HW9

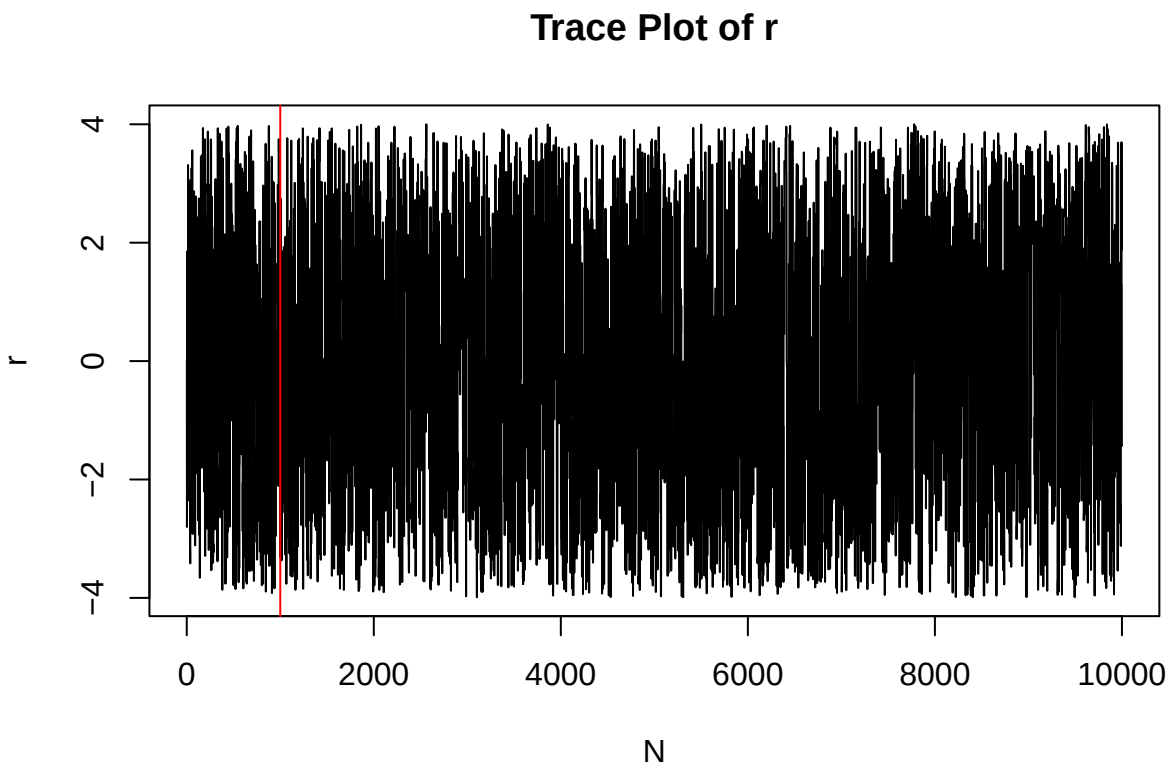
Lun Li.14415

2025-04-14

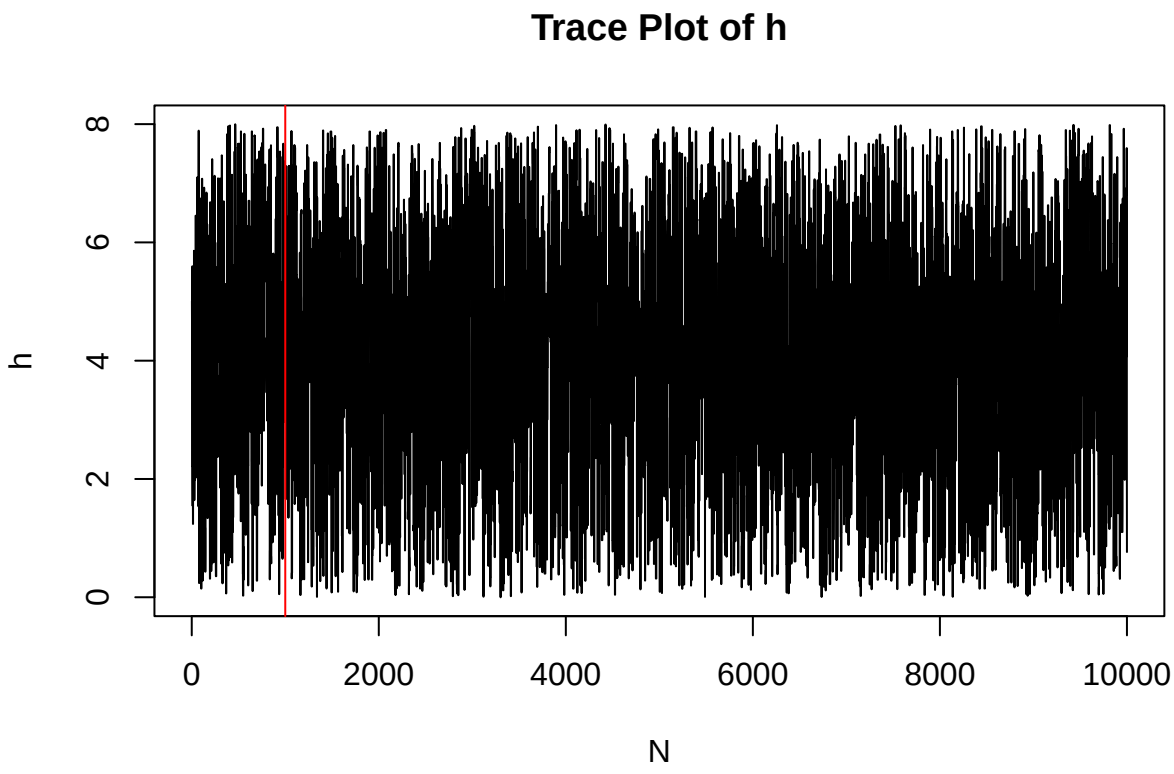
Problem 1

```r
set.seed(123)
N <- 10000
burn_in <- 1000
sigma_r <- 5
sigma_h <- 4.61

r_samples <- numeric(N)
h_samples <- numeric(N)
accept_r <- 0
accept_h <- 0

# Initial values (Set as mean of prior)
r_samples[1] <- 0
h_samples[1] <- 5

prior <- function(r, h) {
  if (abs(r) < 5 && h > 0 && h < 10) {
    return((1/1250) * (5 - abs(r)) * (10 - h))
  } else {
    return(0)
  }
}

likelihood <- function(r, h) {
  if (abs(r) < 4 && h > 0 && h < 8) {
    return((1/32) * r^2 + h / 16)
  } else {
    return(0)
  }
}

posterior <- function(r, h) {
