Concepts of Bayesian Data Analysis

	Raïsa Carmen *s0204278*	**Barouyr Demirjian**
Group 25:	**Vancesca Dinh** *r0830510*	**Chaojie Huang** *r0816192*
	Arthur Terlinden	

1 Task A

Question 1

Here, we write question 1

2 Task B: Dose-response model

(1) Assume that the likelihood of the experiment is specified by

$$y \sim binomial(N, \pi)$$

$$logit(\pi) = \alpha + \beta d.$$

Here β is the parameter of interest. Take vague priors for α and β. Write jags, OpenBugs or Nimble code for this problem. Take 2 MCMC chains with different starting values, and check convergence with the appropriate techniques.

To denote the uncertainty of parameters α and β, we specify two vague priors as a massive variance (extremely small precision) for these two parameters. The first vague prior is:

$$\alpha \; N(0, 10000)$$

$$\beta \; N(0, 10000)$$

And the second vague prior is:

$$\alpha \; t(0, 0.0001, 5)$$

$$\beta \; t(0, 0.0001, 5)$$

Our parameter settings include running 2 MCMC chains with different starting values for α and β. The first chain starts with ($\alpha = 0$, $\beta = 0$), while the second chain starts with ($\alpha = -0.5$,$\beta = 0.1$). We run each chain for a total of 10000 iterations, with the first 5000 iterations used as a burn-in period.

To assess the convergence of the model with a prior normal distribution, we employed several diagnostic tools including trace plots, the Gelman-Rubin diagnostic test, and autocorrelation plots. We observed from the trace plots of α and β (refer to Figure 1) that the estimates from each chain quickly stabilized around a steady state. Additionally, both chains were found to converge around the same conclusion. Furthermore, we conducted the Gelman-Rubin diagnostic test, which showed that the estimated potential scale reduction factors of α and β were both 1. These results suggest that our model has converged well. Moreover, the Gelman-Rubin diagnostic plots (refer to Figure 1) support this conclusion, as both the potential scale reduction factors of α and β were found to decrease quickly and remain stable as the number of iterations increased. We also examined the autocorrelation plots (refer to Figure 2), which indicated low autocorrelation. The autocorrelation decreased and remained around zero as the lag number increased, indicating that the chains have mixed well. Overall, these diagnostic tools suggest that our model that the prior is normal distribution has converged well, and the inference based on the Markov chain Monte Carlo simulation is reliable.

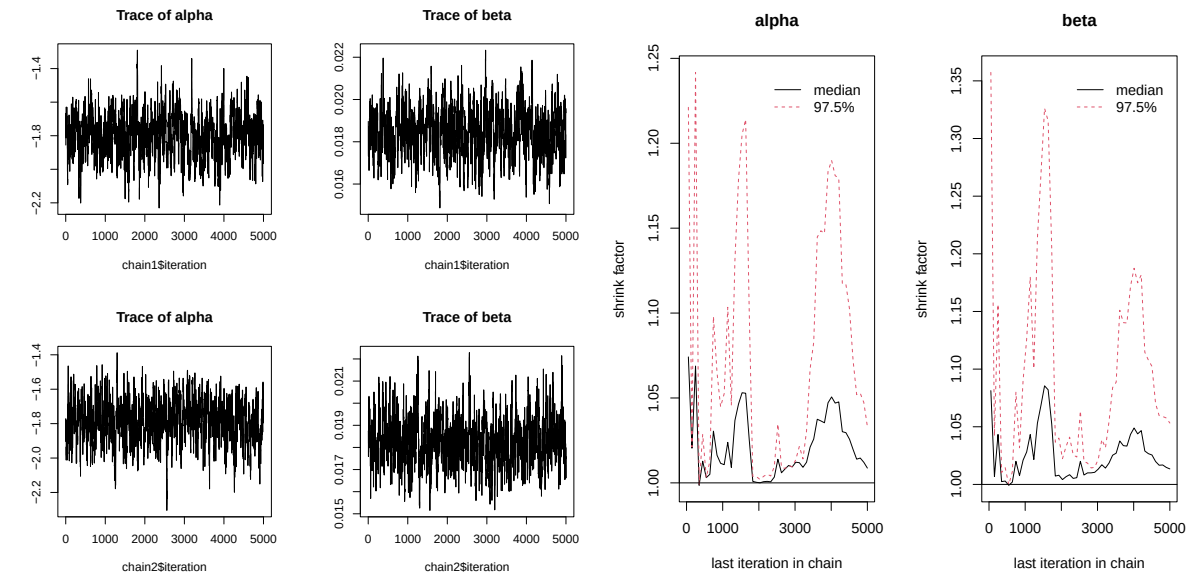

Figure 1: Trace and Gelman-Rubin diagnostic plots of α and β (prior: normal distribution)

