Introduction to Bayesian Inference

Group 13

Group members:

Calo, Theresse Joy

Colman, Roos

Merezhko, Maria

Noe, Sebastian

Ormando Aramburu, Santiago

Due date: 01.06.2022

Exercise No. 4

4.1 Model the data as using a nonlinear logistic growth function. Take flat uniform priors for the parameters α, β, γ and a noninformative prior for σ. Write out the Bayesian model (likelihood and priors).

Likelihood

$$lik(y|\alpha,\beta,\gamma) = \prod_{i}^{n} \frac{\alpha}{1+\beta^{\gamma \cdot x_i}} + \epsilon_i$$

Priors and prior assumptions

$$\alpha = \frac{1}{b-a}$$

$$\beta = \frac{1}{b-a}$$

$$\gamma = \frac{1}{b-a}$$

4.2 Run 5000 iterations, then look at history plots and autocorrelation plots of the sample traces and calculate the Gelman and Rubin convergence diagnostic for each of the parameters you have monitored. Do the simulations look like they have converged? If not, carry out some more updates and check again, until you are happy with convergence.

Since no information was available on the expected values for alpha, beta and gamma, we initially started with a uniform distribution from -10 to +10 for each of the three parameters. Since the estimated values for these parameters as well as the 95% credible intervals are far from -10 and +10, it was decided that the initial values and the distributions were appropriate.

Traceplots and autocorrelation plots for the model using 5000 iterations and 1000 burnins are displayed in 1 and 2, respectively. Results of the Gelman and Rubin diagnostics are displayed in table 1

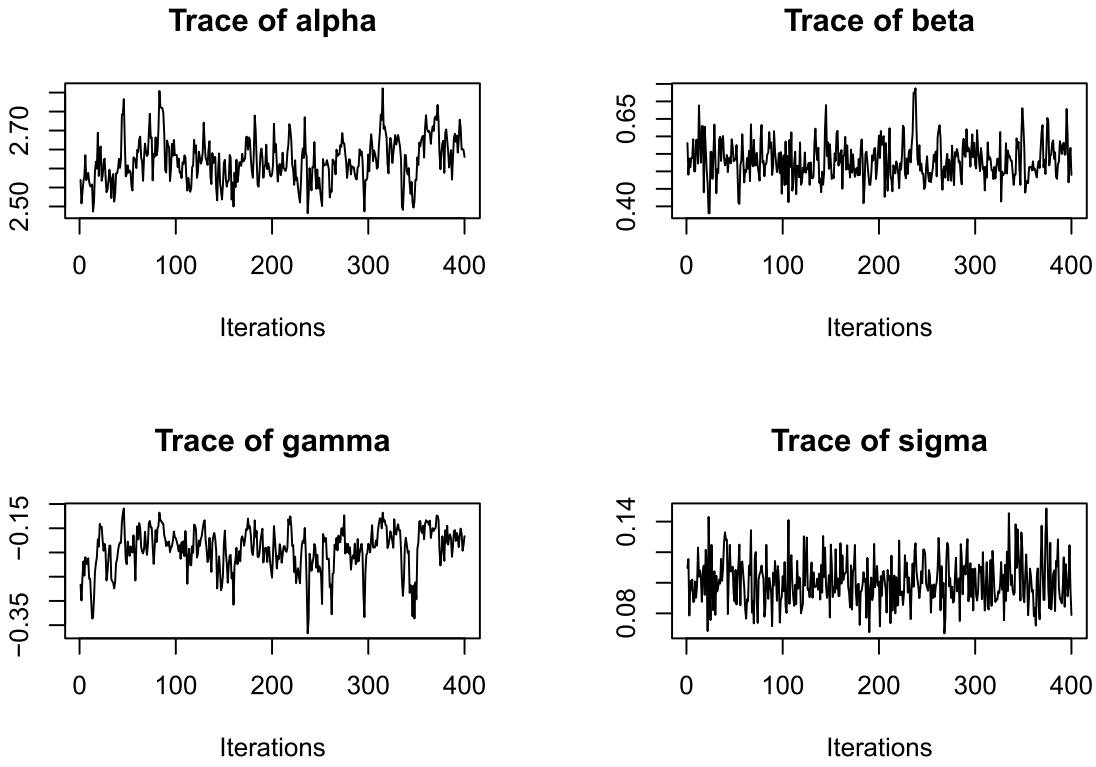


Figure 1: Traceplots of the parameters alpha, beta, gamma, and sigma after MCMC-sampling from the posterior distribution with 5,000 iterations and a burn-in of 1,000.

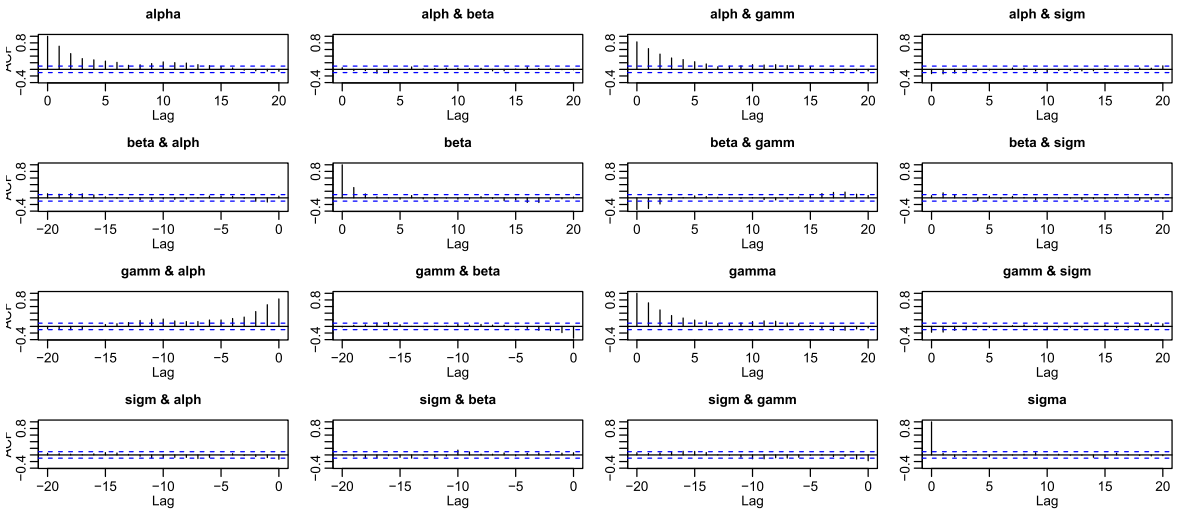


Figure 2: Plots of autocorrelation between the parameters in the model with 5,000 iterations and 1,000 burn-ins.

Table 1: Gelman and Rubin convergence diagnostics for each parameter in the model using 5,000 interations and 1,000 burnins.

	Point est.	Upper C.I.
alpha	1.012	1.024
beta	1.004	1.010
gamma	1.010	1.023
sigma	1.001	1.003

To further increase convergence for the estimates of the parameters, numbers of iterations and burnin were increased. The Gelman and Rubin diagnostics implied, that a burn-in of 4,000 is a reasonable choice, since scale-reduction is sufficient before 4,000 iterations were reached. Therefore, the results reported in the remainder of this document will be based on 9,000 iterations of which 4,000 are burn-ins. Traceplots and autocorrelation plots for the parameters in the model under these conditions can be found in figures 3 and 4, respectively. Gelman and Rubin convergence diagnostic for each of the parameters are displayed in table 2, while the mulivariate potential scale reduction factor was found to be 1.013; the corresponding plots for the Gelman and Rubin diagnostic each of the parameters are displayed in figure 5.

Table 2: Gelman and Rubin convergence diagnostics for each parameter in the model.

	Point est.	Upper C.I.
alpha	1.013	1.031
beta	1.009	1.015
gamma	1.022	1.043
sigma	1.002	1.004

4.3 Produce summary statistics and kernel density plots for the posterior samples of the regression parameters. Check the Monte Carlo standard error for each of the parameters to assess the accuracy of your estimates.

