With diabetes" = "#4FC3F7",
                    "Without diabetes" = "#4DB6AC")) +
  labs(
    x = "Probability",
    y = "Frequency",
    fill = "Case",
    title = "Posterior of CHD Probability for WOMEN"
  ) +
  theme_minimal()
```




```
ggplot(m2_diabetes_women.post, aes(diff)) +
  geom_density(linewidth = 1, color = "#00ACC1") +
  geom_vline(xintercept = ci_women, linetype = "dashed", color = "#616161") +
  labs(x = "Probability with diabetes - Probability without diabetes",
       y = "Density",
       title = "Difference in probabilities for WOMEN with 95% Interval") +
  theme_minimal()
```



Task 2.2

Run a Bayesian logistic regression model to estimate the effect of age on the risk of developing a coronary heart disease (TenYearCHD), separately for women and men. Ensure that the regression intercept represents the risk of women and men with average age. Does age increase the risk for men or women?

Answer

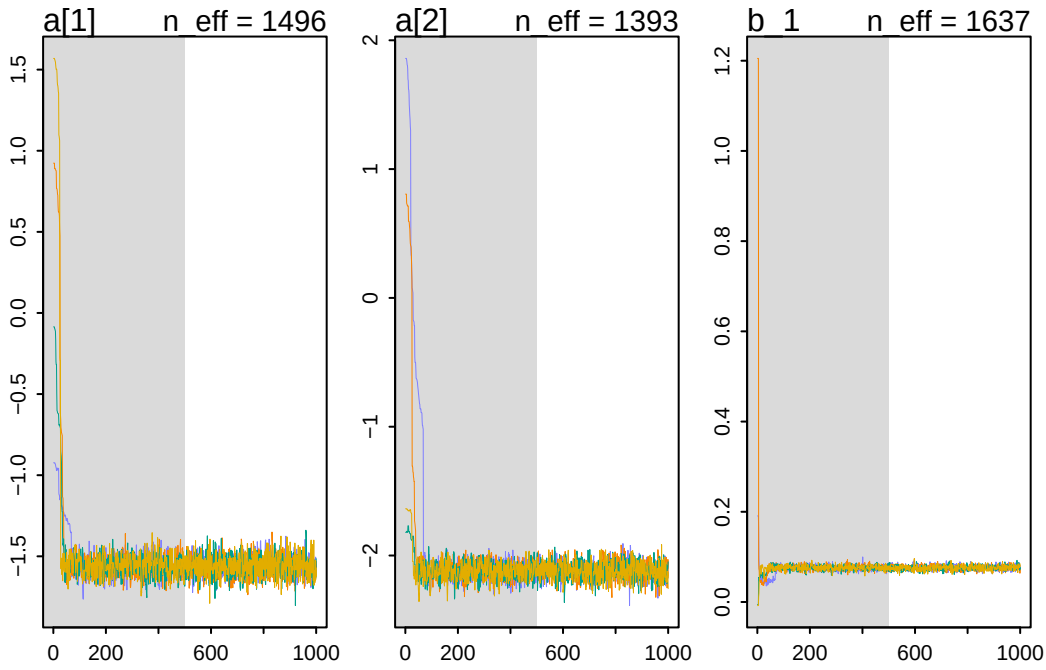
The dependent variable is coronary heart disease, and the independent variables are age and gender. The model has to be centered around the age variable. I tried using both indicator and index, considering that for both of these the gender variable has to be labeled differently for the package to work.

```
dat_heart$A_c <- dat_heart$A - mean(dat_heart$A)
```

```
m2_gender_index <- ulam(
  alist(
    # likelihood
    CHD ~ dbern(p),
    logit(p) <- a[G]+b_1*(A_c),
```

```
    # priors
    a[G] ~ dnorm(0,1),
    b_1 ~ dnorm(0,1)

  ) ,
  data=dat_heart,
  chains = 4,
  cores = 4,
  #file = "my_scripts/my_models/assignment4_m2_gender_index"
)



# Changing the labels of the gender variable to be able to implement it as
# an indicator, gender is 1 if it is a man, 0 if woman, so women is the
# base group
dat_heart$G <- ifelse(dat_heart$G==1,1,0)

m2_gender_indicator <- ulam(
  alist(
    # likelihood
    CHD ~ dbern(p) ,
    logit(p) <- a+b_1*(A_c)+b_2*G,

