Final Project

Hwanho Kim

2025-04-26

1 Abstract

This project predicts the probability that each stock will have a positive return in the future by using the hierarchical Bayesian model. In the data, each stock will have 1 if that stock had a positive return in that period, 0 otherwise. I treat the outcome of the observed stocks as Bernoulli outcomes, and the probability of the stock is affected by the industry to which the stock belongs. Each stock's probability of a positive return is assumed to arise from a sector-level Beta distribution with the parameters reflecting market uncertainty. The Exploratory Data Analysis process will show the differences not only among stocks but also among sectors, and we can summarize our data numerically and graphically. I fit the model using Markov Chain Monte Carlo (MCMC) methods with JAGS and diagnose convergence. Posterior analysis allows us to estimate the probability that each sector offers the best investment opportunities, and to distinguish the most promising stock within each sector.

2 Data

2.1 Input Data Description

The dataset shows us the return record of 50 stocks in different periods, and each stock belongs to one of 5 sectors. Individual stocks are observed over 30 periods, and they have 1 if that stock in a specific sector had a positive return, 0 otherwise. The dataset has 3 columns: sectors, stock,

and flip, and each row represents whether a stock within a specific sector had a positive return in that period or not. The dataset has 5 sectors, and each sector contains 10 stocks. Thus, there are 1,5000 observations total in the dataset (50 stocks x 30 periods each).

We have only 30 periods for each stock, which is a relatively small number of stocks to clarify the uncertainty in parameter estimates. Also, we assume that returns within each stock are independently and identically distributed. These limitations of the dataset enable our model to process easily and plausibly.

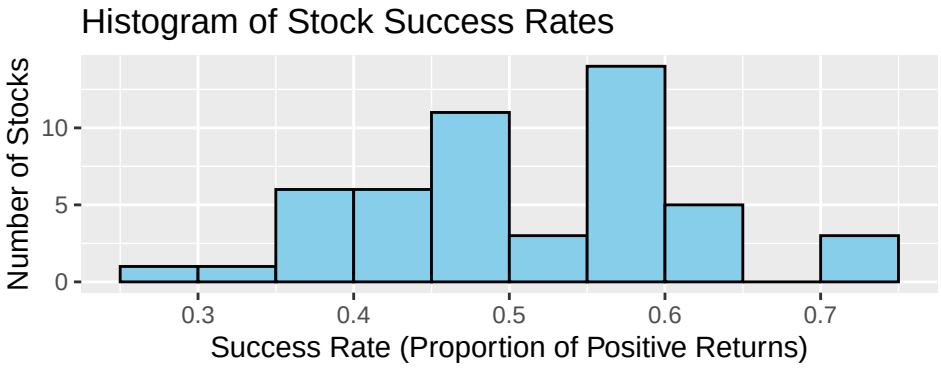
2.2 Explanatory Data Analysis

In this project, the goal of EDA was to summarize the dataset in terms of market, sector (industry), and stock. Overall, the dataset consists of 1,500 observations, each binary outcome representing whether or not a specific stock in a certain sector had a positive return during the given period.

Table 1: Table 1: Summary of Flip Outcomes

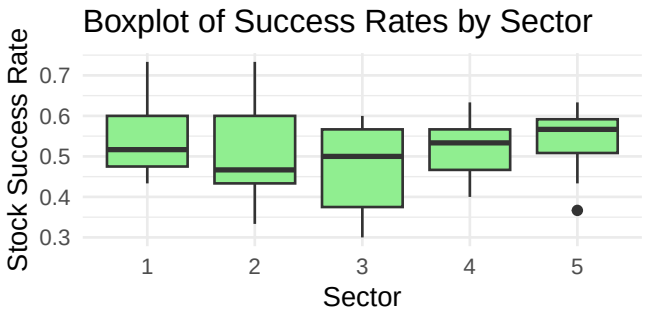
Flip	Count	Proportion
0	721	0.4807
1	779	0.5193

In a market aspect, I count the number of 0 and 1, and compare them on a proportion scale. In this market, 721 stocks had negative returns, and 779 stocks had positive returns. In other words, 48.067 percent of stocks had negative returns, and 51.933 percent of stocks had positive returns.



In the stock aspect, individual stocks show various success rates, ranging approximately from 0.33 to 0.73. The wide probability range demonstrates that each stock have a different success rate, and the probability of a positive return for each stock is affected by other factors. In the histogram, the success rate is more focused on the interval [0.4, 0.5] and [0.5, 0.6]. We can conclude that each stock has a different probability of positive return, and the distribution is focused on a near 0.5 success rate interval.

	sector	sector_success_rate
1	1	0.5567
2	2	0.5033
3	3	0.4767
4	4	0.52
5	5	0.54



In the sector aspect, they have quite different rates of success, even thoughthe rates are close with 0.5. The table and box plot have different means because I try to group the data by sector, and calculate the mean for the table while I use summary of data which is grouped by sector and plotted as box plot. However, these two clearly show that different industry affect to the probability of stocks within that.

3 Model

3.1 Model Selection

I use a hierarchical Bayesian model to predict the probability that a stock produces a positive return. The model has three levels: individual observations, sectors, and the overall market.

3.1.1 Observation Level

$y_{ijt} \sim Bernouli(\theta_{ij})$ where $i = 1, 2, 3, ...10$, $j = 1, 2, 3, 4, 5$, $t = 1, 2, 3, ...30$

For each stock i in sector j in period t, y_{ijt} will have 1 if it has a positive return.

y_{ijt} model each flip given the stock's positive return probability.

3.1.2 Stock Level

$\theta_{ij} \sim Beta(\alpha_j, \beta_j)$

This variable models each stock's success probability inside each sector.

θ_{ij} is the probability of positive return of stock i in sector j.

α_j and β_j are parameters controlling the mean and variance of success rates within sector j.

3.1.3 Sector Level

$\alpha_j \sim Gamma(2, 0.1)$, $\beta_j \sim Gamma(2, 0.1)$

The parameters α_j and β_j have weakly informative priors.

This model uncertainty across sectors.

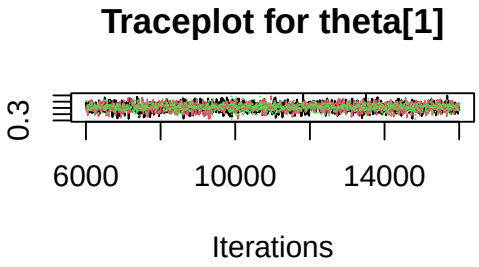
3.2 Justification

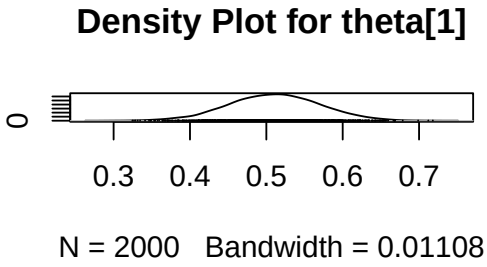
This hierarchical structure considers not only the variability of stocks within the sector but also the variability between sectors, and parameters are affected by the previous level's parameters. Therefore, a hierarchical Bayesian model is appropriate for modeling the flip.

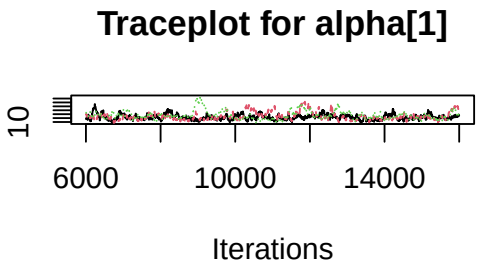
3.3 Model Building

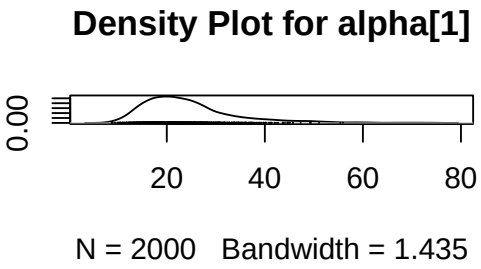
I implement the hierarchical Bayesian model using the rjags package in R to perform Markov Chain Monte Carlo (MCMC) sampling. For the algorithm setting, I use 3 chains, and 10,000 iterations occur per chain, total of 30,000 iterations. The model will use the first 5,000 iterations as a Burn-in period. The model will use random starting values for the parameters. For example, $\theta_{i,j}$ will be initialized by a random value from a $Beta(1, 1)$ distribution.