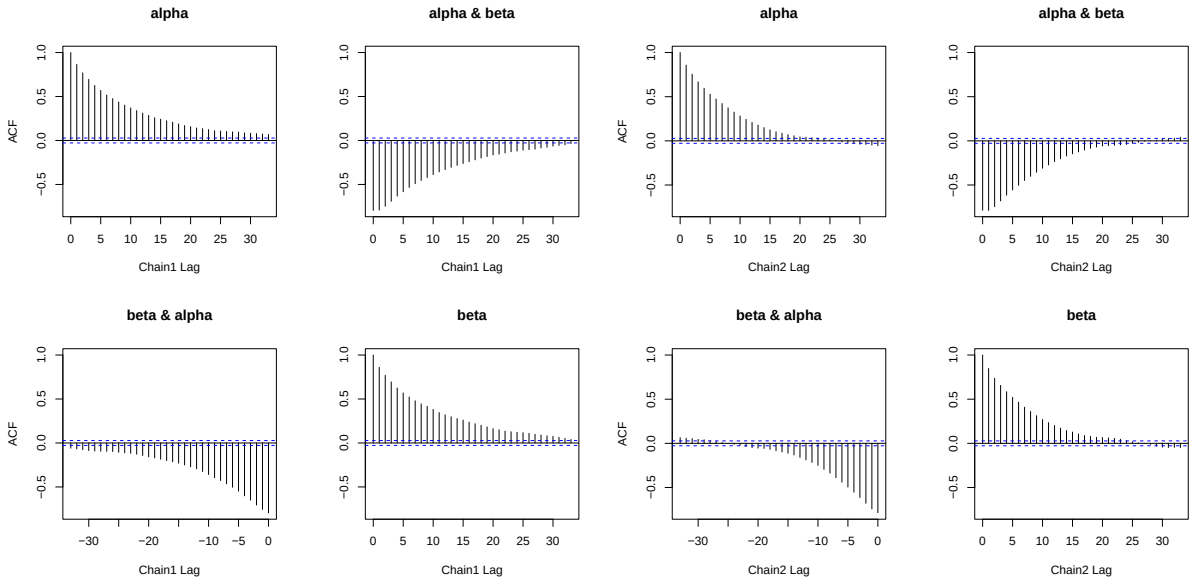

Figure 2: Autocorrelation plots of α and β (prior: normal distribution)

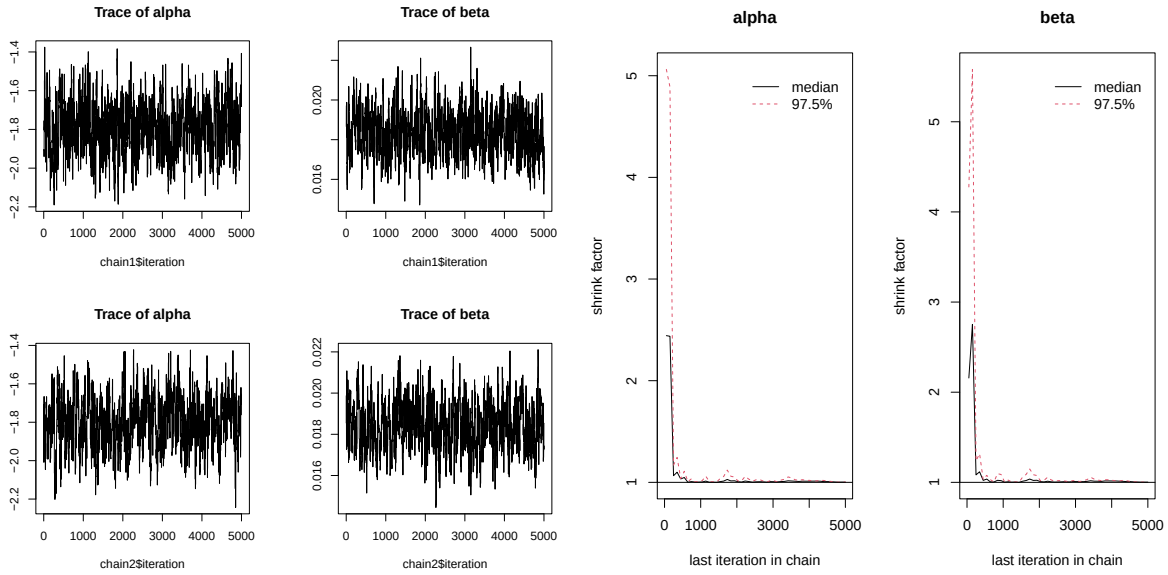

Figure 3: Trace and Gelman-Rubin diagnostic plots of α and β (prior: t-distribution)