  return(prior(r, h) * likelihood(r, h))
}

for (t in 2:N) {
  # Current values
  r_curr <- r_samples[t - 1]
```

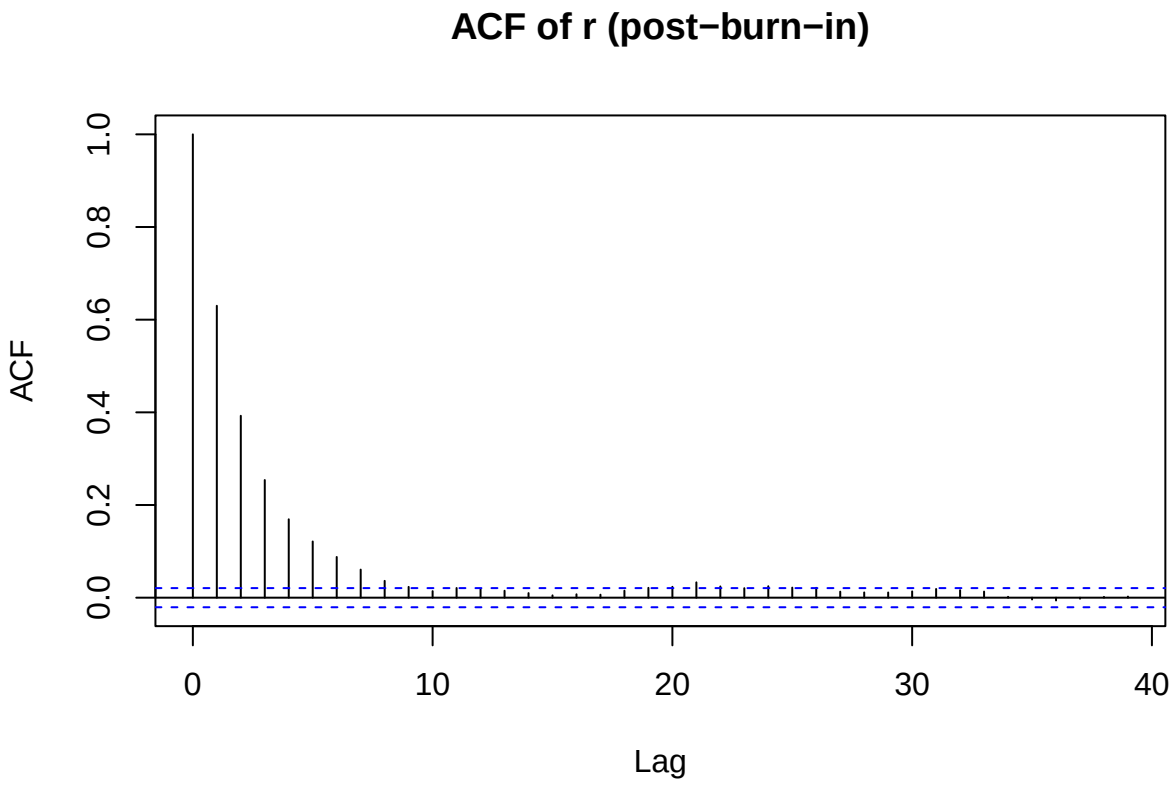
```r
  h_curr <- h_samples[t - 1]

  # 1. Update r
  r_prop <- r_curr + rnorm(1, mean = 0, sd = sigma_r)
  alpha_r <- min(1, posterior(r_prop, h_curr) / posterior(r_curr, h_curr))

  if (runif(1) < alpha_r) {
    r_new <- r_prop
    accept_r <- accept_r + 1
  } else {
    r_new <- r_curr
  }

  # 2: Update h
  h_prop <- h_curr + rnorm(1, mean = 0, sd = sigma_h)
  alpha_h <- min(1, posterior(r_new, h_prop) / posterior(r_new, h_curr))

  if (runif(1) < alpha_h) {
    h_new <- h_prop
    accept_h <- accept_h + 1
  } else {