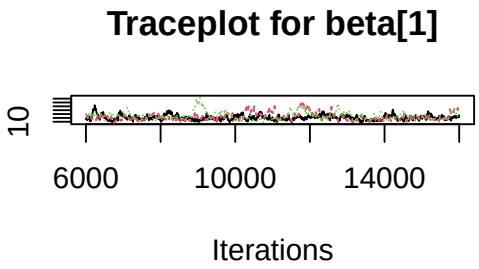
3.4 Convergence Diagnostics

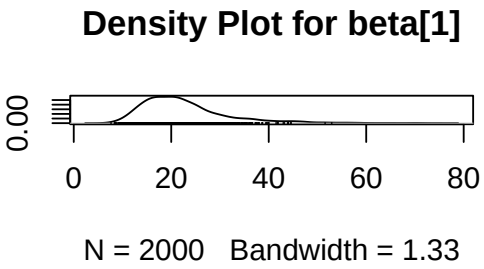


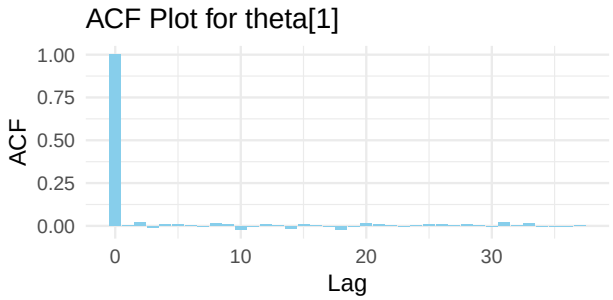












	Parameter	ESS
1	theta[1]	5799
2	alpha[1]	131
3	beta[1]	148

To assess convergence of the MCMC algorithm, we can use the traceplots and effective sample sizes for all monitored parameters. Originally, there should be 5 alpha and beta results, and 50 theta results, but I extracted one example for each parameter now. Three traceplots for parameters shows the stable and well-mixed behavior across chains without visible trends, indicating good mixing.

ESS (Effective Sample Sizes) for all parameters were substantially large, ensuring reliable posterior estimation with minimal autocorrelation, and ACF plot also prove that this model is effective and converges.

3.5 Result

	stockID	posterior_mean_theta	sector	stock
theta[1]	1	0.510612656421657	1	1
theta[2]	2	0.498445881583002	1	2
theta[3]	3	0.5247181952176	1	3
theta[4]	4	0.511196213633361	1	4
theta[5]	5	0.551606529766784	1	5

	sector	sector_mean_theta
1	1	0.53434222253278
2	2	0.512565651493733
3	3	0.50177166674761
4	4	0.519662130708468
5	5	0.527717048491503

Posterior mean estimates of each stock's probability of a positive return were calculated based on the MCMC draws. The posterior means across stocks varied, with most success probabilities ranging from approximately 0.50 to 0.60. We also can see that average success probabilities by sectors are almost similar each other in the table which are near 50 percent. However, sector 1 is the most likely to have positive return than others.

Table 2: Table: Best Stock Within Each Sector Based on Posterior Mean Success Rate

stockID	posterior_mean_theta	sector	stock
8	0.6075014	1	8
15	0.6073803	2	5
26	0.5529039	3	6
31	0.5662480	4	1
41	0.5668229	5	1

I extract the stock that has the highest average probability of positive return for each sector. In sector 1, stock 10 has the highest average probability and so on.

4 Conclusion

In this project, we modeled stock returns using a hierarchical Bayesian framework, treating individual stock returns as Bernoulli trials with sector-level Beta priors. Exploratory data analysis revealed meaningful differences across sectors and stocks. Our Bayesian model was fit using MCMC sampling, and convergence diagnostics confirmed successful convergence. Posterior analysis identified the sector with the highest average success probability and the most promising stock within each sector, providing insights into future investment opportunities.

5 Appendix

5.1 EDA

```
# 1. Basic table: number of 0s and 1s
# Make the basic table
flip_table <- table(stock$flip)

# Convert it into a tidy data frame
flip_summary <- data.frame(
  flip = as.numeric(names(flip_table)),
  count = as.numeric(flip_table),
  proportion = as.numeric(flip_table) / sum(flip_table)
)

# View the summary table
print(flip_summary)
```

```
##   flip count proportion
## 1    0   721  0.4806667
## 2    1   779  0.5193333
```

```r
# 2. Mean flip (success rate) per stock
stock_summary <- stock %>%
  group_by(sector, stock) %>%
  summarize(success_rate = mean(flip),
            num_obs = n(),
            .groups = "drop")

# View the stock summary
print(stock_summary)
```

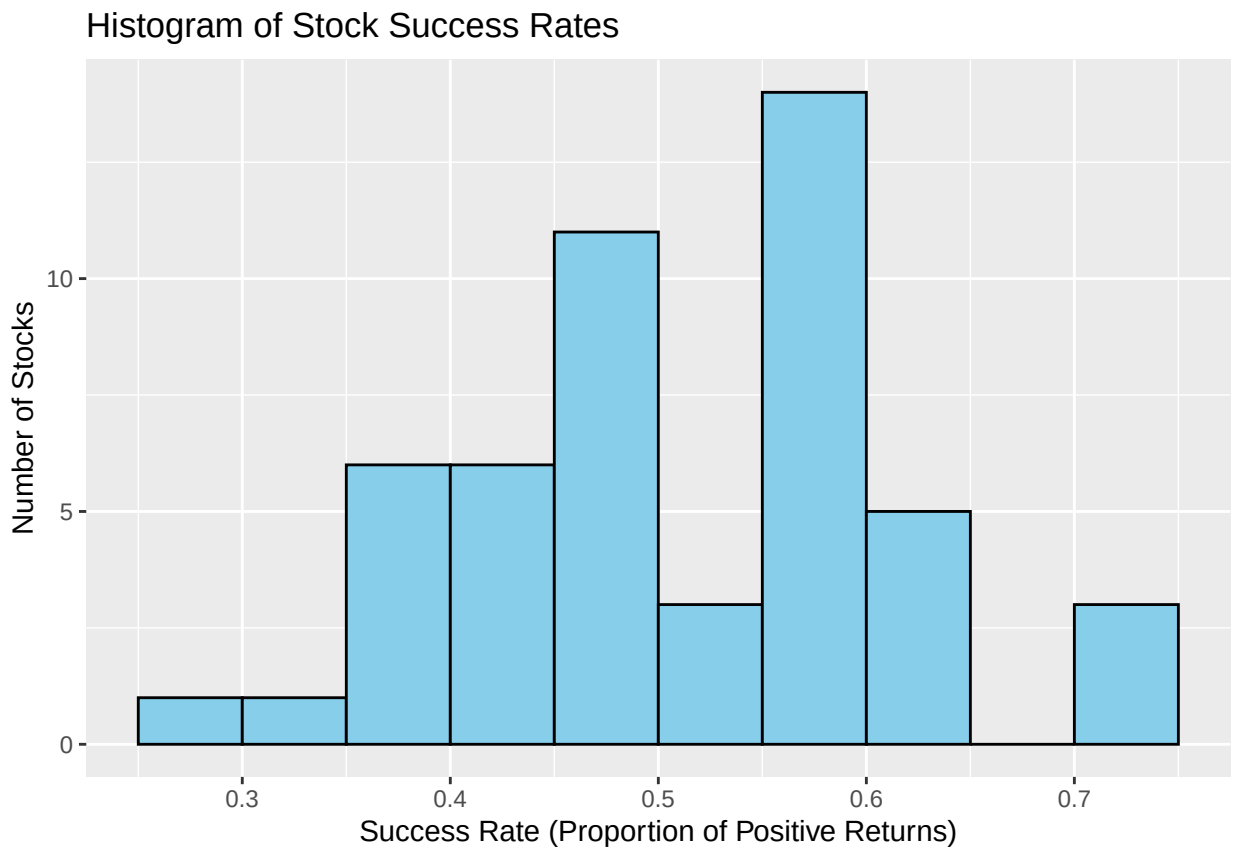
```
## # A tibble: 50 x 4
##    sector stock success_rate num_obs
##     <dbl> <dbl>        <dbl>   <int>
##  1      1     1        0.5        30
##  2      1     2        0.467      30
##  3      1     3        0.533      30
##  4      1     4        0.5        30
##  5      1     5        0.6        30
##  6      1     6        0.467      30
##  7      1     7        0.6        30
##  8      1     8        0.733      30
##  9      1     9        0.433      30
## 10      1    10        0.733      30
## # i 40 more rows
```

```r
# 3. Mean flip (success rate) per sector
sector_summary <- stock %>%
  group_by(sector) %>%
  summarize(sector_success_rate = mean(flip))
```

```r
# View the sector summary
print(sector_summary)
```

```
## # A tibble: 5 x 2
##   sector sector_success_rate
##    <dbl>               <dbl>
## 1      1               0.557
## 2      2               0.503
## 3      3               0.477
## 4      4               0.52
## 5      5               0.54
```

```r
# 4. Plot: Histogram of stock success rates
ggplot(stock_summary, aes(x = success_rate)) +
  geom_histogram(binwidth = 0.05, boundary = 0, color = "black", fill = "skyblue") +
  labs(title = "Histogram of Stock Success Rates",
       x = "Success Rate (Proportion of Positive Returns)",
       y = "Number of Stocks")
```




```r
# 5. Plot: Boxplot of success rates across sectors
ggplot(stock_summary, aes(x = factor(sector), y = success_rate)) +
  geom_boxplot(fill = "lightgreen") +
  labs(title = "Boxplot of Stock Success Rates by Sector",
       x = "Sector",
       y = "Stock Success Rate") +
  theme_minimal()
```