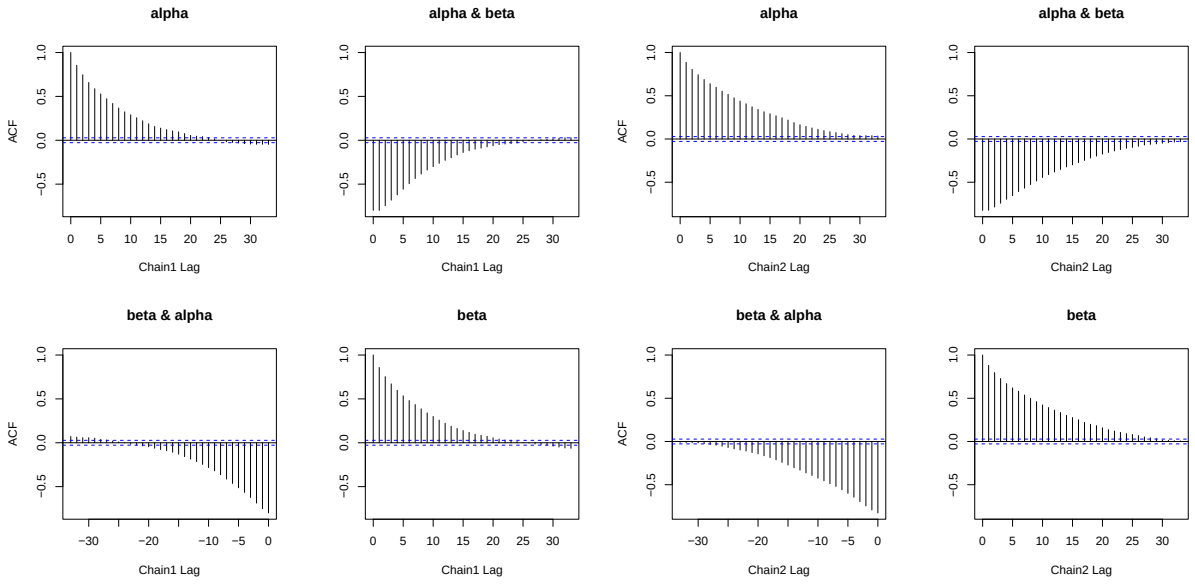

Figure 4: Autocorrelation plots of α and β (prior: t-distribution)

We analyzed the convergence of the model with a prior t-distribution using trace plots, autocorrelation plots, and Gelman-Rubin diagnostic plots, which are presented in Figure 3 and Figure 4. We observed that all the trends remained consistent with the plots obtained using a normal distribution for α and β. Moreover, the estimated potential scale reduction factors of α and β were found to be equal to 1, indicating good convergence of the model. These results provide compelling evidence that the model with a prior t-distribution has achieved good convergence and that the inference derived from the Markov chain Monte Carlo simulation is reliable.

(2) Summarize all results graphically and summarize with the usual Bayesian posterior measures. What do you conclude from these?

(3) Plot the posterior dose-response relationship together with the observed probabilities of a malformation per dose.

(4) A safe level of exposure can be defined as a dose corresponding to a very small increase in excess risk of q, e.g. q = 0.05. This is called the Benchmark dose (BMD) d^* and can be obtained by solving the equation

$$r(d^*) = \frac{P(d^*) - P(0)}{1 - P(0)} = q$$

with $P(d)$ the probability of an adverse effect at dose level d. For a logistic regression with a linear dose model, the BMD is given by

$$BMD = \frac{logit(q^*) - \alpha}{\beta}$$

with $q^* = q(1 - P(0)) + P(0)$. Determine the posterior estimate of the safe level of exposure for DYME corresponding with an excess risk of q = 0.05.

(5) As an alternative, a safe level of exposure can be obtained from a threshold model, defined as

$$y \sim binomial(N, \pi)$$

$$logit(\pi) = \alpha + \beta(d - \tau)I(d > \tau)$$

, with τ the threshold dose below which there is no excess risk. Write code for this model, and summarize the results. How do these results compare with previous results?