    h_new <- h_curr
  }

  r_samples[t] <- r_new
  h_samples[t] <- h_new
}

library(coda)

r_chain <- r_samples[(burn_in + 1):N]
h_chain <- h_samples[(burn_in + 1):N]

# Trace plots
plot(r_samples, type = "l", main = "Trace Plot of r", xlab = "N", ylab = "r")
abline(v = burn_in, col = "red")
```



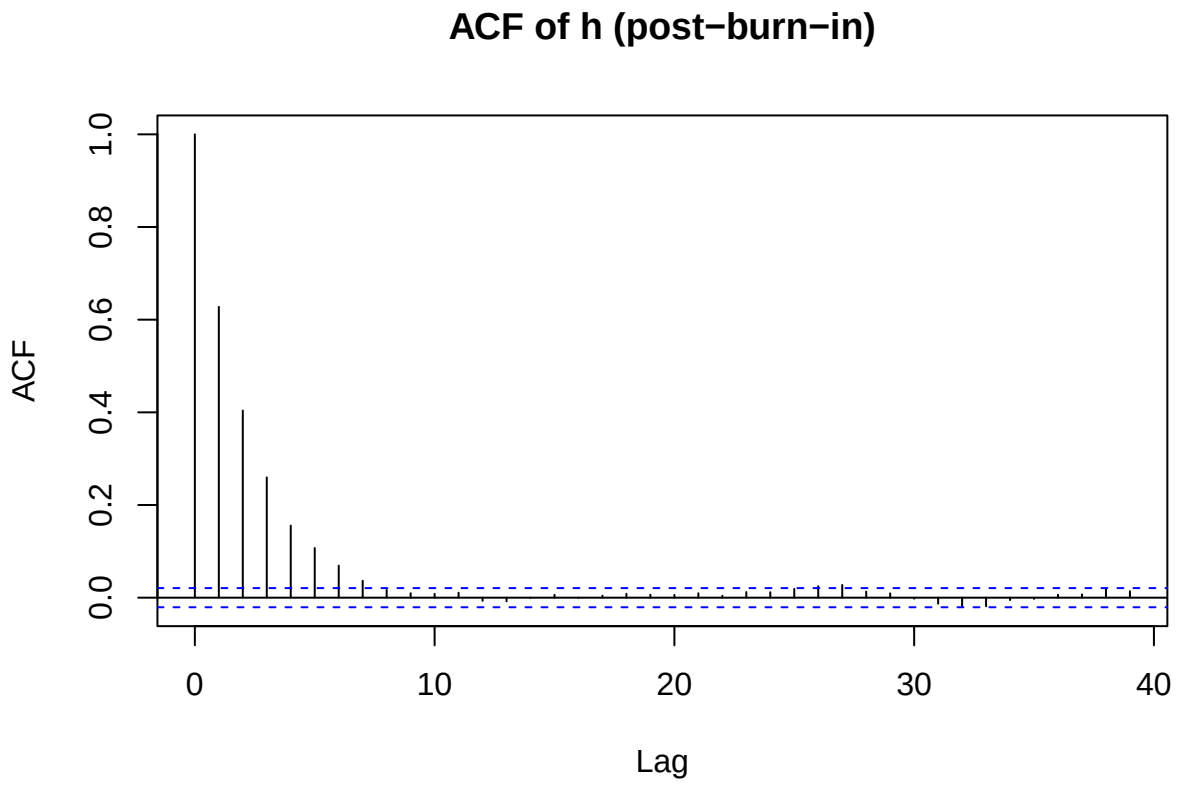

```
plot(h_samples, type = "l", main = "Trace Plot of h", xlab = "N", ylab = "h")
abline(v = burn_in, col = "red")
```



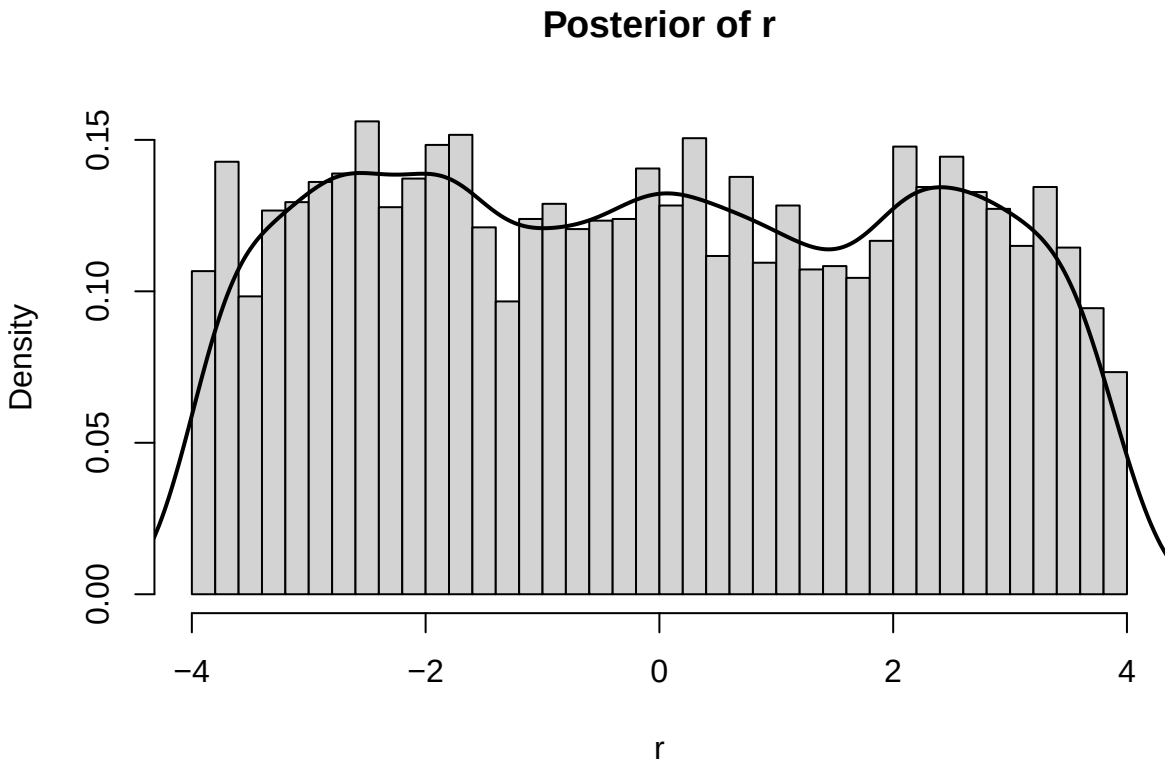

```
# ACF plots
acf(as.mcmc(r_chain), main = "ACF of r (post-burn-in)")
```



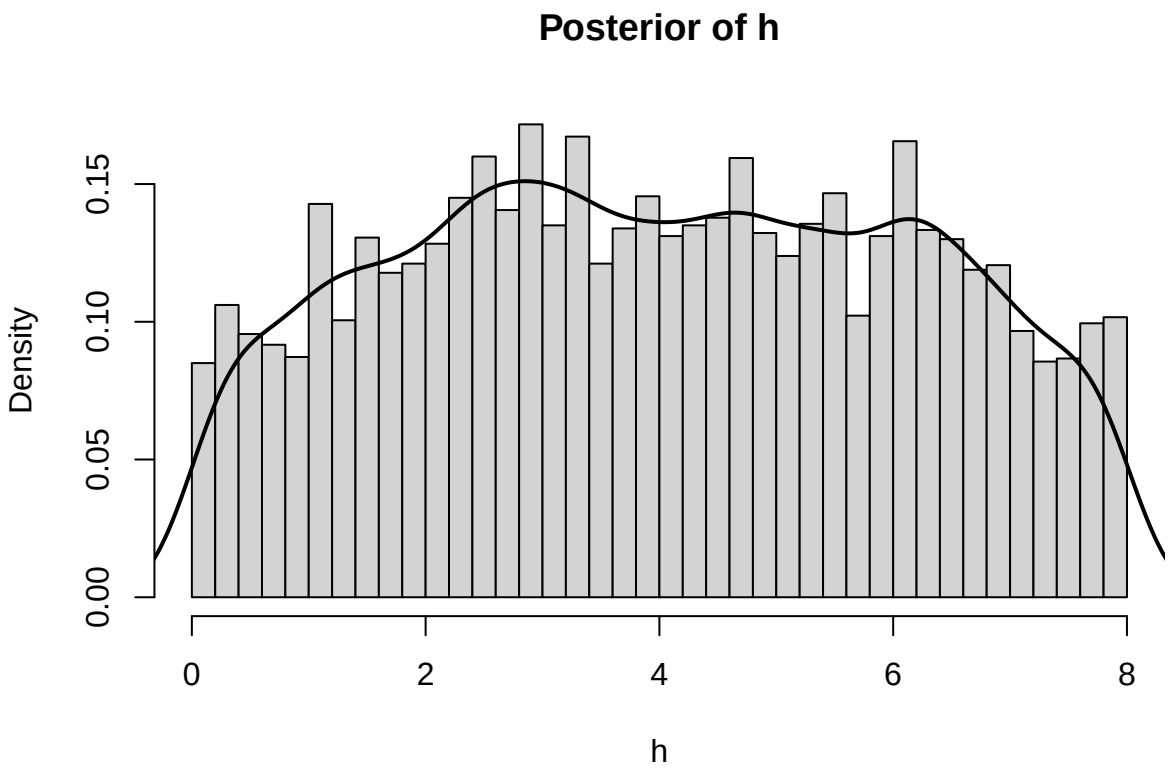

```
acf(as.mcmc(h_chain), main = "ACF of h (post-burn-in)")