5.2 Model Building

```
# Set random seed for reproducibility
set.seed(3303)

# Prepare data for JAGS
# Reindex stocks uniquely (sector 1-5 and stock 1-10 inside)
stock$stockID <- (stock$sector - 1) * 10 + stock$stock

# Number of sectors and stocks
n_sectors <- length(unique(stock$sector))
n_stocks <- length(unique(stock$stockID))
```

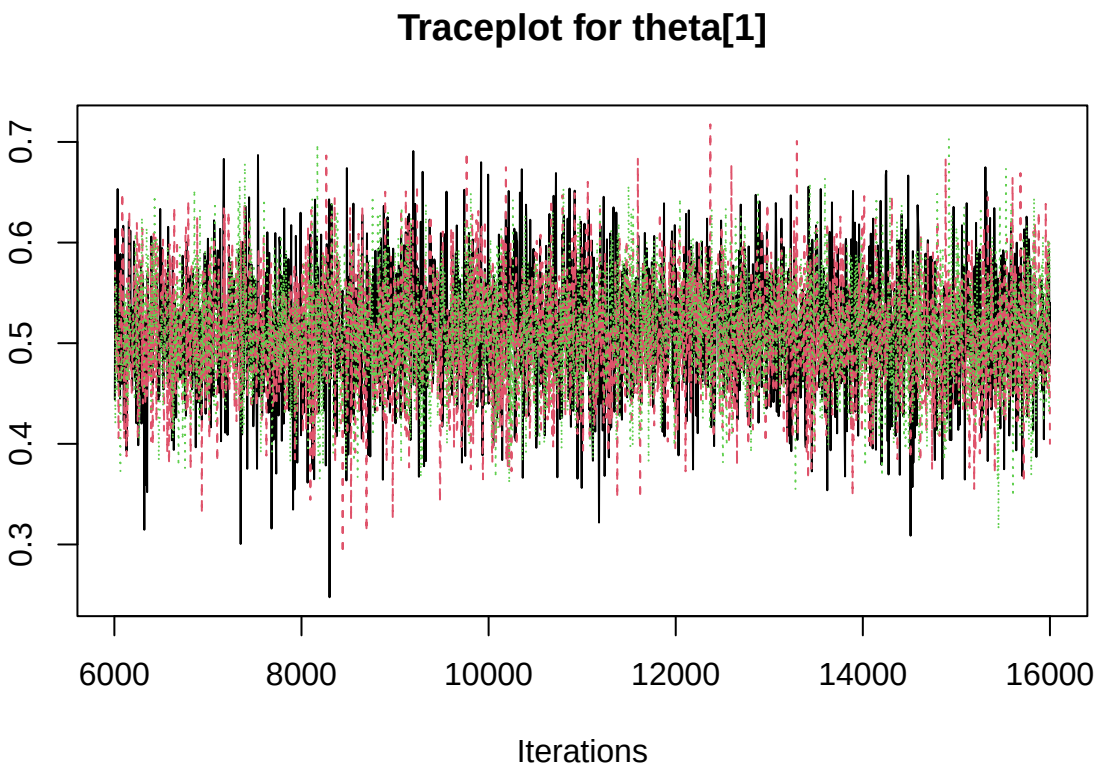
```
# Bundle data for JAGS
jags_data <- list(
  flip = stock$flip,
  stockID = stock$stockID,
  sectorID = stock$sector,
  N = nrow(stock),
  n_sectors = n_sectors,
  n_stocks = n_stocks
)

# Write the model
model_string <- "
model {
  for (n in 1:N) {
    flip[n] ~ dbern(theta[stockID[n]])
  }

  for (i in 1:n_stocks) {
    theta[i] ~ dbeta(alpha[sectorID[i]], beta[sectorID[i]])
  }

  for (j in 1:n_sectors) {
    alpha[j] ~ dgamma(1, 0.1)
    beta[j] ~ dgamma(1, 0.1)
  }
}
"

# Write model to temporary file
```

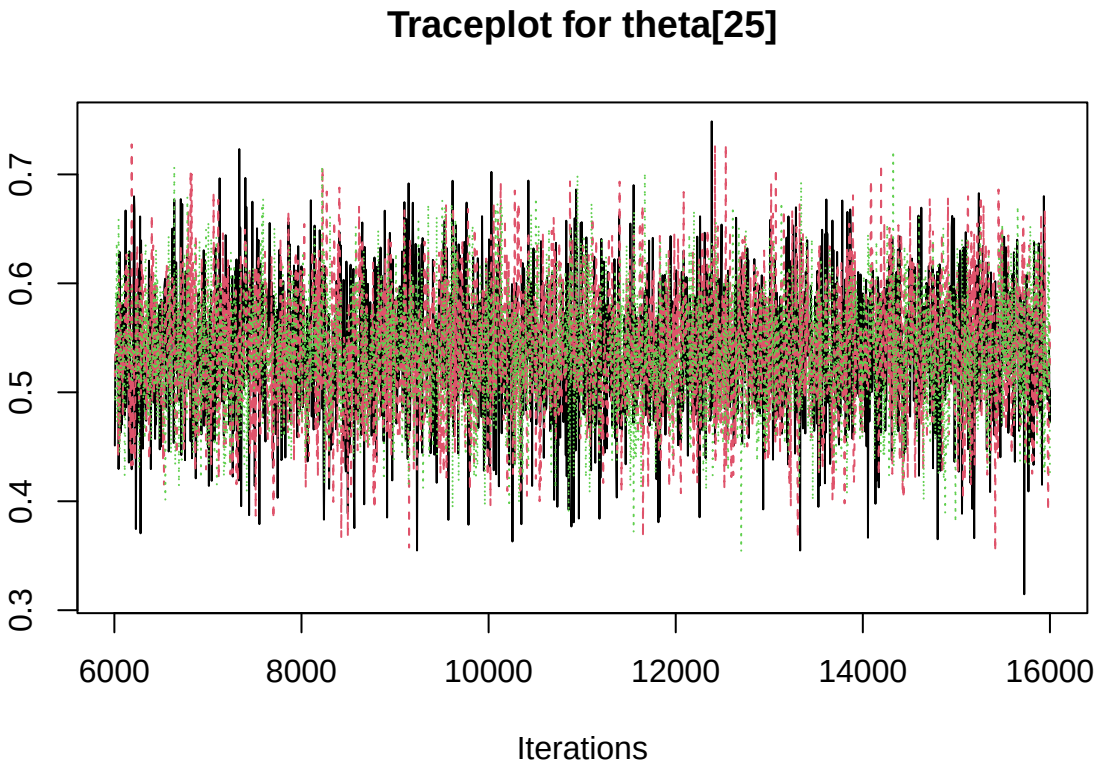
```r
writeLines(model_string, con = "model.bug")


# Set starting values
inits <- function() {
  list(
    alpha = rgamma(n_sectors, 1, 0.1),
    beta = rgamma(n_sectors, 1, 0.1),
    theta = rbeta(n_stocks, 1, 1)
  )
}


# Parameters to monitor
params <- c("theta", "alpha", "beta")


# MCMC settings
n.chains <- 3      # Number of chains
n.iter <- 10000    # Total iterations per chain
n.burnin <- 5000   # Burn-in
n.thin <- 5


# Run JAGS
model <- jags.model(file = "model.bug",
                    data = jags_data,
                    inits = inits,
                    n.chains = n.chains,
                    n.adapt = 1000)
```