Summary statistics of the parameters for the posterior distribution can be found in table 3 while the plots for the density of the parameters of the posterior distribution are displayed in figure 6.

MCMC standard errors were 0.006, 0.003, 0.005, and 0.001 for α, β, γ, and σ, respectively.

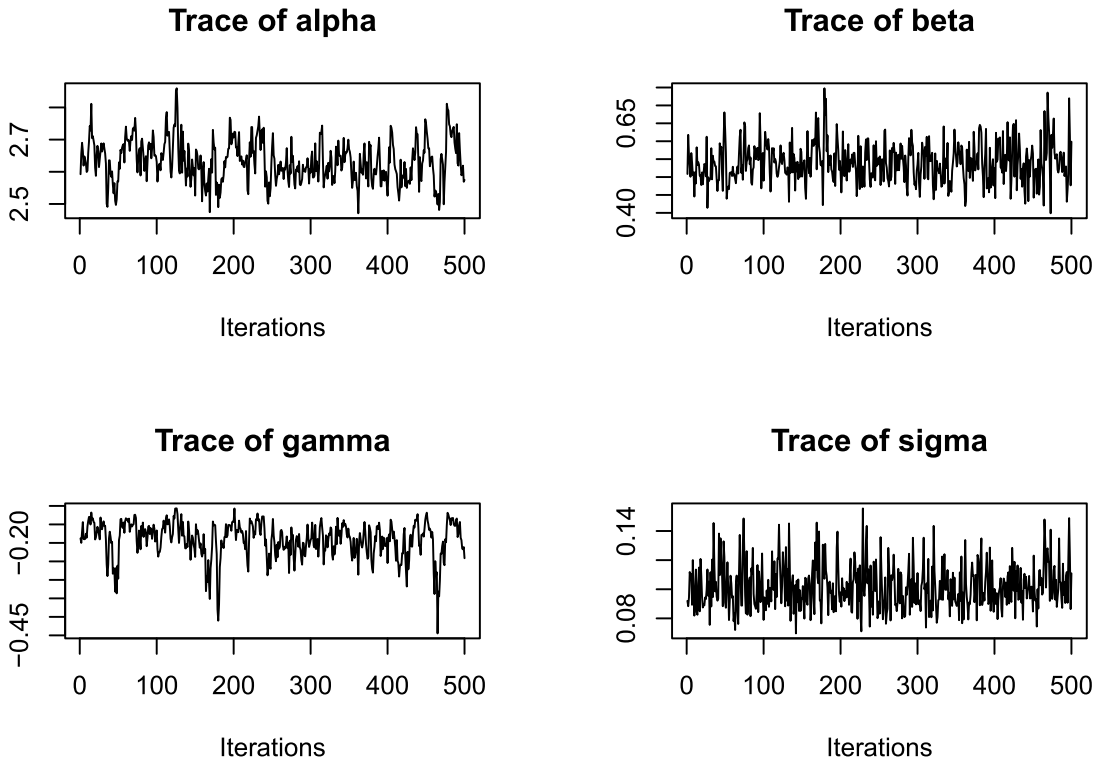


Figure 3: Traceplots of the parameters alpha, beta, gamma, and sigma after MCMC-sampling from the posterior distribution with 9,000 iterations and a burn-in of 4,000.

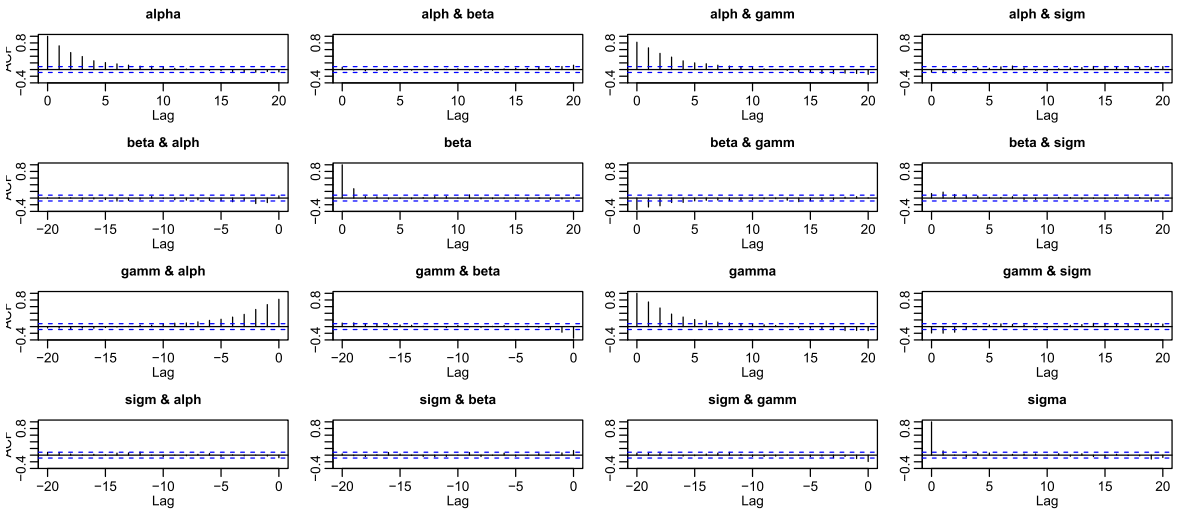


Figure 4: Plots of autocorrelation between the parameters in the model with 9000 iterations and 4000 burn-ins.

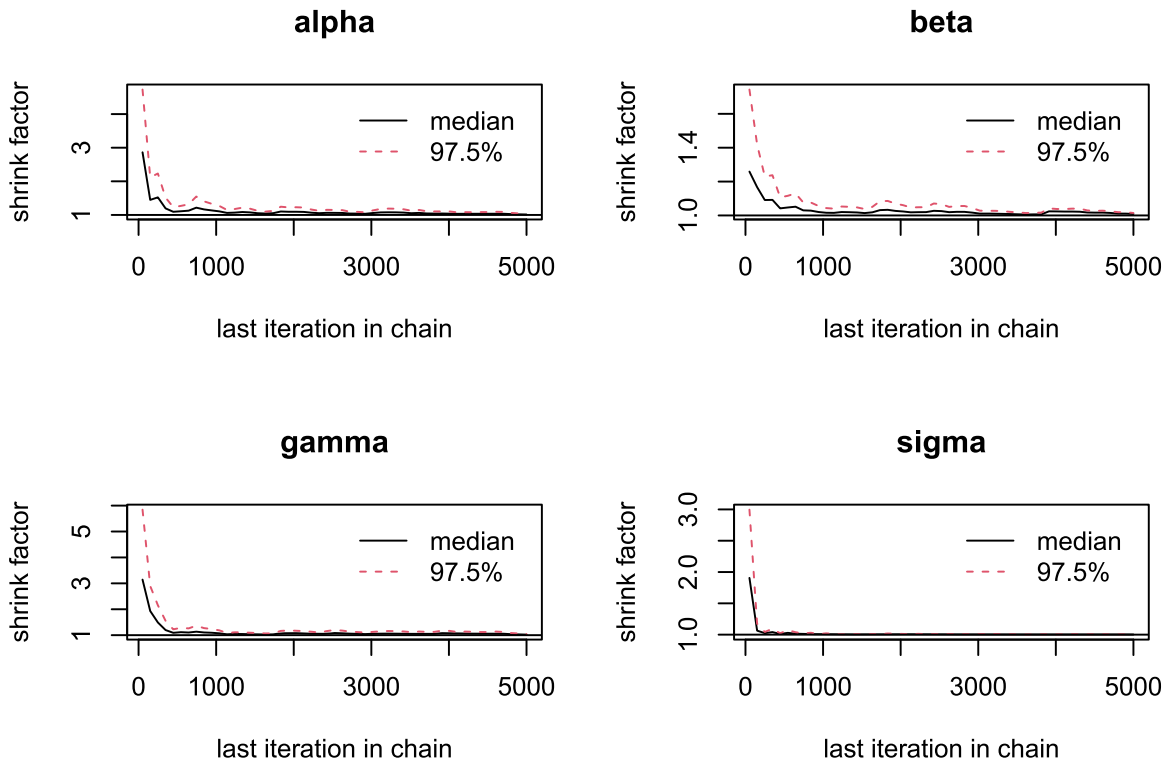


Figure 5: Gelman and Rubin plots for the convergence of parameter estimations.