```




```r
# Posterior histograms and densities
hist(r_chain, breaks = 50, probability = TRUE, main = "Posterior of r", xlab = "r")
lines(density(r_chain), lwd = 2)
```




```r
hist(h_chain, breaks = 50, probability = TRUE, main = "Posterior of h", xlab = "h")
lines(density(h_chain), lwd = 2)
```




```
# Posterior summaries
cat("Posterior Mean (r):", mean(r_chain), "  Var:", var(r_chain), "\n")
```

```
## Posterior Mean (r): -0.08747496   Var: 5.1547
```

```
cat("Posterior Mean (h):", mean(h_chain), "  Var:", var(h_chain), "\n")
```

```
## Posterior Mean (h): 3.994898   Var: 4.6506
```

```
# Correlation and Effective Sample Size
cat("Posterior Correlation (rho_rh | G=1):", cor(r_chain, h_chain), "\n")
```

```
## Posterior Correlation (rho_rh | G=1): 0.03344716
```

```
cat("Acceptance Rate for r:", accept_r / N, "\n")
```

```
## Acceptance Rate for r: 0.4797
```

```
cat("Acceptance Rate for h:", accept_h / N, "\n")
```

```
## Acceptance Rate for h: 0.5102
```

```r
cat("Effective Sample Size:\n")
```

```
## Effective Sample Size:
```

```r
print(effectiveSize(as.mcmc(cbind(r_chain, h_chain))))
```

```
##  r_chain  h_chain
## 2042.745 1974.656
```

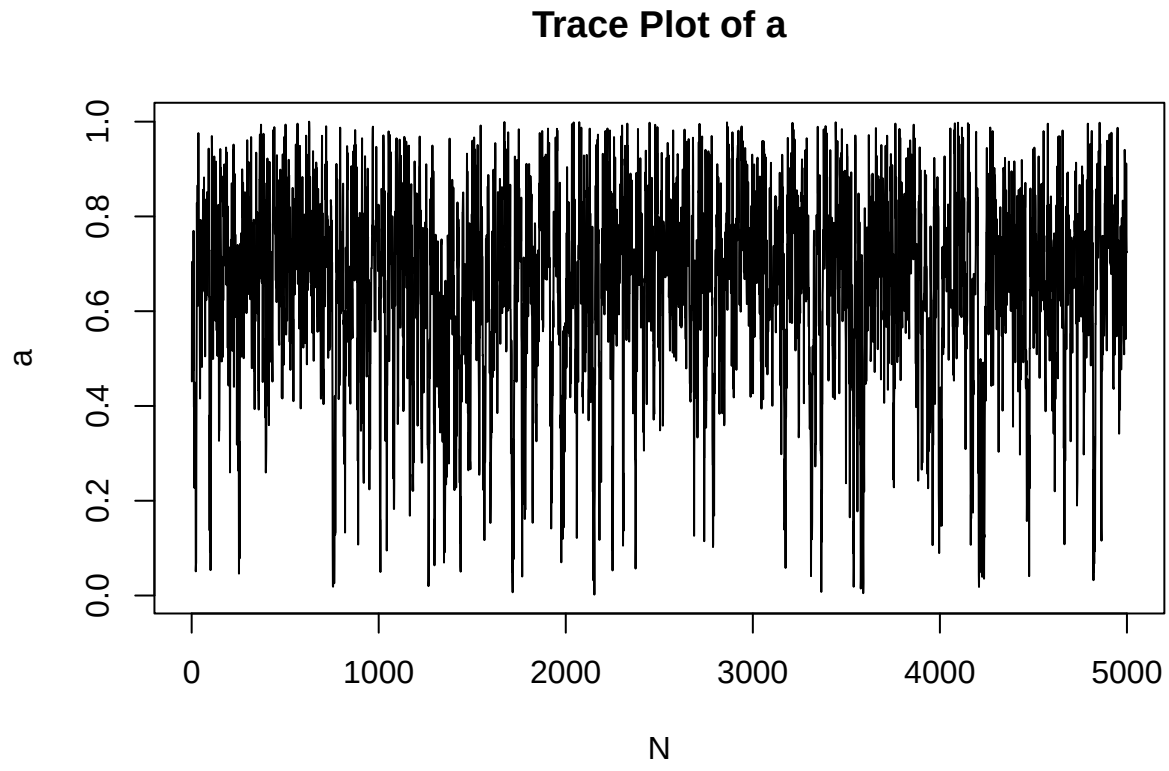
Problem 2

```r
set.seed(123)
n <- 4
a <- 0.5
B <- 10000

loss <- function(a, theta) {