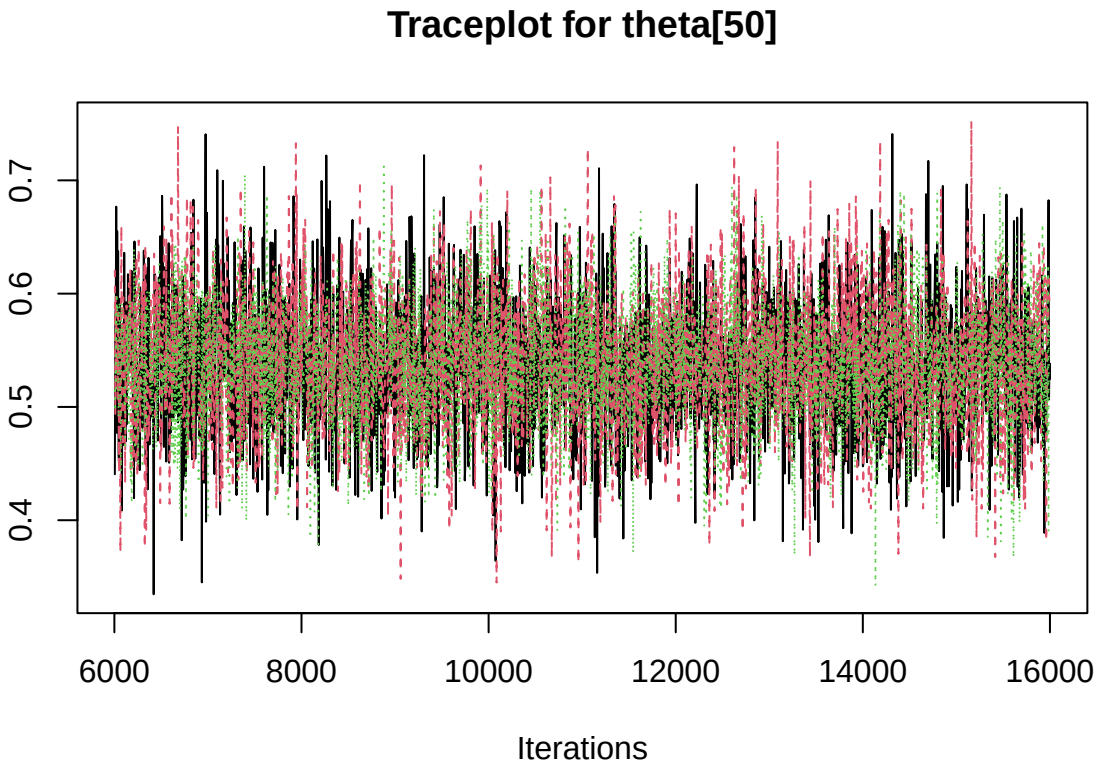
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
```

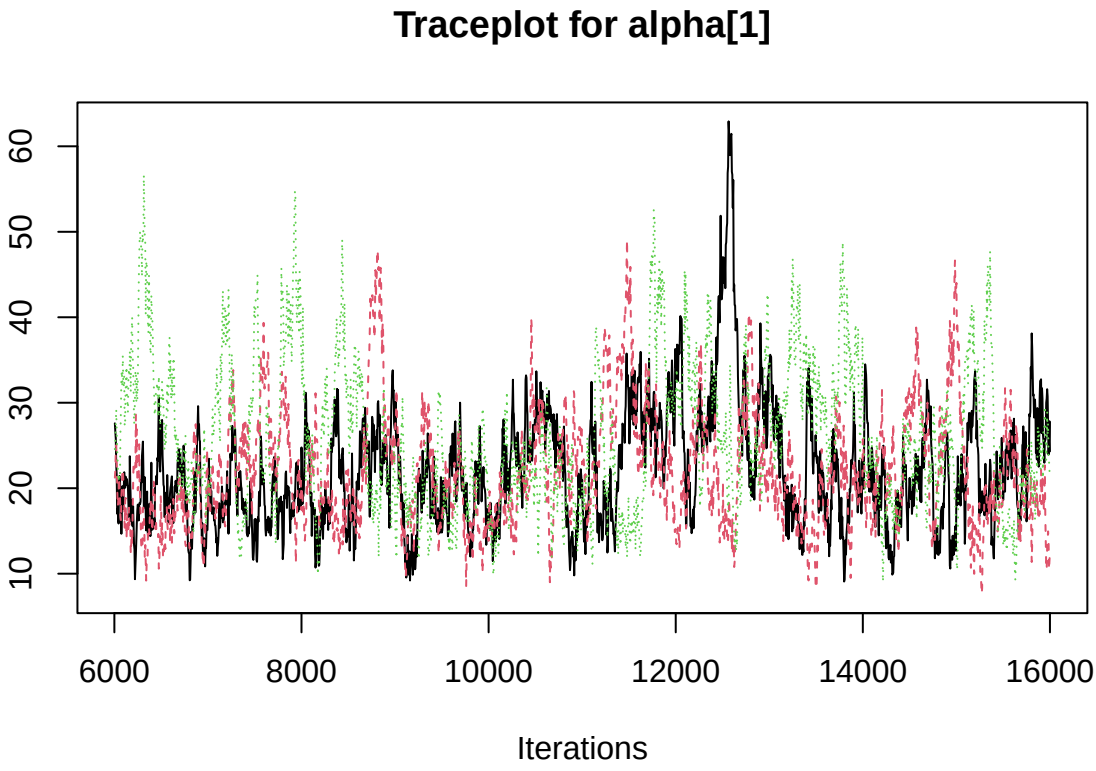
```
## Graph information:
##    Observed stochastic nodes: 1500
##    Unobserved stochastic nodes: 60
##    Total graph size: 4565
##
## Initializing model
```

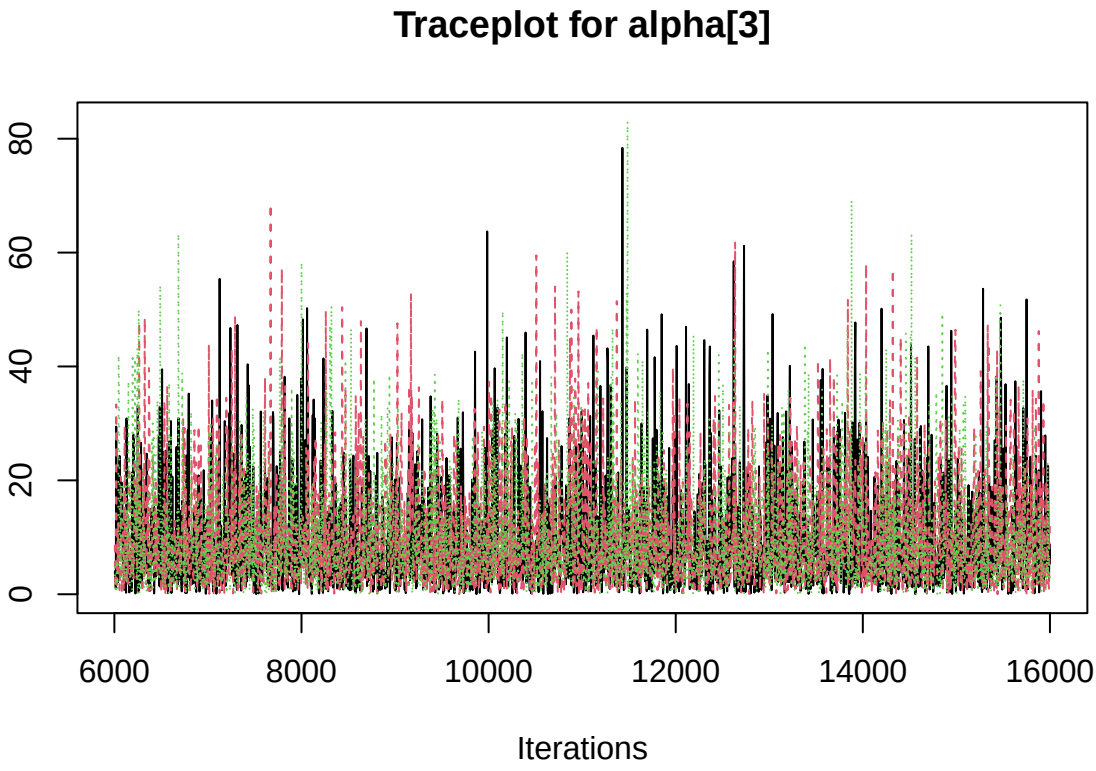
```r
update(model, n.iter = n.burnin)  # Burn-in