Table 3: Summary statistics for the parameters of the posterior distribution.

				95% credible interval	
	mean	median	SD		
alpha	2.6334689	2.6301049	0.0650808	2.5098464	2.7623211
beta	0.5402406	0.5367236	0.0551801	0.4393449	0.6562372
gamma	-0.1913138	-0.1842175	0.0486798	-0.3060345	-0.1240828
sigma	0.1018716	0.0995674	0.0156298	0.0781796	0.1389337

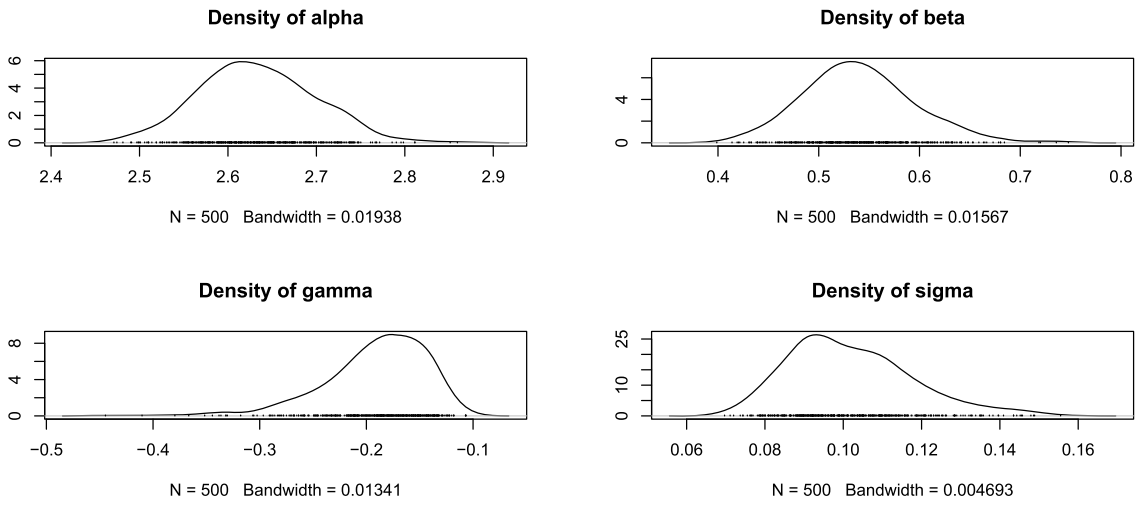


Figure 6: Density plot of the posterior parameters.

4.4 Are the parameters a posteriori correlated? Check by making pairwise scatterplots of the MCMC samples.

A visual exploration of correlation between the parameters of the posterior distribution can be found in figure 7. A clear posterior correlation between the estimates for α and γ was observed, while for all other pairwise correlation, no clear trend was obvious.

4.5 Check the sensitivity of the priors. Change the priors by increasing or decreasing the variance. Does this affect the results?

A sensitivity analysis of changes of the variance for the prior assumptions on the parameter estimates can be found in table 4. Results for different assumptions were comparable.

4.6 Make a plot of the posterior growth curve. Visualize also the uncertainty of the plotted curve. How does this compare to the observed data?

The estimated posterior growth curve is visualized in black, the uncertainty of the plotted curve is visualized in blue. Overall, the estimated posterior growth curve fits the given data well. For each of the 5000 interations, the corresponding growth curve was created based on the values corresponding to alpha, beta,

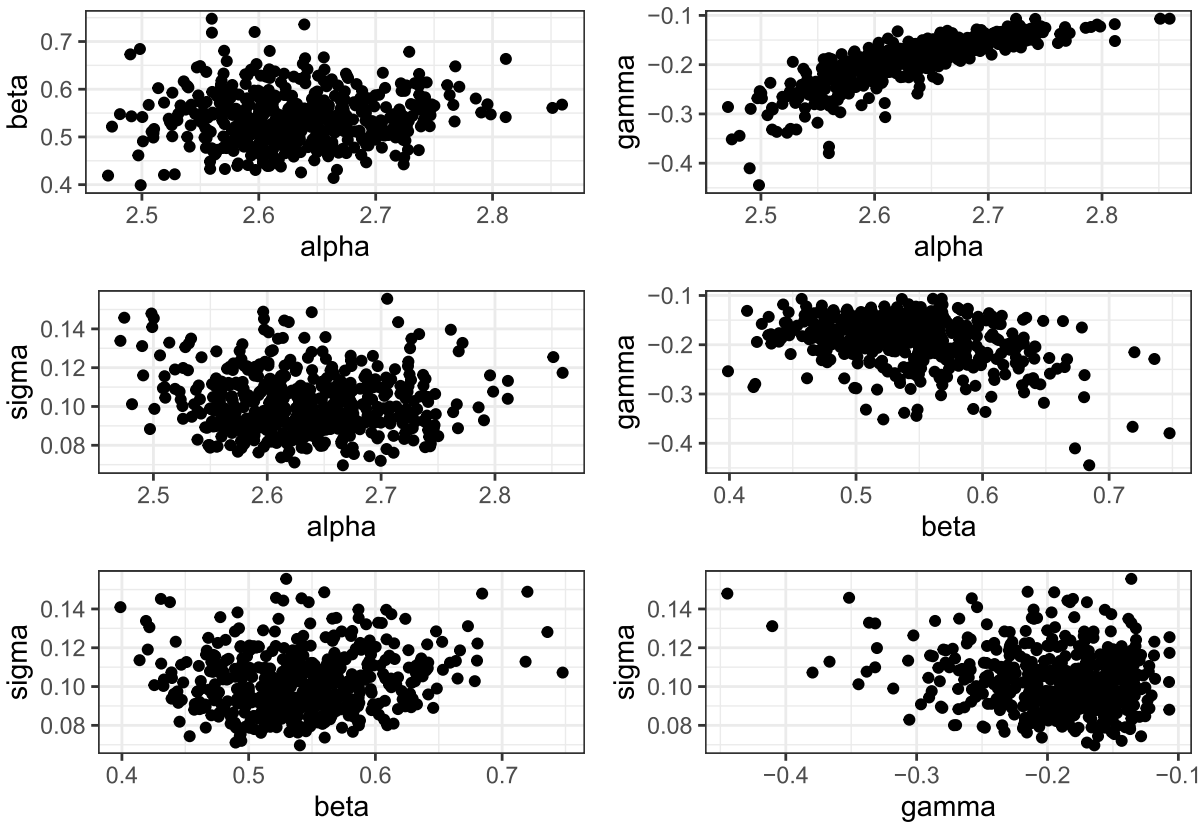


Figure 7: Pairwise correlations between the parameters of the posterior distribution.

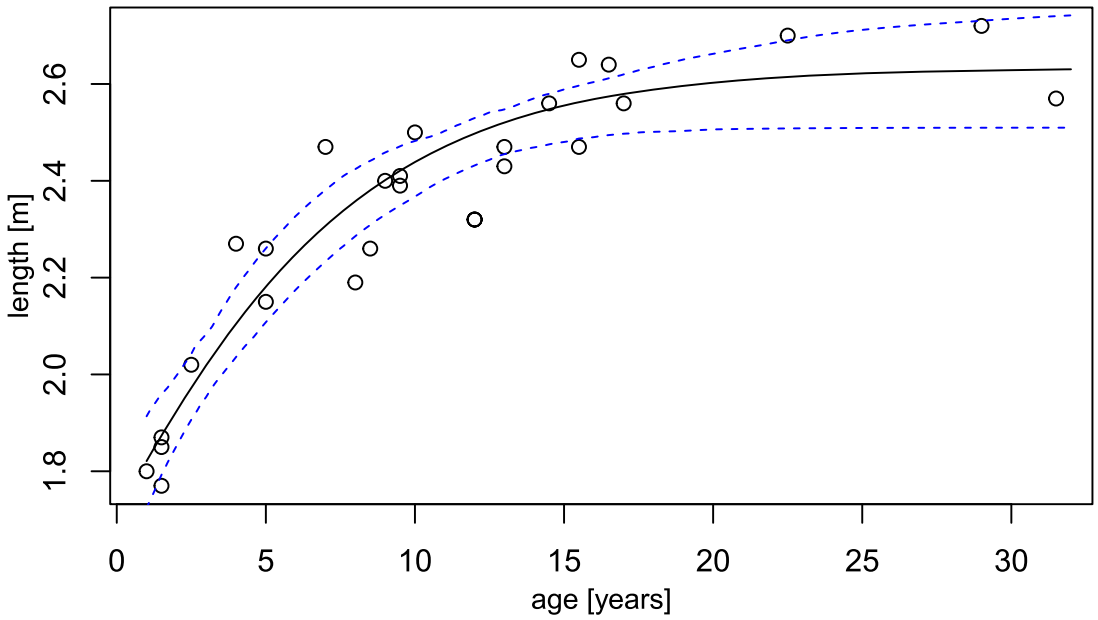
Table 4: Summary statistics for the parameters of the posterior distribution.