    # priors
    a ~ dnorm(0,1),
    b_1 ~ dnorm(0,1),
    b_2 ~ dnorm(0,1)

  ) ,
  data=dat_heart,
  chains = 4,
  cores = 4,
  #file = "my_scripts/my_models/assignment4_m2_gender_indicator"
)
```

```
# analyse the results here
# Using the index approach
traceplot(m2_gender_index)
```



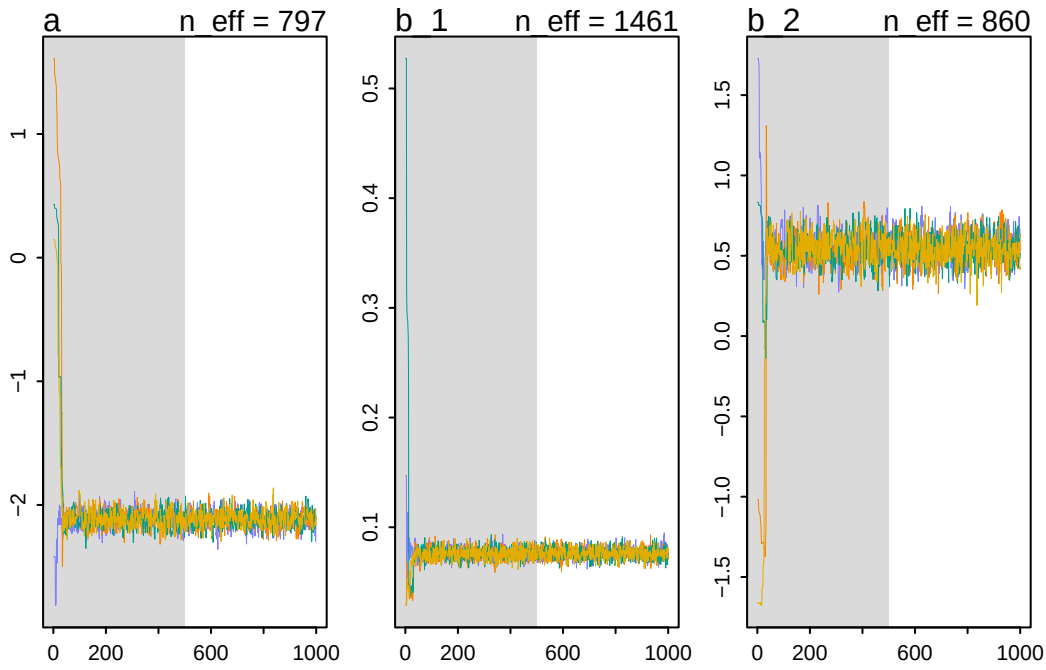

```
precis(m2_gender_index, depth = 3)
```

```
            mean          sd        5.5%       94.5%     rhat ess_bulk
a[1] -1.55938079 0.064271845 -1.66298320 -1.45556725 1.002091 1496.323
a[2] -2.12063703 0.068946526 -2.22928410 -2.00724150 1.002279 1392.604
b_1   0.07603396 0.005370218  0.06757089  0.08449134 1.004433 1636.845
```

```
#a[1] is the intercept for men, a[2] the intercept for women,
# and b_1 age coefficient
inv_logit(coef(m2_gender_index))
```

```
     a[1]      a[2]       b_1
0.1737355 0.1071071 0.5189993
```

```
# Using the indicator approach
traceplot(m2_gender_indicator)
```




```
precis(m2_gender_indicator, depth = 3)
```

```
          mean           sd        5.5%       94.5%     rhat  ess_bulk
a   -2.1169696 0.065987087 -2.21845005 -2.01354855 1.006558  797.4891
b_1  0.0761978 0.005058649  0.06828923  0.08458777 1.002799 1461.0386
b_2  0.5467370 0.088727275  0.40318924  0.68782511 1.002946  860.1775
```

```
# intercept for men
inv_logit(coef(m2_gender_indicator)[1]+coef(m2_gender_indicator)[3])
```