# Draw samples
samples <- coda.samples(model,
                        variable.names = params,
                        n.iter = n.iter,
                        thin = n.thin)


# Combine chains
samples_combined <- do.call(rbind, samples)


# Traceplots for selected parameters only
selected_params <- c("theta[1]", "theta[25]", "theta[50]", "alpha[1]", "alpha[3]", "beta[1]",


# Traceplots
for (param in selected_params) {
  traceplot(samples[, param], main = paste("Traceplot for", param))
}
```

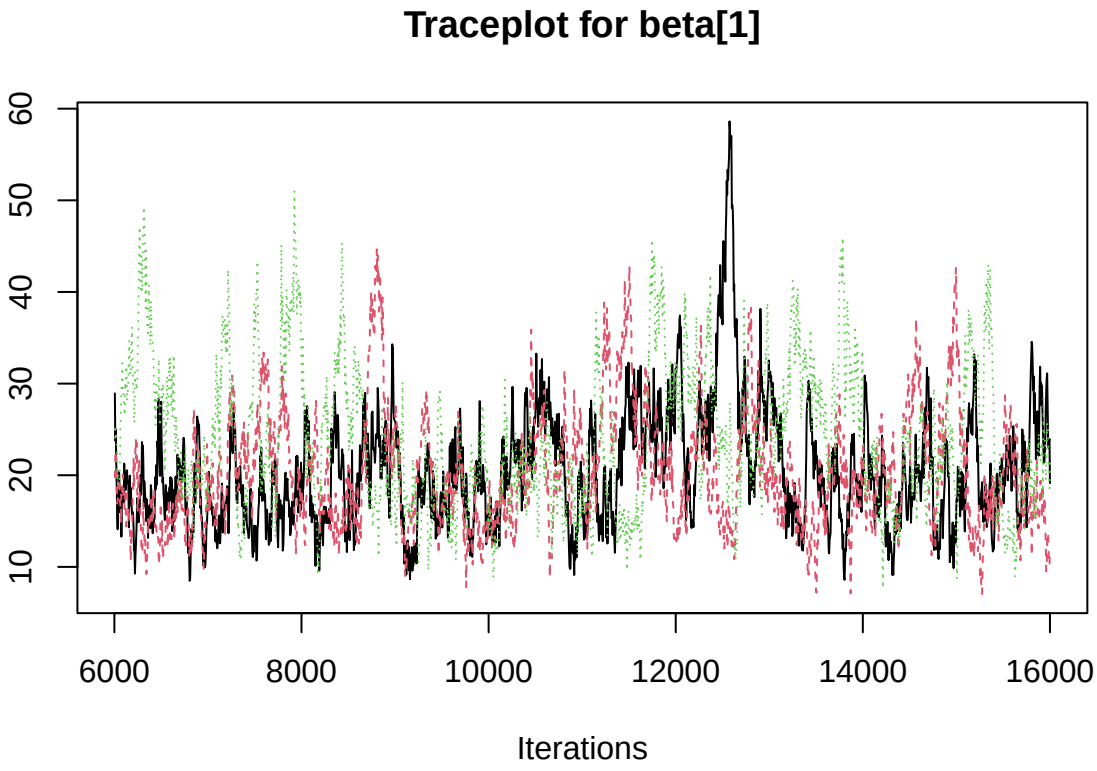

Traceplot for theta[1]
0.3
0.4
0.5
0.6
0.7
6000
8000
10000
12000
14000
16000
Iterations

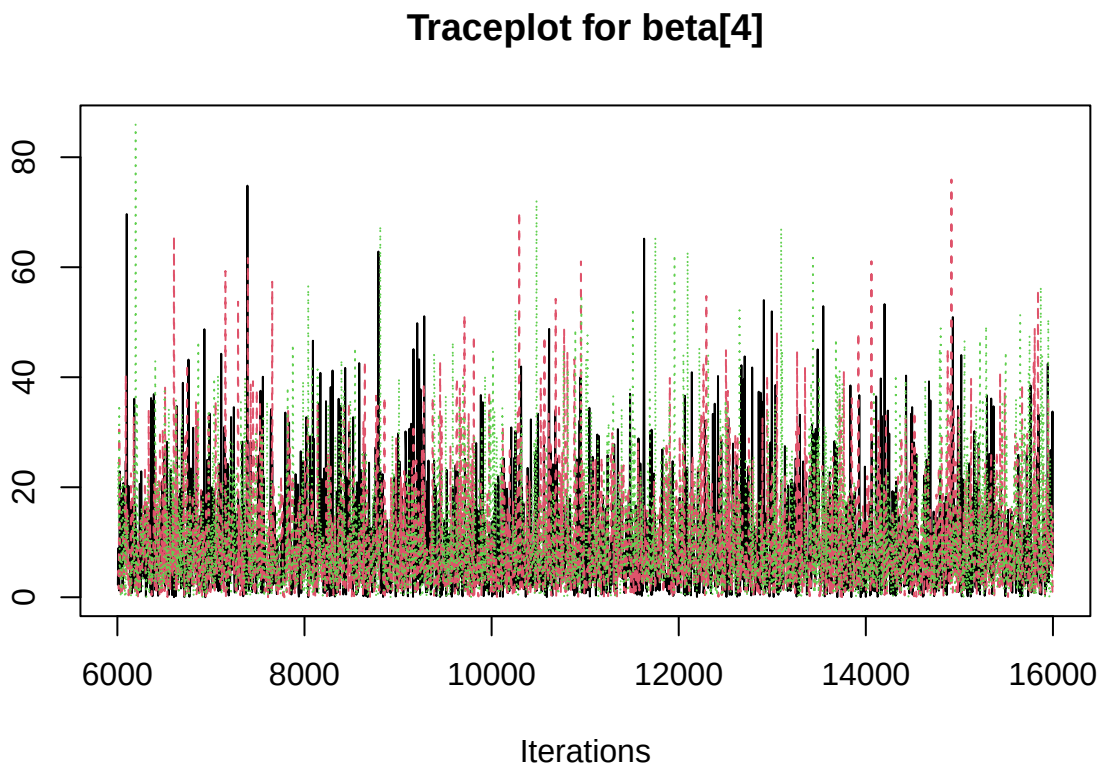

Traceplot for theta[25]
0.3
0.4
0.5
0.6
0.7
6000
8000
10000
12000
14000
16000
Iterations


Traceplot for theta[50]
0.4
0.5
0.6
0.7
6000
8000
10000
12000
14000
16000
Iterations


Traceplot for alpha[1]
10
20
30
40
50
60
6000
8000
10000
12000
14000
16000
Iterations


Traceplot for alpha[3]
0
20
40
60
80
6000
8000
10000
12000
14000
16000
Iterations


Traceplot for beta[1]
10
20
30
40
50
60
6000
8000
10000
12000
14000
16000
Iterations

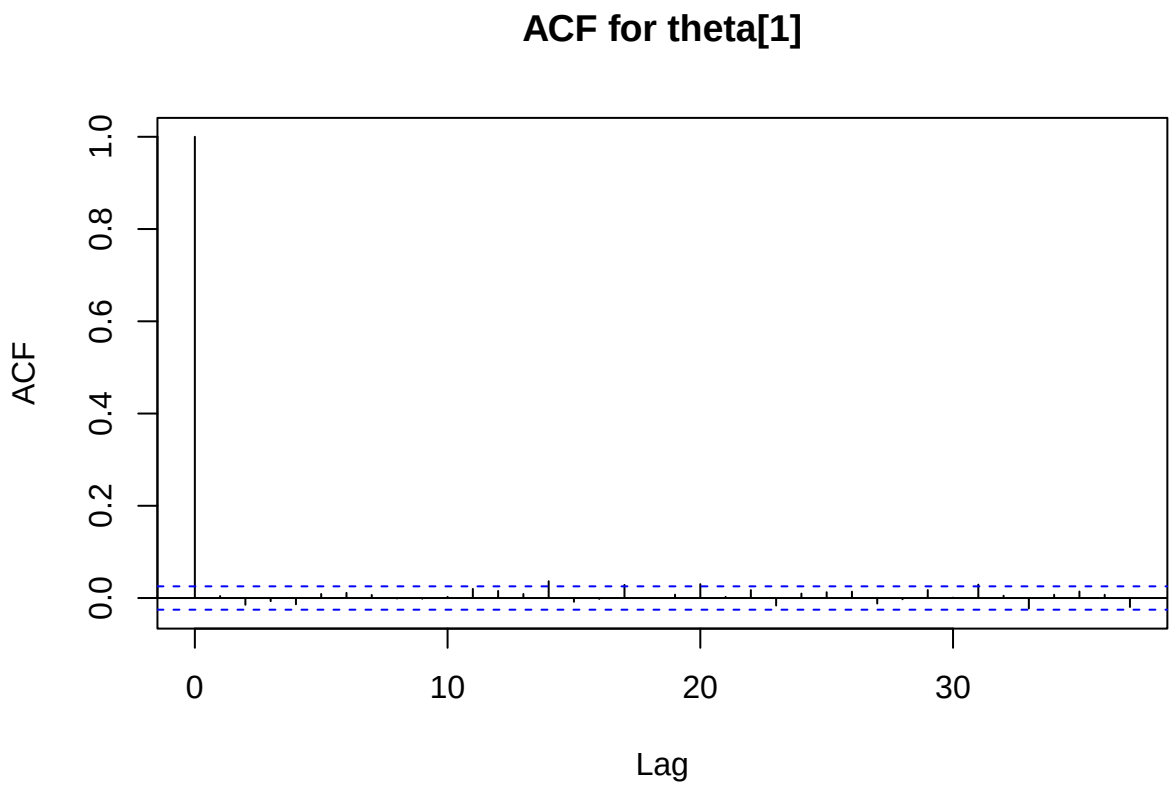


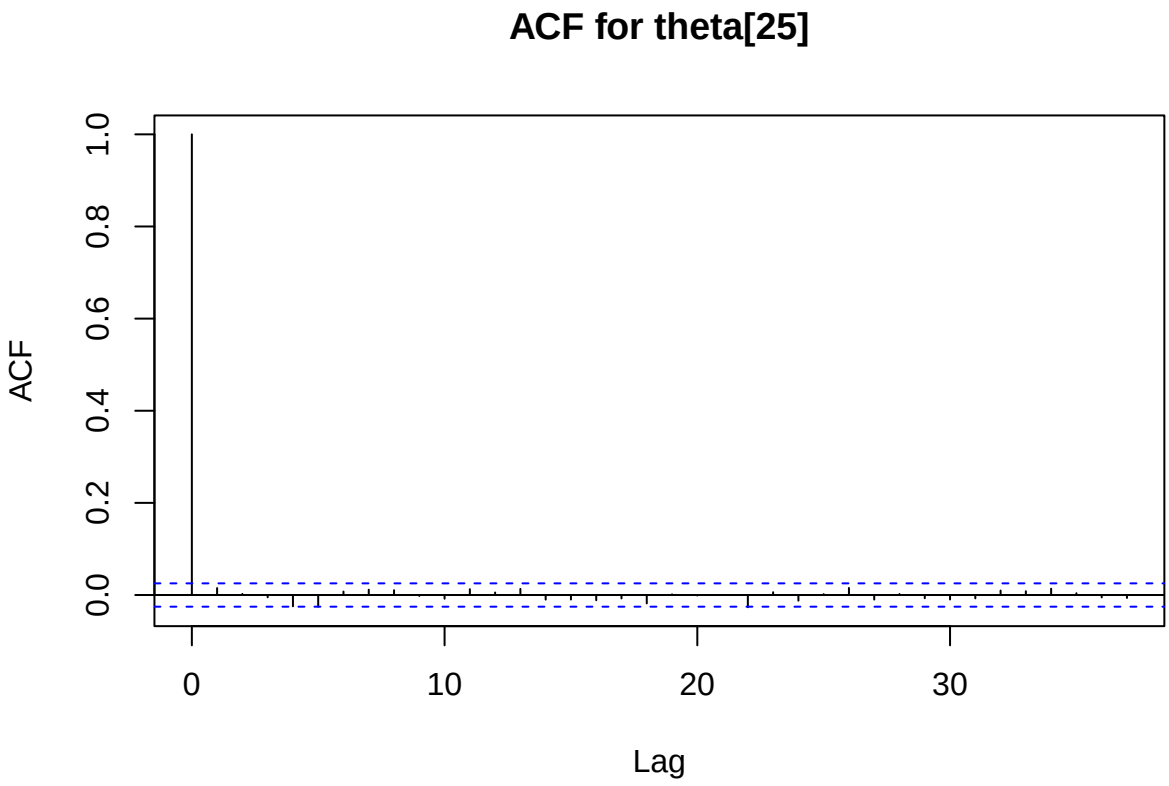