A Appendix

A.1 All code for the report

```
#---------------------------load all packages----------------------------------#
library(tidyverse)
library(rprojroot)
library(kableExtra)
library('nimble')
library(coda)
library(cowplot)

knitr::opts_chunk$set(echo = TRUE)
knitr::opts_knit$set(
  root.dir = find_root(criterion = has_file("BayesianInference.Rproj")))
#-----------------------------setup task B---------------------------------#
n <- 5
# Dose level
dose <- c(0,62.5,125,250,500)
# Number of fetus
N <- c(282,225,290,261,141)
# Number of malformations
y <- c(67,34,193,250,141)
#--------------------------------Question B1-------------------------------#
init1 <- list(alpha = 0, beta = 0)
init2 <- list(alpha = -0.5, beta = 0.1)
initial.values <- list(init1, init2)

# MCMC settings
n.iter <- 10000  # iterations
n.burnin <- 5000 # burn-in
n.chains <- 2    # chains

# Model settings
model.data <- list('dose' = dose, 'N' = N,'y' = y)
model.constant <- list('n' = n)

# Model 1
# A prior with Normal Distribution
model_1 <- nimbleCode({
  # Specify a vague prior with normal distribution
  alpha ~ dnorm(0, sd = 10000)
  beta ~ dnorm(0, sd = 10000)
  # likelihood
  for (i in 1:n) {
    logit(p[i]) <-  alpha + beta * dose[i]
    y[i] ~ dbin(p[i],N[i])
  }})

# Output of Model 1
mcmc.output1 <- nimbleMCMC(code = model_1,
                           data = model.data,
                           constants = model.constant,
                           inits = initial.values,
```

```
                          niter = n.iter,
                          nburnin = n.burnin,
                          summary = TRUE,
                          nchains = n.chains
)
# Trace plots
pdf("images/trace_normal.pdf")
par(mfrow = c(2,2))
traceplot(as.mcmc(mcmc.output1$samples$chain1), xlab = "chain1$iteration")
traceplot(as.mcmc(mcmc.output1$samples$chain2), xlab = "chain2$iteration")
dev.off()
# Gelman-Rubin diagnostic plots
combinedchains1 = mcmc.list(as.mcmc(mcmc.output1$samples$chain1), as.mcmc(mcmc.output1$samples$chain2))
gelman.diag(combinedchains1)
pdf("images/gelman_normal.pdf")
gelman.plot(combinedchains1,xlim = c(0,5000))
dev.off()
# Autocorrelation plots
par(mfrow = c(2, 2))
#densplot(as.mcmc(mcmc.output1$samples$chain1), main = "Chain1 Density of alpha")
#densplot(as.mcmc(mcmc.output1$samples$chain2), main = "Chain2 Density of alpha")
pdf("images/ac1_normal.pdf")
acf(as.mcmc(mcmc.output1$samples$chain1), xlab = 'Chain1 Lag')
dev.off()
pdf("images/ac2_normal.pdf")
acf(as.mcmc(mcmc.output1$samples$chain2), xlab = 'Chain2 Lag')
dev.off()
# Model 2
# A prior with t Distribution
model_2 <- nimbleCode({
  # Specify a vague prior with t distribution
  alpha ~ dt(0, 0.0001, 5)
  beta ~ dt(0, 0.0001, 5)
  # likelihood
  for (i in 1:n) {
    logit(p[i]) <-  alpha + beta * dose[i]
    y[i] ~ dbin(p[i],N[i])
  }})

# Output of Model 1
mcmc.output2 <- nimbleMCMC(code = model_2,
                           data = model.data,
                           constants = model.constant,
                           inits = initial.values,
                           niter = n.iter,
                           nburnin = n.burnin,
                           summary = TRUE,
                           nchains = n.chains
)

# Trace plots
pdf("images/trace_t.pdf")
par(mfrow = c(2,2))
traceplot(as.mcmc(mcmc.output2$samples$chain1), xlab = "chain1$iteration")
```

```
traceplot(as.mcmc(mcmc.output2$samples$chain2), xlab = "chain2$iteration")
dev.off()
# Autocorrelation plots
par(mfrow = c(2,2))
densplot(as.mcmc(mcmc.output2$samples$chain1), main = "Chain1 Density of alpha")
densplot(as.mcmc(mcmc.output2$samples$chain2), main = "Chain2 Density of alpha")

pdf("images/ac1_t.pdf")
acf(as.mcmc(mcmc.output2$samples$chain1), xlab = 'Chain1 Lag')
dev.off()
pdf("images/ac2_t.pdf")
acf(as.mcmc(mcmc.output2$samples$chain2), xlab = 'Chain2 Lag')
dev.off()
# Gelman-Rubin diagnostic plots
combinedchains2 = mcmc.list(as.mcmc(mcmc.output2$samples$chain1), as.mcmc(mcmc.output2$samples$chain2))
gelman.diag(combinedchains2)
pdf("images/gelman_t.pdf")
gelman.plot(combinedchains2,xlim = c(0,5000))
dev.off()
```