```
        a
0.1721832
```

```
# intercept for women
inv_logit(coef(m2_gender_indicator)[1])
```

```
        a
0.1074584
```

```r
# age coefficient
inv_logit(coef(m2_gender_indicator)[2])
```

```
      b_1
0.5190402
```

Task 3

Assume that data is randomly generated from a normal distribution according to:

$$y_i \sim N(\mu = 100, \sigma = 10)\ .$$

Simulate $N = 500$ data points from this distribution. Write a MCMC (Metropolis) algorithm that fits a normal distribution to the simulated data and recovers the mean and standard deviation. Use 2000 iterations and discard the first 1,000 as warm-up. Plot the posterior distribution and the MCMC chain for both parameters.

Answer

Taking the analogy we did in class, the "islands" would be the parameter values, the "population sizes" are the (posterior) probabilities at each parameter value, an the "days" are samples taken from the joint posterior of the parameters in the model, in this case I guess the iterations.

Given that we are interested in maximizing the posterior probability, going back to Bayes Theorem, the posterior is proportional to the multiplication of the likelihood and the prior.

Then, the posterior $\propto$ likelihood * prior. Or, more detailed,

$$p(\theta|Data) \propto p(Data|\theta) * p(\theta)$$

Moreover, by assuming a normal distribution and taking the parameters as independent, then the joint prior will end up being the multiplication, and the previous expression for the posterior can be rewritten as

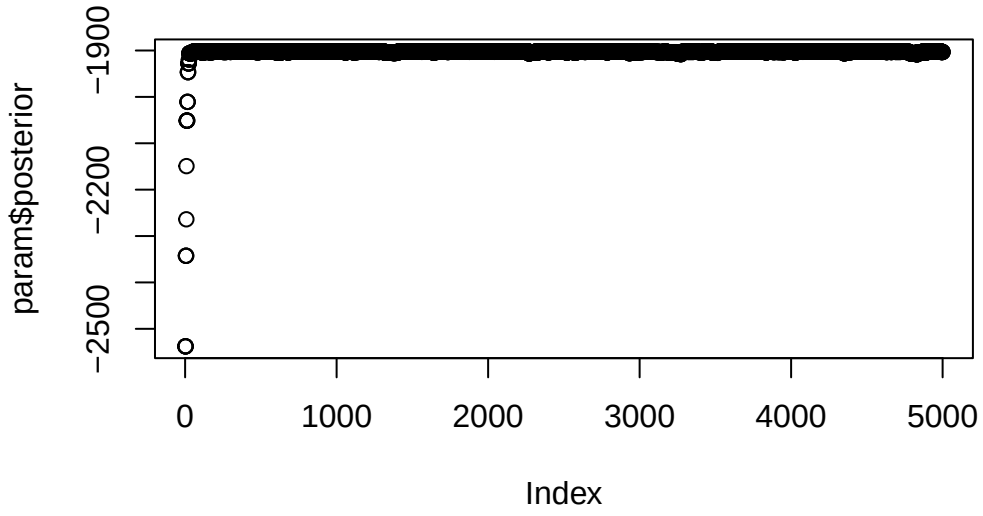
$$p(Data|\mu, \sigma) * p(\mu) * p(\sigma)$$

The likelihood function can be calculated according to the probability density function of an assumed distribution, in this case normal. Since the observations are taken to be independent, the likelihood will be the product of the density function for all observed data points.

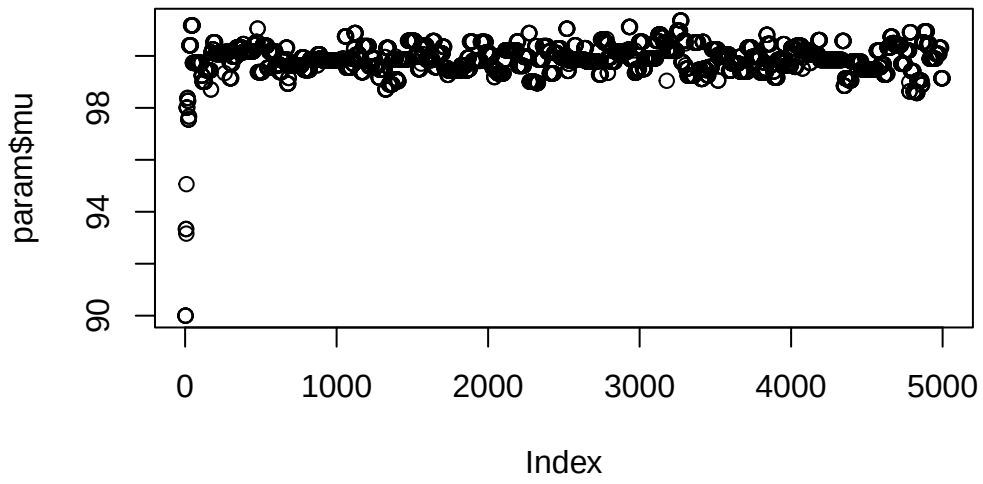
```
#simulated data
simu <- rnorm(500, mean=100, sd=10)