```
# Effective Sample Size (ESS) for selected parameters
ess_subset <- effectiveSize(samples[, selected_params])
ess_table <- data.frame(
  Parameter = names(ess_subset),
  ESS = round(as.numeric(ess_subset), 0)
)
print(ess_table)
```

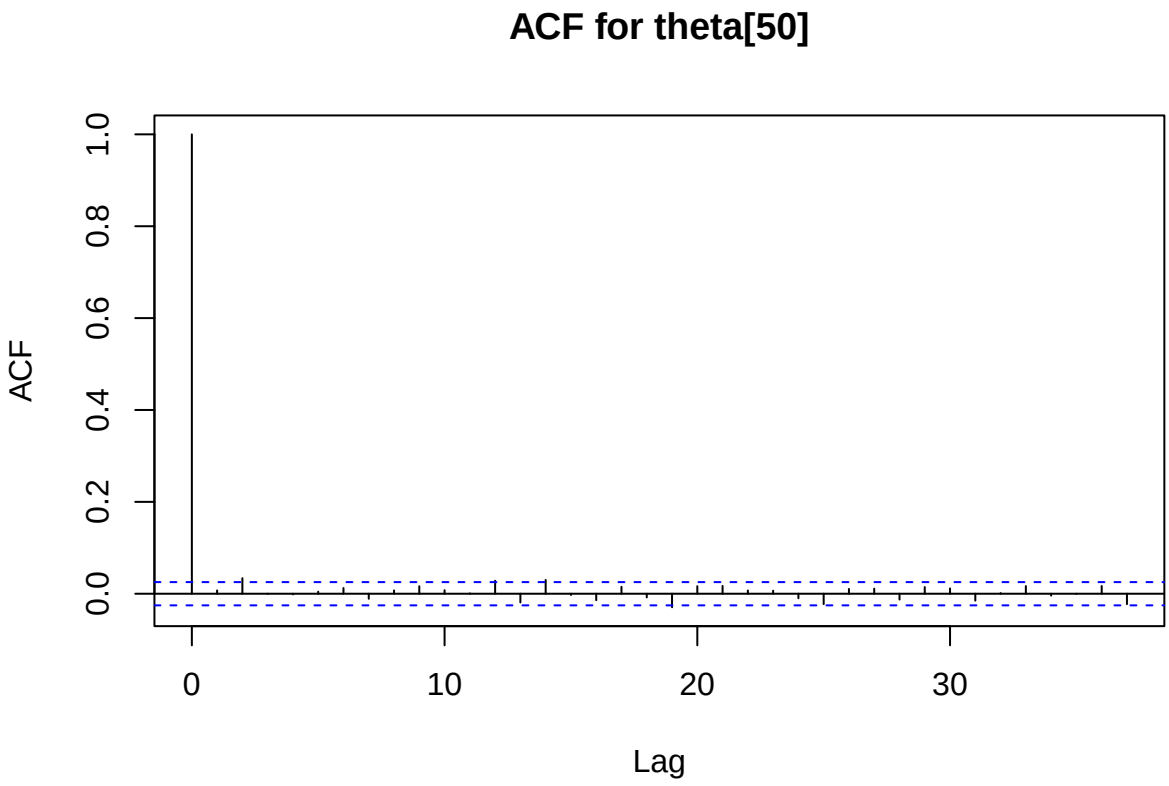
```
##   Parameter  ESS
## 1  theta[1] 6279
## 2 theta[25] 5787
## 3 theta[50] 5766
## 4  alpha[1]  164
## 5  alpha[3] 6000
```

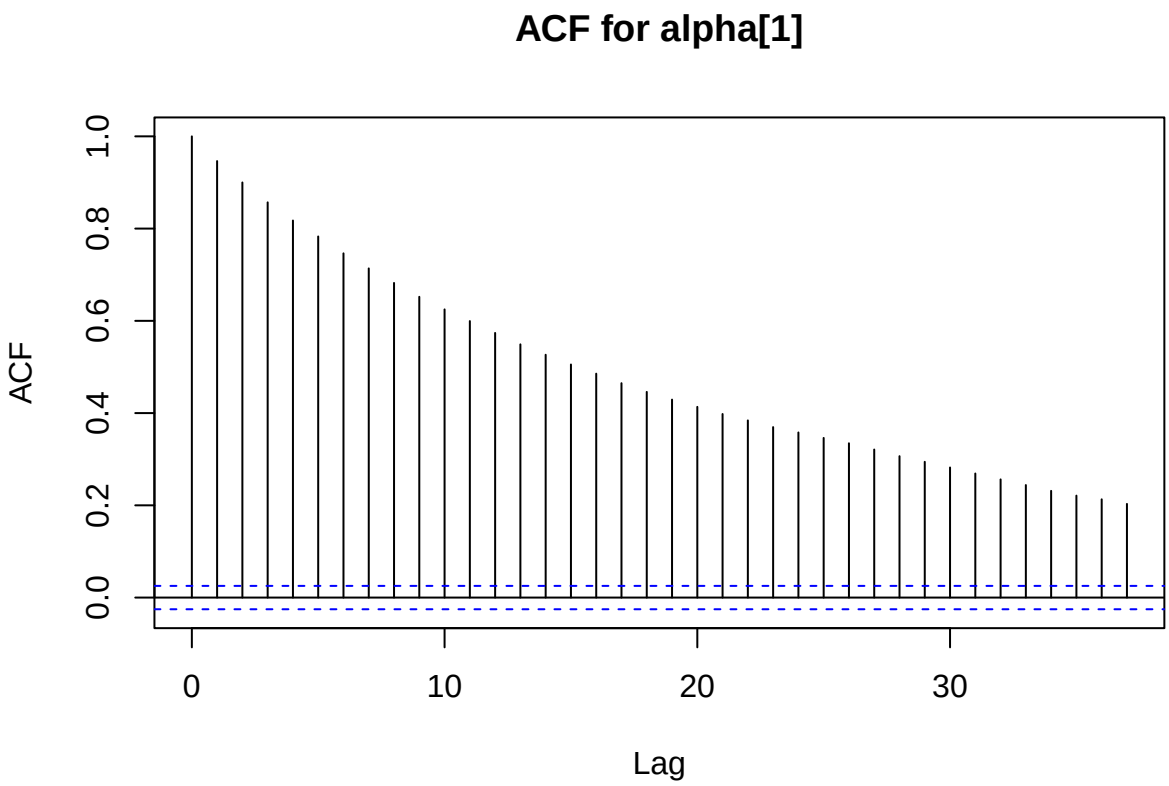
```
## 6   beta[1]  149
## 7   beta[4] 6423
```

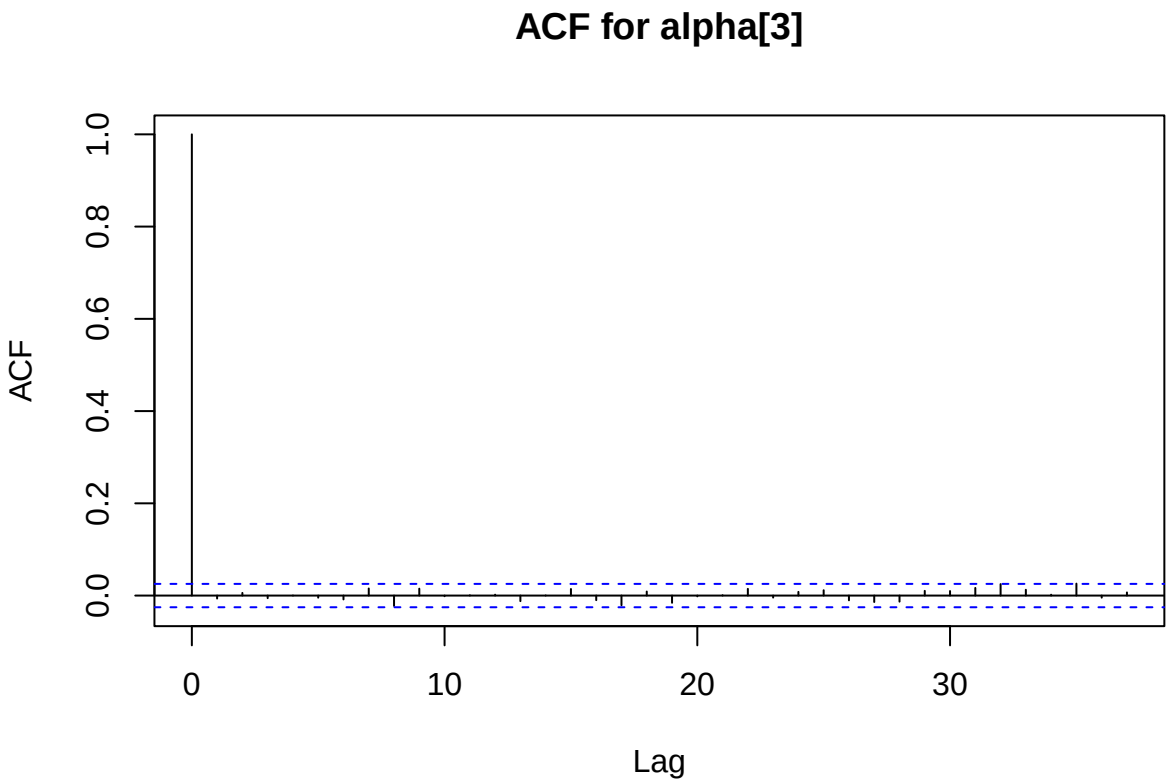
```
# ACF plots one by one
for (param in selected_params) {