				95% credible interval	
	mean	median	SD		
Reduced variance					
alpha	-3.8042913	-3.6221359	0.7108961	-4.9665573	-2.7645656
beta	-2.8355000	-2.7326462	0.3524108	-3.4602882	-2.3255547
gamma	-0.0064552	-0.0064109	0.0011374	-0.0088560	-0.0042817
sigma	0.1798487	0.1761456	0.0260811	0.1406500	0.2421814
Original variance					
alpha	2.6334689	2.6301049	0.0650808	2.5098464	2.7623211
beta	0.5402406	0.5367236	0.0551801	0.4393449	0.6562372
gamma	-0.1913138	-0.1842175	0.0486798	-0.3060345	-0.1240828
sigma	0.1018716	0.0995674	0.0156298	0.0781796	0.1389337
Increased variance					
alpha	2.6417483	2.6372335	0.0586705	2.5389138	2.7584422
beta	0.5282686	0.5269210	0.0464055	0.4414705	0.6274570
gamma	-0.1791826	-0.1773175	0.0363095	-0.2547171	-0.1191812
sigma	0.0882138	0.0867776	0.0101765	0.0711172	0.1108807

Table 5: summary table of the posterior predictive distribrution for a new observation of an animal at age 25 years.

mean	sd	median	p0.025	p0.975
2.617902	0.1130482	2.617414	2.394872	2.835766

gamma and sigma. For different ages rangong from 1 to 31.5, the 2.5% and 97.5% percentiles of the estimated animal lengths were calculated.



4.7 Give the posterior predictive distribution of the length of a new animal at age 25.

A histogram and a summary table of the posterior predictive distribrution of the length of a new animal at age 25 are given in figure 8 and table 5, respectively. The expected length of a new animal at age 25 is 2.6179023.

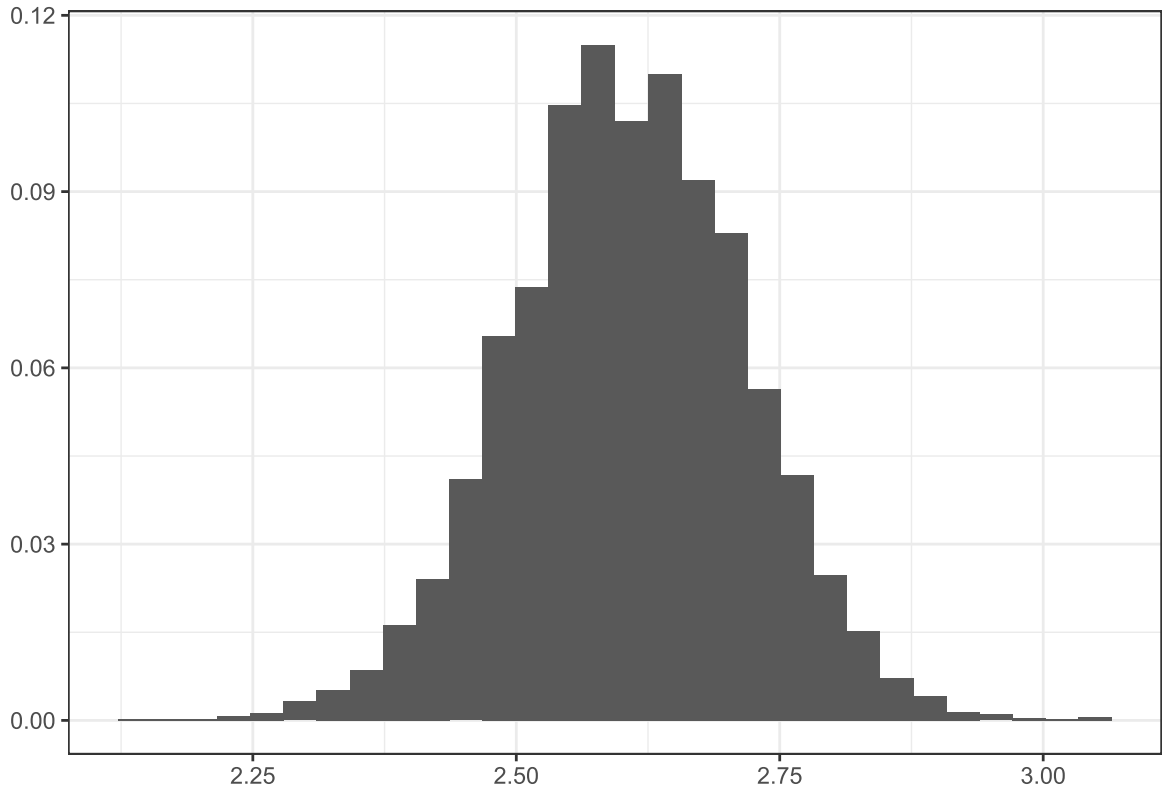


Figure 8: Histogram for the posterior predictive distribution for a new observation of an animal at age 25 years.

Exercise No. 5

5.1 Explain the model that is given in the code. What is the likelihood (write it out)? What prior is assumed for the parameters?

This model assumes that the number of people demonstrating HIV-seroconversion (r) within this study (n) follows a binomial distribution with parameters θ_i for people randomized to either the vaccine ($i = 1$) or placebo ($i = 2$), with a prior distribution of θ described by a β-distribution with parameters α=β=0.5 (a transformation invariant, non-informative prior, also called Jeffreys prior).
The likelihood function is described by:

$$lik(\theta_i|y) = \binom{n_i}{r_i} \theta_i^{r_i} \cdot (1-\theta_i)^{n_i - r_i}$$

With n_i and r_i being the total number of participants and number of participants with seroconversion for the treatmentgroups i as defined above, respectively.

5.2 Use 10,000 MCMC iterations with a burn-in of 1,000 MCMC iterations for estimation of the parameters. What is the MCMC error? Check convergence of the MCMC chain for all parameters. Discuss.

The MCMC errors as an estimate of inaccuracy of Monte Carlo sampling were 8.519761×10^{-6} and 1.0229613×10^{-5} for θ_1 and θ_2, respectively. In general, a large MCMC error can be decreased by increasing the number of iterations in the sampling procedure. Potential scale reduction factors (upper CI) were found to be 1.0003 (1.001) and 1 (1.0001) for θ_1 and θ_2, respectively, while the corresponding Gelman and Rubin plot being displayed in figure 9. Taken all information together, we observed good convergence of the parameters.

5.3 Give the posterior mean and median for the proportions. Give also 95 credible intervals.

Posterior mean and median together with a 95% credible interval can be found in table 6.