# likelihood function supposing the data comes from a normal distribution with
# unknown parameters, since the draws are independent, then the joint
# likelihood is the product of the individual results when evaluation the
# probability density function. Since the numbers are too small, working with
```

```r
# logarithms is better because the multiplication turns into an addition.
likelihood = function(data, mu, sigma) {
  sum(dnorm(data, mu, sigma, log = TRUE))
}

# Then some priors for both parameters are set for both unknown parameters.
# They mean is assumed to be normally distributed, and the standard deviation
# also but it is shifted a lot to the right to avoid it taking negative values
# as much as possible
# mu ~ N(90, 5)
# sd ~ N(15, 0.5)
mu_curr <- 90#rnorm(1, 90, 5)
sigma_curr <- 7#rnorm(1, 12, 2)


# Running the algorithm
iter <- 5000
param <- data.frame(mu = rep(0,iter), sigma = rep(0,iter),
                    posterior = rep(0,iter))

for (i in c(1:iter)){
  # recording the parameters
  param[i,"mu"] = mu_curr
  param[i,"sigma"] = sigma_curr

  # calculating posterior distribution of the current values of parameters
  likelihood_current <- likelihood(simu, mu_curr, sigma_curr)
  prior_mu_current <- dnorm(mu_curr, mean = 90, sd = 5, log = TRUE)
  prior_sigma_current <- dnorm(sigma_curr, mean = 12, sd = 2, log = TRUE)
  post_current <- likelihood_current + prior_mu_current + prior_sigma_current
  param[i,"posterior"] <- post_current

  # generate new proposal based on the proposed distribution
  mu_new <- rnorm(1, mu_curr, 5)
  sigma_new <- rnorm(1, sigma_curr, 1)

  # calculating posterior with the new proposal
  likelihood_new <- likelihood(simu, mu_new, sigma_new)
  prior_mu_new <- dnorm(mu_new, mean = 90, sd = 5, log = TRUE)
  prior_sigma_new <- dnorm(sigma_new, mean = 12, sd = 1, log = TRUE)
  post_new <- likelihood_new + prior_mu_new + prior_sigma_new
```

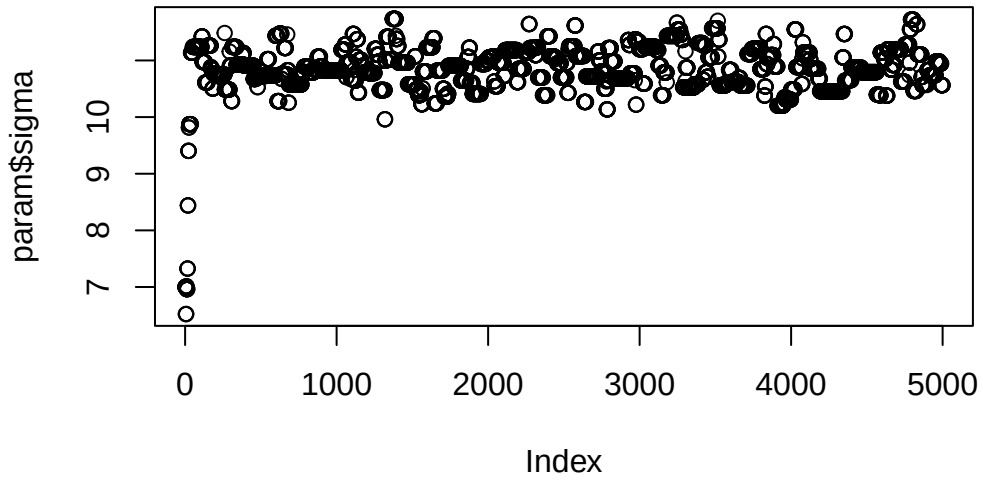
```
  # calculate the ratio to check if the new one is "visited" or not, since it
  # is based on the log scale, the ratio is calculated by subtraction. To turn
  # it into a comparable ratio with the algorithm it is reversed using "exp"
  ratio <- exp(post_new-post_current)
  if(ratio>runif(1)){
    mu_curr <- mu_new
    sigma_curr <- sigma_new
  }

}

# Trace plots considering all iterations to show where the chain started and
# where it converged



# Ignore the first 1000 as warm-up period for plotting the posterior
post_param <- tail(param, 1000)

# plot of the posterior probability for mu


plot(param$posterior)
```




```
plot(param$mu)
```



```
plot(param$sigma)
```



To do this task, I used the following resources:

Rampinelli, 2019

Eisenberg, 2017