  acf(samples_combined[, param], main = paste("ACF for", param))
}
```

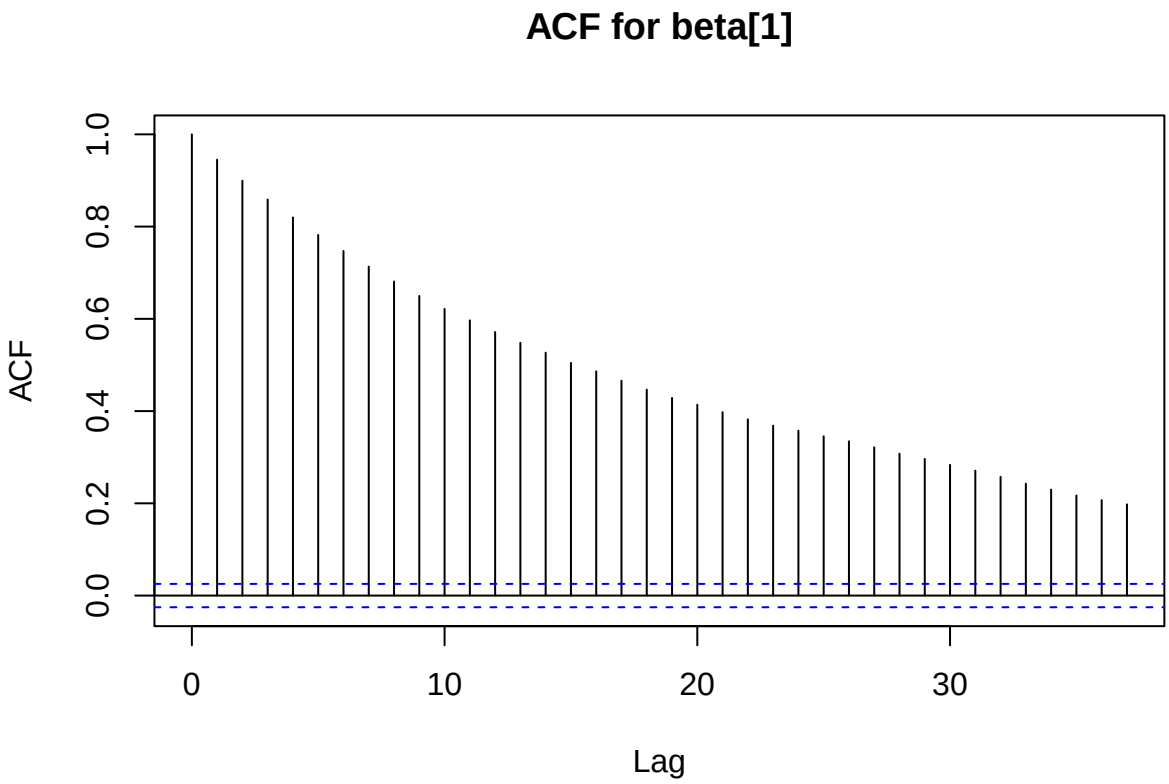


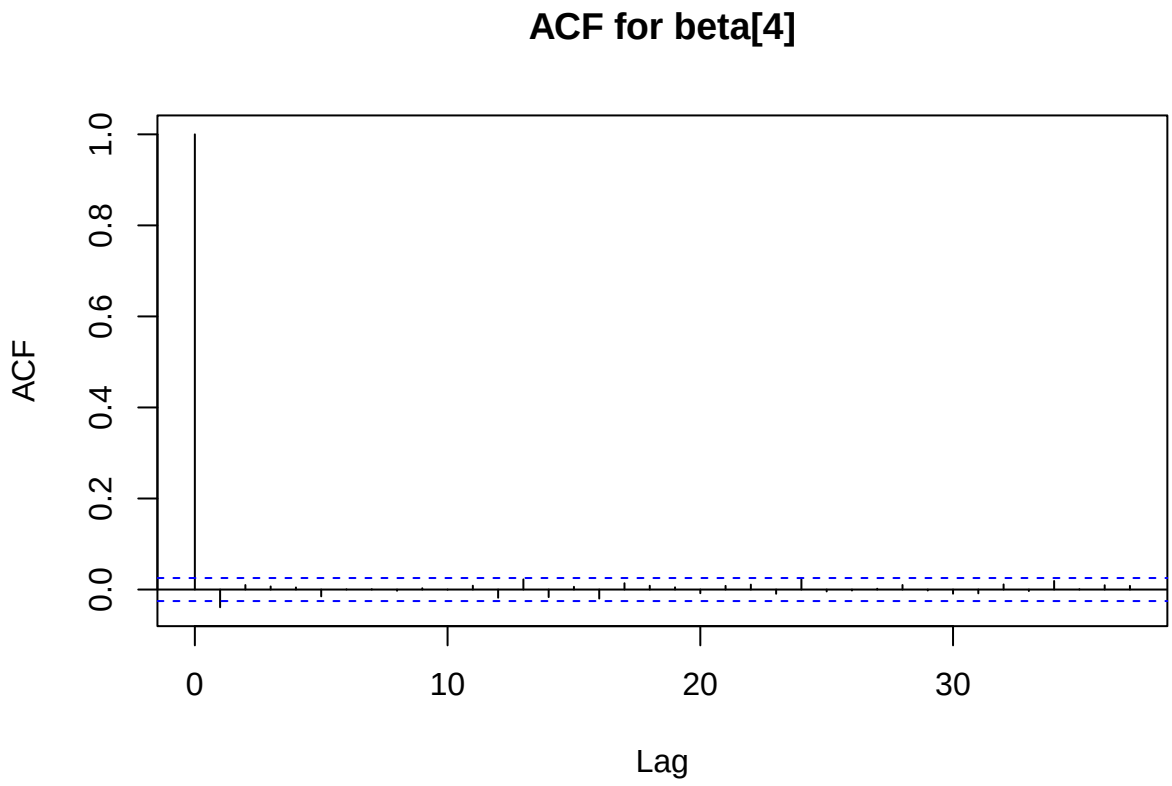

ACF for theta[25]
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
Lag


ACF for theta[50]
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
Lag


ACF for alpha[1]
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
Lag


ACF for alpha[3]
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
Lag


ACF for beta[1]
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
Lag



5.3 Result Summary