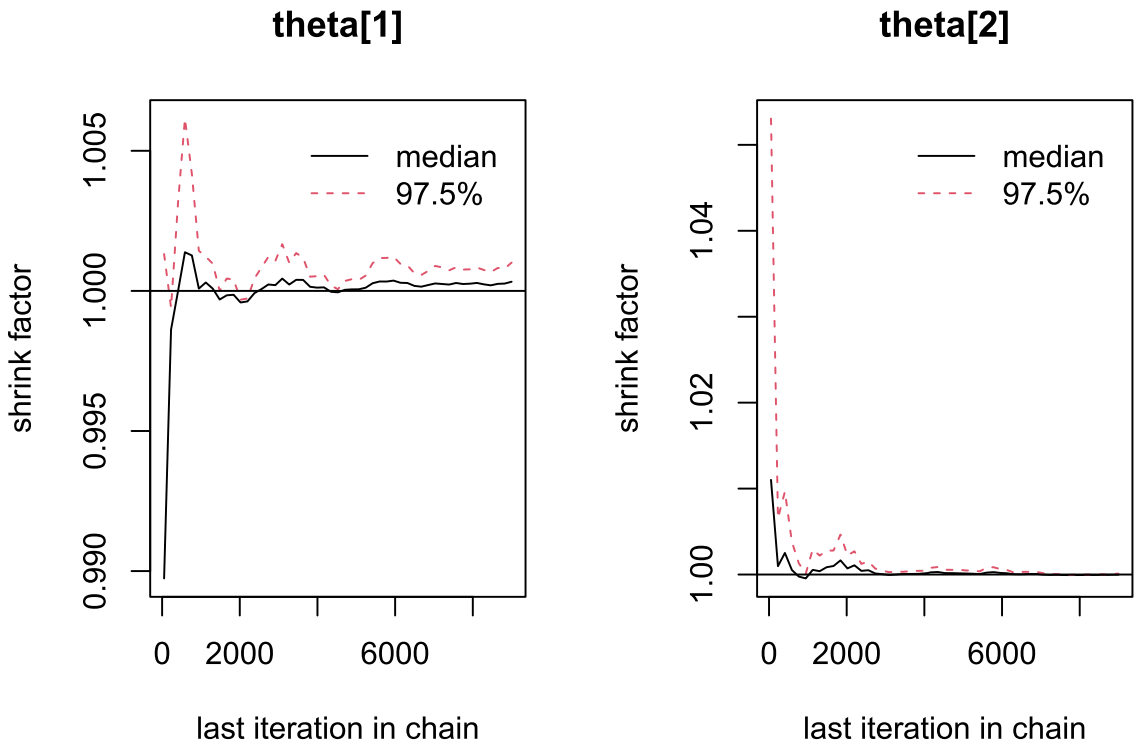


Figure 9: Gelman and Rubin plots for parameters θ_1 and θ_2.

Table 6: Means and medians with 95 credible intervals for the parameters of the posterior distribution

			95% credible interval	
	mean	median		
theta[1]	0.0062773	0.0062394	0.0046946	0.0081170
theta[2]	0.0090779	0.0090163	0.0071584	0.0113099

5.4 Interest is in the vaccine efficacy (VE), defined as one minus the relative hazard rate of HIV in vaccine versus placebo group:

$$VE = 1 - \frac{\theta_1}{\theta_2}$$

Obtain a 95% credible interval for VE. How can we interpret this interval?

	Mean	Median	St.Dev.	95%CI_low	95%CI_upp
Efficacy	0.2992135	0.3102101	0.1290563	0.0167689	0.5209288

Give a plot of the posterior density of VE.

The plot of the posterior density of VE is displayed in figure 10.

5.5 What is the posterior probability that the proportion θ_1 of infected amongst those vaccinated is actually smaller than the proportion θ_2 of infected among those that take the placebo?

The posterior probability is 0.02.

5.6 Give a plot of the posterior predictive distribution for a future 100 subjects that are vaccinated, as well as the posterior predictive distribution for a future 100 subjects that take the placebo. Give also a 95% posterior predictive set.

Plot for the posterior density of VE can be found in figure 11.

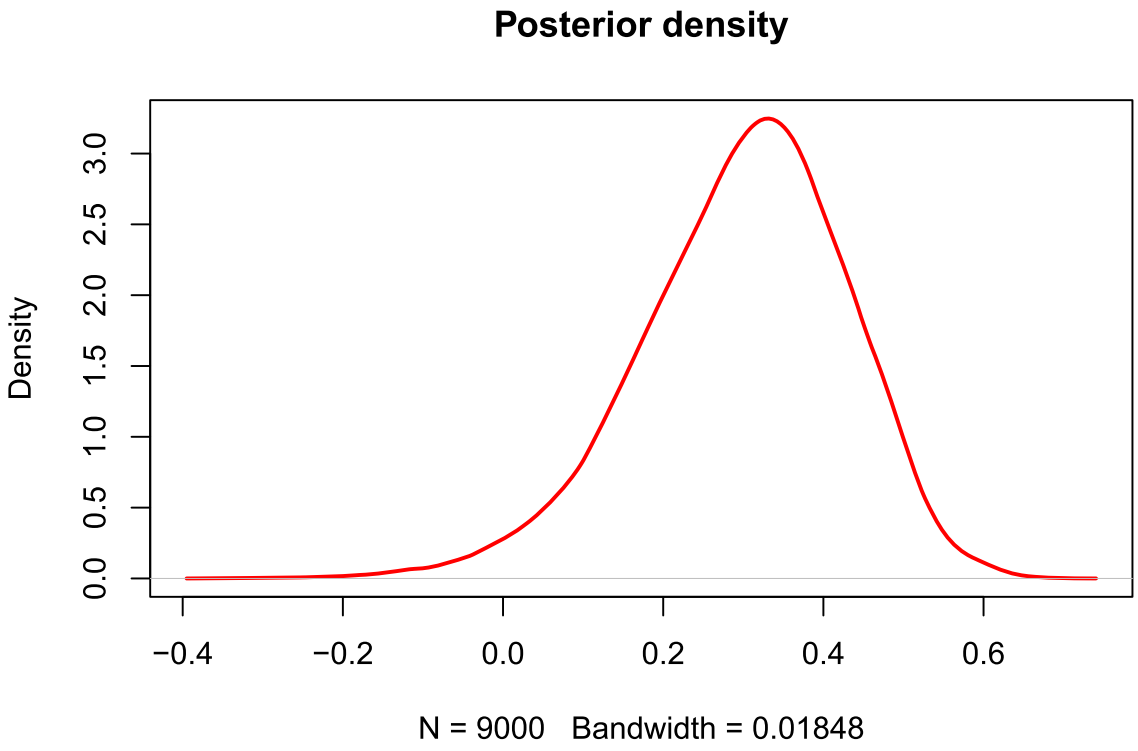


Figure 10: Plot of the posterior density of the vaccine efficacy (VE).

To constract the posterior predictive distribution (PPD), we have to use the posterior distribution as prior. As the posterior distribution is beta distribution, the data are binomial, we can calculate the shape parameters of the PPD as following:

$$\alpha_{post} = \alpha_0 + y$$

$$\beta_{post} = \beta_0 + n - y$$

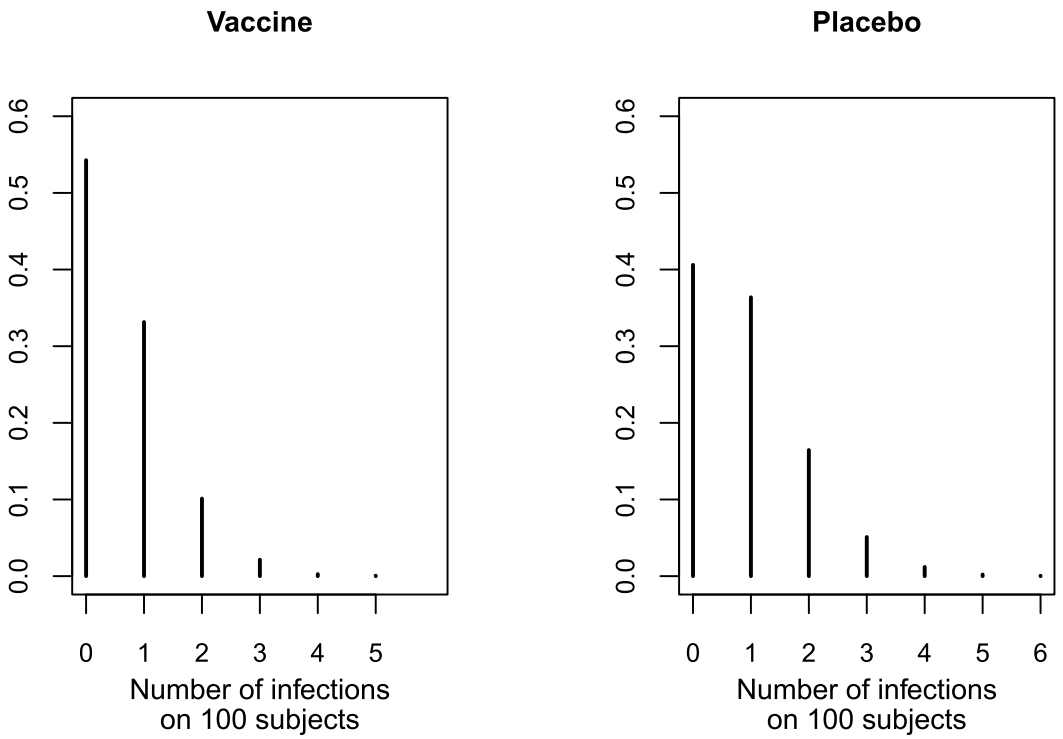


Figure 11: Posterior predictive distribution for future observations in the vaccine and placebo groups.

Table 7: 95 % predictive set for θ_1 and θ_2.

	95% credible interval	
	lower	upper
θ_1	0.0046886	0.0081033
θ_2	0.0071484	0.0112512

5.7 Do a sensitivity analysis of the prior. How do results change in case you use the following priors:

Use a non-informative prior for both θ_1 and θ_2.

Results of the posterior summaries for the estimates for both parameters using the original prior assumptions as well as the assumption of weakly informative (flat) priors for the distribution or θ_1 and θ_2 can be found in table 8. Of notice, the effects of the changes in the prior are negligible.

Table 8: Results for sensitivity analysis on the priors, assuming no prior knowledge or deciding againgst including prior knowledge

	Mean	Median	St.Dev.	95%CI_low	95%CI_upp
Original prior					
theta[1]	0.0062773	0.0062394	0.0008748	0.0046946	0.0081170
theta[2]	0.0090779	0.0090163	0.0010637	0.0071584	0.0113099
Assumption of non-informative priors for theta					
theta[1]	0.0063321	0.0062899	0.0008661	0.0047627	0.0081190
theta[2]	0.0091718	0.0091476	0.0010591	0.0072090	0.0113912

Use a prior that reflects a previous study (r_1=56, n_1=8,202 for people who have taken the vaccine; r_2=76, n_2=8,200 for people who have taken placebo).

The Beta distribution is defined by parameters alpha and beta, with alpha = x + 1 and beta = n - x + 1, where x is the number of positive events out of n trials. Therefore, in our example, the prior distribution has parameters alpha_0_v=57, beta_0_v=8147; alpha_0_p=77, beta_0_p=8125.

Results of the posterior summaries for the estimates for both parameters using the original prior as well as the assumption based on the result of a previous study for the distribution or θ_1 and θ_2 can be found in table 9. Of notice, the effects of the changes in the prior are negligible on the estimates, however, with the prior assumptions supporting the data, the standard error of both parameters decreased.

Table 9: Results for sensitivity analysis on the priors, assuming no prior knowledge or deciding againgst including prior knowledge

	Mean	Median	St.Dev.	95%CI_low	95%CI_upp
Original prior					
theta[1]	0.0062773	0.0062394	0.0008748	0.0046946	0.0081170
theta[2]	0.0090779	0.0090163	0.0010637	0.0071584	0.0113099
Prior based on previous study results					
theta[1]	0.0065761	0.0065516	0.0006322	0.0053924	0.0078714
theta[2]	0.0091987	0.0091795	0.0007500	0.0077934	0.0107056