```
# samples_combined has all MCMC draws


# Extract only theta samples
theta_samples <- samples_combined[, grep("^theta", colnames(samples_combined))]


# Calculate posterior mean for each stock
posterior_means_theta <- colMeans(theta_samples)


# Make a dataframe for stocks
posterior_summary <- data.frame(
  stockID = 1:50,
```

```
  posterior_mean_theta = posterior_means_theta
)

# Recover sector and stock number inside sector
posterior_summary$sector <- ceiling(posterior_summary$stockID / 10)
posterior_summary$stock <- posterior_summary$stockID - (posterior_summary$sector - 1) * 10

# View the summary
print(posterior_summary)
```

```
##           stockID posterior_mean_theta sector stock
## theta[1]        1            0.5111606      1     1
## theta[2]        2            0.4964088      1     2
## theta[3]        3            0.5242340      1     3
## theta[4]        4            0.5108416      1     4
## theta[5]        5            0.5530283      1     5
## theta[6]        6            0.4971426      1     6
## theta[7]        7            0.5528694      1     7
## theta[8]        8            0.6072761      1     8
## theta[9]        9            0.4831754      1     9
## theta[10]      10            0.6081898      1    10
## theta[11]      11            0.4702009      2     1
## theta[12]      12            0.4831783      2     2
## theta[13]      13            0.5654058      2     3
## theta[14]      14            0.5679584      2     4
## theta[15]      15            0.6074555      2     5
## theta[16]      16            0.5104895      2     6
## theta[17]      17            0.4423746      2     7
## theta[18]      18            0.4822784      2     8
## theta[19]      19            0.5109533      2     9
```

```
## theta[20]      20            0.4834284      2    10
## theta[21]      21            0.4684564      3     1
## theta[22]      22            0.5394026      3     2
## theta[23]      23            0.4568459      3     3
## theta[24]      24            0.4557436      3     4
## theta[25]      25            0.5384138      3     5
## theta[26]      26            0.5515056      3     6
## theta[27]      27            0.4988163      3     7
## theta[28]      28            0.5533650      3     8
## theta[29]      29            0.4280509      3     9
## theta[30]      30            0.5258278      3    10
## theta[31]      31            0.5674537      4     1
## theta[32]      32            0.4841585      4     2
## theta[33]      33            0.5399082      4     3
## theta[34]      34            0.4974535      4     4
## theta[35]      35            0.4706638      4     5
## theta[36]      36            0.5112278      4     6
## theta[37]      37            0.5526852      4     7
## theta[38]      38            0.4981398      4     8
## theta[39]      39            0.5375287      4     9
## theta[40]      40            0.5392520      4    10
## theta[41]      41            0.5661348      5     1
## theta[42]      42            0.4829654      5     2
## theta[43]      43            0.5242351      5     3
## theta[44]      44            0.5656187      5     4
## theta[45]      45            0.4569388      5     5
## theta[46]      46            0.5525250      5     6
## theta[47]      47            0.5395215      5     7
## theta[48]      48            0.5388983      5     8
## theta[49]      49            0.5118490      5     9
```

```
## theta[50]      50            0.5387058      5    10
```

```
# Sector-level average posterior mean
sector_summary_post <- posterior_summary %>%
  group_by(sector) %>%
  summarize(sector_mean_theta = mean(posterior_mean_theta))

print(sector_summary_post)
```

```
## # A tibble: 5 x 2
##   sector sector_mean_theta
##    <dbl>             <dbl>
## 1      1             0.534
## 2      2             0.512
## 3      3             0.502
## 4      4             0.520
## 5      5             0.528
```

```
# Identify best sector
best_sector <- sector_summary_post %>%
  filter(sector_mean_theta == max(sector_mean_theta))

print(best_sector)
```

```
## # A tibble: 1 x 2
##   sector sector_mean_theta
##    <dbl>             <dbl>
## 1      1             0.534
```

```r
# Identify best stock within each sector
best_stock_each_sector <- posterior_summary %>%
  group_by(sector) %>%
  filter(posterior_mean_theta == max(posterior_mean_theta))


print(best_stock_each_sector)
```

```
## # A tibble: 5 x 4
## # Groups:   sector [5]
##   stockID posterior_mean_theta sector stock
##     <int>                <dbl>  <dbl> <dbl>
## 1      10                0.608      1    10
## 2      15                0.607      2     5
## 3      28                0.553      3     8
## 4      31                0.567      4     1
## 5      41                0.566      5     1
```