Appendix: All code for this report

```
knitr::opts_chunk$set(include=TRUE, echo = FALSE, warning = FALSE, message = FALSE, error = FALSE)


library(coda)


library(DescTools)

library(dplyr)


library(extraDistr)


library(ggplot2)

library(gridExtra)


library(kableExtra)


library(LaplacesDemon)


library(nimble)


library(rootSolve)
```

```
pairwise.correlation = function(Arg1, Arg2){
  result = cor.test(Arg1, Arg2, method="spearman")
  est = result[[4]]
  if(est < 0.01){est = "<0.01"} else {est = paste0("=", round(est,2))}
  p = result$p.value
  if(p < 0.001){p = "<0.001"} else {p = paste0("=", round(p,3))}
  paste0("ρ",est," (p", p, ")")
}

#set.seed(1)

x = c( 1.0, 1.5, 1.5, 1.5, 2.5, 4.0, 5.0, 5.0,
       7.0, 8.0, 8.5, 9.0, 9.5, 9.5, 10.0, 12.0, 12.0, 13.0,
       13.0, 14.5, 15.5, 15.5, 16.5, 17.0, 22.5, 29.0, 31.5)
N = 27

Y = c(1.80, 1.85, 1.87, 1.77, 2.02, 2.27, 2.15, 2.26, 2.47,
      2.19, 2.26, 2.40, 2.39, 2.41, 2.50, 2.32, 2.32, 2.43, 2.47,
      2.56, 2.65, 2.47, 2.64, 2.56, 2.70, 2.72, 2.57)

model4.data <- list('x' = x, 'Y'=Y)
model4.constant <- list('N' = N)
model4.inits <- list(sigma=1, alpha=0, beta=0, gamma=0 )

Model4.code <- nimbleCode(
  {
    for (i in 1:N) {
      Y[i] ~  dnorm( alpha / (1 + beta*exp(gamma *x[i])),tau)
    }
    alpha ~ dunif(-10,10)
    beta ~ dunif(-10,10)
    gamma ~ dunif(-10,10)
```

```
    sigma ~ T(dnorm(0, 100), 0, 1000)
    tau <- 1/(sigma^2)
  })

Model4 <- nimbleModel(Model4.code, constants=model4.constant, data=model4.data, inits=model4.inits)
Model4.compiled <- compileNimble(Model4)

set.seed(12)
Model4.init.output = nimbleMCMC(Model4.compiled, niter=5000, nburnin=1000, thin=10, summary=T)

set.seed(12)
Model4.output = nimbleMCMC(Model4.compiled, niter=9000, nburnin=4000, thin=10, summary=T)



set.seed(12)

Model4.init.GuR.output = nimbleMCMC(Model4, niter=5000, nburnin=1000, thin=1, nchains=5)

Model4.init.GuR.List = mcmc.list(list(
  as.mcmc(Model4.init.GuR.output$chain1),
  as.mcmc(Model4.init.GuR.output$chain2),
  as.mcmc(Model4.init.GuR.output$chain3),
  as.mcmc(Model4.init.GuR.output$chain4),
  as.mcmc(Model4.init.GuR.output$chain5)
))

set.seed(12)

Model4.GuR.output = nimbleMCMC(Model4, niter=9000, nburnin=4000, thin=1, nchains=5)



Model4.GuR.List = mcmc.list(list(
```

```
  as.mcmc(Model4.GuR.output$chain1),
  as.mcmc(Model4.GuR.output$chain2),
  as.mcmc(Model4.GuR.output$chain3),
  as.mcmc(Model4.GuR.output$chain4),
  as.mcmc(Model4.GuR.output$chain5)
))


par(mfrow=c(2,2))

traceplot(as.mcmc(Model4.init.output$samples))


acf(as.mcmc(Model4.init.output$samples))

gelman.diag(Model4.init.GuR.List)[1] %>%
  kable(linesep="", booktabs = TRUE, align = "cc", digits = 3,
        caption="Gelman and Rubin convergence diagnostics for each parameter in the model using 5,000 i
  kable_styling(bootstrap_options = c("condensed"), latex_options = "hold_position") %>%
  column_spec(1, bold=T)

par(mfrow=c(2,2))

traceplot(as.mcmc(Model4.output$samples))


acf(as.mcmc(Model4.output$samples))

gelman.diag(Model4.GuR.List)[1] %>%
  kable(linesep="", booktabs = TRUE, align = "cc", digits = 3,
        caption="Gelman and Rubin convergence diagnostics for each parameter in the model.") %>%
  kable_styling(bootstrap_options = c("condensed"), latex_options = "hold_position") %>%
```

```
  column_spec(1, bold=T)


gelman.plot(Model4.GuR.List)



Model4.output$summary %>%
  "colnames<-"(c("mean", "median", "SD", " ", " ")) %>%
  kable(align="ccccc", linesep ="", booktabs=T, caption = "Summary statistics for the parameters of the
  kable_styling(bootstrap_options = c("condensed")) %>%
  column_spec(1, bold=T) %>%
  add_header_above(c(" "=4, "95% credible interval"=2))



par(mfrow=c(2,2))
densplot(as.mcmc(Model4.output$samples))



mcmcerror1 = MCSE(Model4.output$samples[,1], method="batch.means", batch.size="sqrt")$se
mcmcerror2 = MCSE(Model4.output$samples[,2], method="batch.means", batch.size="sqrt")$se
mcmcerror3 = MCSE(Model4.output$samples[,3], method="batch.means", batch.size="sqrt")$se
mcmcerror4 = MCSE(Model4.output$samples[,4], method="batch.means", batch.size="sqrt")$se



par(mfrow=c(3,2))

p1 = ggplot(as.data.frame(Model4.output$samples)) +
  geom_point(aes(x=alpha, y=beta)) +
  labs(x="alpha", y="beta") + theme_bw()
p2 = ggplot(as.data.frame(Model4.output$samples)) +
  geom_point(aes(x=alpha, y=gamma))+
  labs(x="alpha", y="gamma") + theme_bw()
p3 = ggplot(as.data.frame(Model4.output$samples)) +
```

```
  geom_point(aes(x=alpha, y=sigma))+
  labs(x="alpha", y="sigma") + theme_bw()
p4 = ggplot(as.data.frame(Model4.output$samples)) +
  geom_point(aes(x=beta, y=gamma))+
  labs(x="beta", y="gamma") + theme_bw()
p5 = ggplot(as.data.frame(Model4.output$samples)) +
  geom_point(aes(x=beta, y=sigma))+
  labs(x="beta", y="sigma") + theme_bw()
p6 = ggplot(as.data.frame(Model4.output$samples)) +
  geom_point(aes(x=gamma, y=sigma))+
  labs(x="gamma", y="sigma") + theme_bw()

grid.arrange(p1,p2,p3,p4,p5,p6, nrow=3, ncol=2)

#set.seed(1)

Model4.lo.code <- nimbleCode(
  {
    for (i in 1:N) {
      Y[i] ~  dnorm( alpha / (1 + beta*exp(gamma *x[i])),tau)
    }
    alpha ~ dunif(-5,5)
    beta ~ dunif(-5,5)
    gamma ~ dunif(-5,5)
    sigma ~ T(dnorm(0, 10), 0, 1000)
    tau <- 1/(sigma^2)
  })

Model4.lo = nimbleModel(Model4.lo.code, constants=model4.constant, data=model4.data, inits=model4.inits
Model4.lo.output = nimbleMCMC(Model4.lo, niter=9000, nburnin=4000, thin=10, summary=T)

#set.seed(1)
```

```
Model4.hi.code <- nimbleCode(
  {
    for (i in 1:N) {
      Y[i] ~  dnorm( alpha / (1 + beta*exp(gamma *x[i])),tau)
    }
    alpha ~ dunif(-100,100)
    beta ~ dunif(-100,100)
    gamma ~ dunif(-100,100)
    sigma ~ T(dnorm(0, 1000), 0, 1000)
    tau <- 1/(sigma^2)
  })

Model4.hi = nimbleModel(Model4.hi.code, constants=model4.constant, data=model4.data, inits=model4.inits)
Model4.hi.output = nimbleMCMC(Model4.hi, niter=9000, nburnin=4000, thin=10, summary=T)



rbind(Model4.lo.output$summary,
      Model4.output$summary,
      Model4.hi.output$summary) %>%
  "colnames<-"(c("mean", "median", "SD", " ", " ")) %>%
  kable(align="ccccc", linesep ="", booktabs=T, caption = "Summary statistics for the parameters of the
  kable_styling(bootstrap_options = c("condensed")) %>%
  pack_rows("Reduced variance", 1, 4) %>%
  pack_rows("Original variance", 5, 8) %>%
  pack_rows("Increased variance", 9, 12) %>%
  column_spec(1, bold=T) %>%
  add_header_above(c(" "=4, "95% credible interval"=2))



samples <- as.data.frame(Model4.output$samples)


nSamp <- nrow(samples)
```

```
x2 <- seq(round(min(x),0.01), round(max(x),0.01), by=0.1)
N2 <- length(x2)
predictedSamples <- matrix(0, nSamp, N2)

set.seed(1)
for(i in 1:nSamp){
predictedSamples [i,] <- samples[i,"alpha"]/(1+ samples[i,"beta"]*exp(samples[i,"gamma"] *x2))
}

Pred <- data.frame(t(apply(predictedSamples, 2, quantile, probs=c(0.025, 0.975))))
colnames(Pred) <- c("Quantile2.5_mean", "Quantile97.5_mean")


#code to check observed vs predicted + prediction interval
pred <- Model4.output$summary[1,1] / (1 + Model4.output$summary[2,1] *exp(Model4.output$summary[3,1]*x2

plot(x, Y, xlab="", ylab="")
lines(x2, pred)
lines(x2, Pred$Quantile2.5_mean, col="blue", lty=2)
lines(x2, Pred$Quantile97.5_mean, col="blue", lty=2)
title(xlab="age [years]", ylab="length [m]", line=2, cex.lab=0.9)


set.seed(12)

x = c( 1.0, 1.5, 1.5, 1.5, 2.5, 4.0, 5.0, 5.0,
       7.0, 8.0, 8.5, 9.0, 9.5, 9.5, 10.0, 12.0, 12.0, 13.0,
       13.0, 14.5, 15.5, 15.5, 16.5, 17.0, 22.5,  29.0, 31.5, 25)
N = length(x)

Y = c(1.80, 1.85, 1.87, 1.77, 2.02, 2.27, 2.15, 2.26, 2.47,
      2.19, 2.26, 2.40, 2.39, 2.41, 2.50, 2.32, 2.32, 2.43, 2.47,
```

```
      2.56, 2.65, 2.47, 2.64, 2.56, 2.70, 2.72, 2.57, NA)

model4.data <- list('x' = x, 'Y'=Y)
model4.constant <- list('N' = N)

Model4.code <- nimbleCode(
  {
    for (i in 1:N) {
      Y[i] ~  dnorm( alpha / (1 + beta*exp(gamma *x[i])),tau)
         Y25 <-  dnorm( alpha / (1 + beta*exp(gamma *25)),tau)
    }
    alpha ~ dunif(-10,10)
    beta ~ dunif(-10,10)
    gamma ~ dunif(-10,10)
    tau <- 1/(sigma^2)
    sigma ~ T(dnorm(0, 100), 0, 1000)
  })

Model4 <- nimbleModel(Model4.code, constants=model4.constant, data=model4.data, inits=model4.inits)
Model4.compiled <- compileNimble(Model4)

set.seed(12)

dataNodes <- Model4$getNodeNames(dataOnly = TRUE)
parentNodes <-Model4$getParents(dataNodes, stochOnly = TRUE)
simNodes <- Model4$getDependencies(parentNodes, self = FALSE)

mcmc    <- buildMCMC(Model4, monitors = parentNodes)
cmcmc   <- compileNimble(mcmc, project = Model4.compiled, resetFunctions = TRUE)
samples <- runMCMC(cmcmc , niter = 9000, nburnin = 4000)

nSamp <- nrow(samples)
```

```
ppSamples <- matrix(0, nSamp, N)


set.seed(1)
for(i in 1:nSamp){
 Model4.compiled[["gamma"]] <- samples[i, "gamma"]
  Model4.compiled[["sigma"]] <- samples[i, "sigma"]
  Model4.compiled[["alpha"]] <- samples[i, "alpha"]
 Model4.compiled[["beta"]] <- samples[i, "beta"]
  Model4.compiled$simulate(simNodes, includeData = TRUE)
  ppSamples[i, ] <- Model4.compiled[["Y"]]
}


ppd25 <- data.frame(
  mean=mean(ppSamples[ , x==25]),
  sd=sd(ppSamples[ , x==25]),
  median=median(ppSamples[ , x==25]),
  p0.025 = quantile(ppSamples[ , x==25], 0.025),
  p0.975 = quantile(ppSamples[ , x==25], 0.975))




ggplot() +
  geom_histogram(aes(x=ppSamples[,25], y = after_stat(count / sum(count)))) +
  labs(x = "", y="") +
  theme_bw()



kable(ppd25, row.names=F, booktabs=T, caption="summary table of the posterior predictive distribrution



n = c(8197, 8198)
```

```
N = sum(n)
r = c(51, 74)
R = sum(r)

theta = c(r[1]/n[1], r[2]/n[2])

Model5.constants = list("n" = n)
Model5.data = list("r" = r)
Model5.inits = list("theta" = theta, "alpha" = alpha, "beta" = beta)

Model5.Code = nimbleCode({
  for (i in 1:2)
    {
      r[i] ~ dbin(theta[i],n[i])
      theta[i] ~ dbeta(0.5, 0.5)
  }
})


Model5 = nimbleModel(Model5.Code, constants=Model5.constants, data=Model5.data, inits=Model5.inits)

Model5.compiled = compileNimble(Model5)

Model5.out = nimbleMCMC(Model5, niter=10000, nburnin=1000, thin=1, summary=T)


mcmcerror1 = MCSE(Model5.out$samples[,1], method="batch.means", batch.size="sqrt")$se
mcmcerror2 = MCSE(Model5.out$samples[,2], method="batch.means", batch.size="sqrt")$se


Model5.GuR.output = nimbleMCMC(Model5, niter=10000, nburnin=1000, thin=1, nchains=5)
```

```
Model5.GuR.List = mcmc.list(list(
  as.mcmc(Model5.GuR.output$chain1),
  as.mcmc(Model5.GuR.output$chain2),
  as.mcmc(Model5.GuR.output$chain3),
  as.mcmc(Model5.GuR.output$chain4),
  as.mcmc(Model5.GuR.output$chain5)
))

Model5.GuR.Diagnostics = gelman.diag(Model5.GuR.List)

  gelman.plot(Model5.GuR.List)

Summary = as.matrix(Model5.out$summary[,c(1,2,4,5)]) %>%
  "colnames<-"(c("mean", "median", " ", " "))

kable(Summary, align="cccc", linesep ="", booktabs=T, caption="Means and medians with 95 credible interv
  kable_styling(bootstrap_options = c("condensed"), latex_options = c("hold_position")) %>%
  column_spec(1, bold=T) %>%
  add_header_above(c(" "=3, "95% credible interval"=2))

hivCode2 <- nimbleCode({
  for (i in 1:2){
      r[i] ~ dbin(theta[i], n[i])
      theta[i] ~ dbeta(0.5, 0.5)
      Efficacy <- 1 - theta[1]/theta[2]
  }
})

hivData <- list(r = c(51, 74), n = c(8197, 8198))
hivInits <- list(theta = 1)

hiv2 <- nimbleModel(code = hivCode2, name = "hiv", data = hivData, inits = hivInits)
```

```
Chiv2 <- compileNimble(hiv2)
Chiv2$theta


set.seed(123)
efficacy.out <- nimbleMCMC(code = hivCode2, inits = hivInits, data = hivData,
                           nburnin = 1000, niter = 10000, summary = TRUE, WAIC = TRUE, monitors = c('Effica
dx <- density(efficacy.out[["samples"]])



efficacy.out$summary %>%
  kable(booktabs=T)



plot(dx, lwd = 2, col = "red", main = "Posterior density")
Efficacy <- efficacy.out[["samples"]]
neg.efficacy = Efficacy %>% subset(Efficacy < 0)

set.seed(123)
alpha_post1 <- 0.5 + 51
beta_post1 <- 0.5 + 8197 - 51

alpha_post2 <- 0.5 + 74
beta_post2 <- 0.5 + 8198 - 74



ppd_vaccine <- rbbinom(n=10000, 100, alpha_post1, beta_post1)
ppd_placebo <- rbbinom(n=10000, 100, alpha_post2, beta_post2)
ppd <- as.data.frame(cbind(ppd_vaccine, ppd_placebo))



par(mfrow=c(1,2))
```

```
plot(prop.table(table(ppd_vaccine)), ylab ="", xlab = "", main = "Vaccine", xlim=c(0, 6), ylim=c(0, 0.6)
title(xlab="Number of infections", line=2, cex.lab=0.9)
title(xlab="on 100 subjects", line=2.8, cex.lab=0.9)

plot(prop.table(table(ppd_placebo)),main = "Placebo", ylab ="", xlab="", xlim=c(0, 6), ylim=c(0, 0.6),
title(xlab="Number of infections", line=2, cex.lab=0.9)
title(xlab="on 100 subjects", line=2.8, cex.lab=0.9)



as.matrix(rbind(
  qbeta(c(0.025, 0.975), shape1 = alpha_post1, shape2 = beta_post1),
  qbeta(c(0.025, 0.975), shape1 = alpha_post2, shape2 = beta_post2))) %>%
  "rownames<-"(c("θ_1", "θ_2")) %>%
   kable(booktabs=T, caption = "95 % predictive set for θ_1 and θ_2.") %>%
  kable_styling(bootstrap_options = c("condensed")) %>%
  add_header_above(c("", "lower" = 1, "upper")) %>%
  add_header_above(c("", "95% credible interval" = 2)) %>%
  column_spec(1, bold=T)



Model.Sens1.Code = nimbleCode({
  for (i in 1:2)
    {
      r[i] ~ dbin(theta[i],n[i])
      theta[i] ~ dbeta(1,1)
  }
})

Model.Sens1 = nimbleModel(Model.Sens1.Code, constants=Model5.constants, data=Model5.data, inits=Model5.

Model.Sens1.compiled = compileNimble(Model.Sens1)
```

```
Model.Sens1.out = nimbleMCMC(Model.Sens1, niter=10000, nburnin=1000, thin=1, summary=T)



rbind(
  Model5.out$summary,
  Model.Sens1.out$summary) %>%
  kable(align="cccc", linesep="", booktabs=T,
        caption="Results for sensitivity analysis on the priors, assuming no prior knowledge or decidin
  kable_styling(latex_options = c("hold_position")) %>%
  pack_rows("Original prior", 1, 2) %>%
  pack_rows("Assumption of non-informative priors for theta", 3, 4)


set.seed(123)


hivCodeSens2 <- nimbleCode({
  for (i in 1:2){
      r[i] ~ dbin(theta[i], n[i])
  theta[1] ~ dbeta(57, 8147)
  theta[2] ~ dbeta(77, 8125)
  }
})
hivData <- list(r = c(51, 74), n = c(8197, 8198))
hivInits <- list(theta = 2)
hivSens2 <- nimbleModel(code = hivCodeSens2, name = "hivSens2", data = hivData, inits = hivInits)
ChivSens2 <- compileNimble(hivSens2)


mcmc.outSens2 <- nimbleMCMC(code = hivCodeSens2,
                        inits = hivInits,
                        data = hivData,
                        nburnin = 1000, niter = 10000,
                        summary = TRUE, WAIC = TRUE,
                        monitors = c('theta'))
```

```
rbind(
  Model5.out$summary,
  mcmc.outSens2$summary) %>%
  kable(align="cccc", linesep="", booktabs=T,
        caption="Results for sensitivity analysis on the priors, assuming no prior knowledge or decidin
  kable_styling(latex_options = c("hold_position")) %>%
  pack_rows("Original prior", 1, 2) %>%
  pack_rows("Prior based on previous study results", 3, 4)
```