  ifelse(a > theta, (a - theta)^2, 4 * (theta - a))
}

for (y in 0:n) {
  theta_samples <- rbeta(B, y + 1, n - y + 1)
  losses <- loss(a, theta_samples)
  cat("Y =", y, "Estimated Expected Loss:", mean(losses), "\n")
}
```

```
## Y = 0 Estimated Expected Loss: 0.1407221
## Y = 1 Estimated Expected Loss: 0.1384349
## Y = 2 Estimated Expected Loss: 0.3305116
## Y = 3 Estimated Expected Loss: 0.7552696
## Y = 4 Estimated Expected Loss: 1.348994
```

```r
set.seed(123)
N <- 5000
burn_in <- 500
w <- 0.3
y <- 2
a_curr <- 0.5
a_trace <- numeric(N)
theta_samples <- rbeta(N, y + 1, n - y + 1)

loss <- function(a, theta) {
  ifelse(a > theta, (a - theta)^2, 4 * (theta - a))
}

loss_curr <- mean(loss(a_curr, theta_samples))

for (i in 1:N) {
  a_prop <- runif(1, max(0, a_curr - w), min(1, a_curr + w))
```

```r
  loss_prop <- mean(loss(a_prop, theta_samples))

  alpha <- min(1, loss_curr / loss_prop)

  if (runif(1) < alpha) {
    a_curr <- a_prop
    loss_curr <- loss_prop
  }

  a_trace[i] <- a_curr
}

plot(a_trace, type = "l", main = "Trace Plot of a", ylab = "a", xlab = "N")
```




```r
cat("Estimated optimal a:", mean(a_trace[(burn_in + 1):N]), "\n")
```

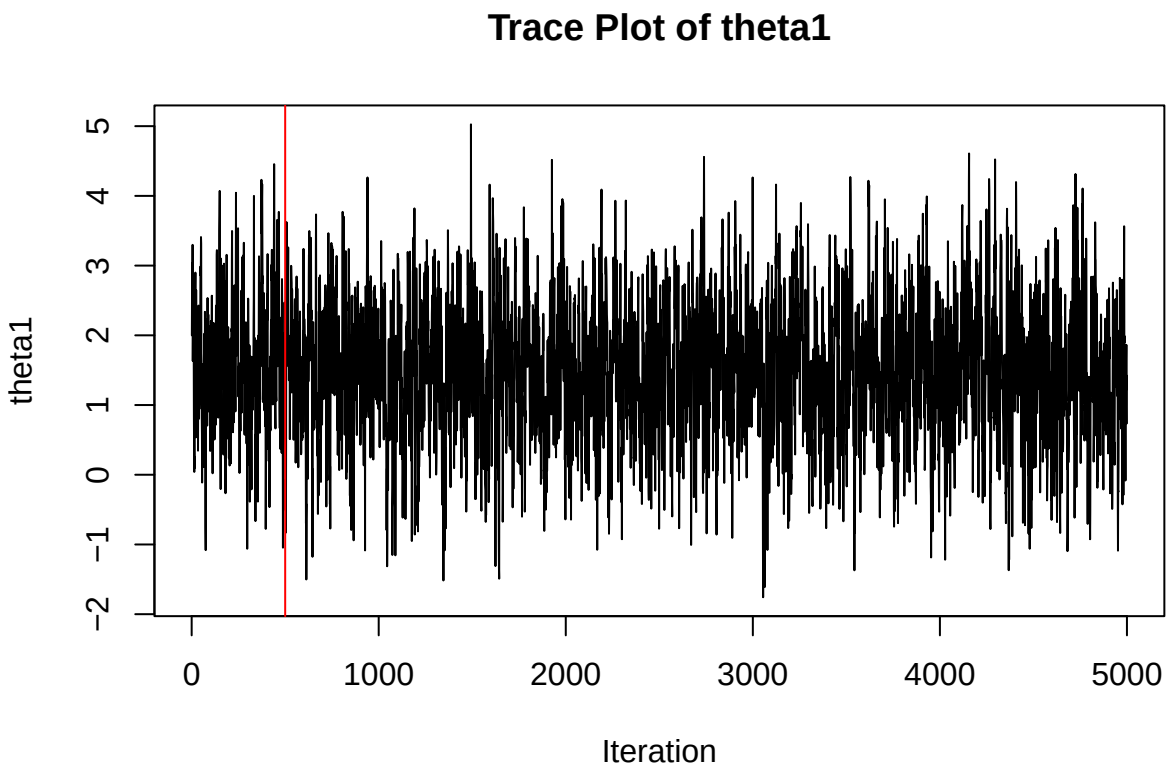
```
## Estimated optimal a: 0.6505666
```

```r
cat("Expected loss:", mean(loss(mean(tail(a_trace, N-burn_in)),theta_samples)), "\n")
```

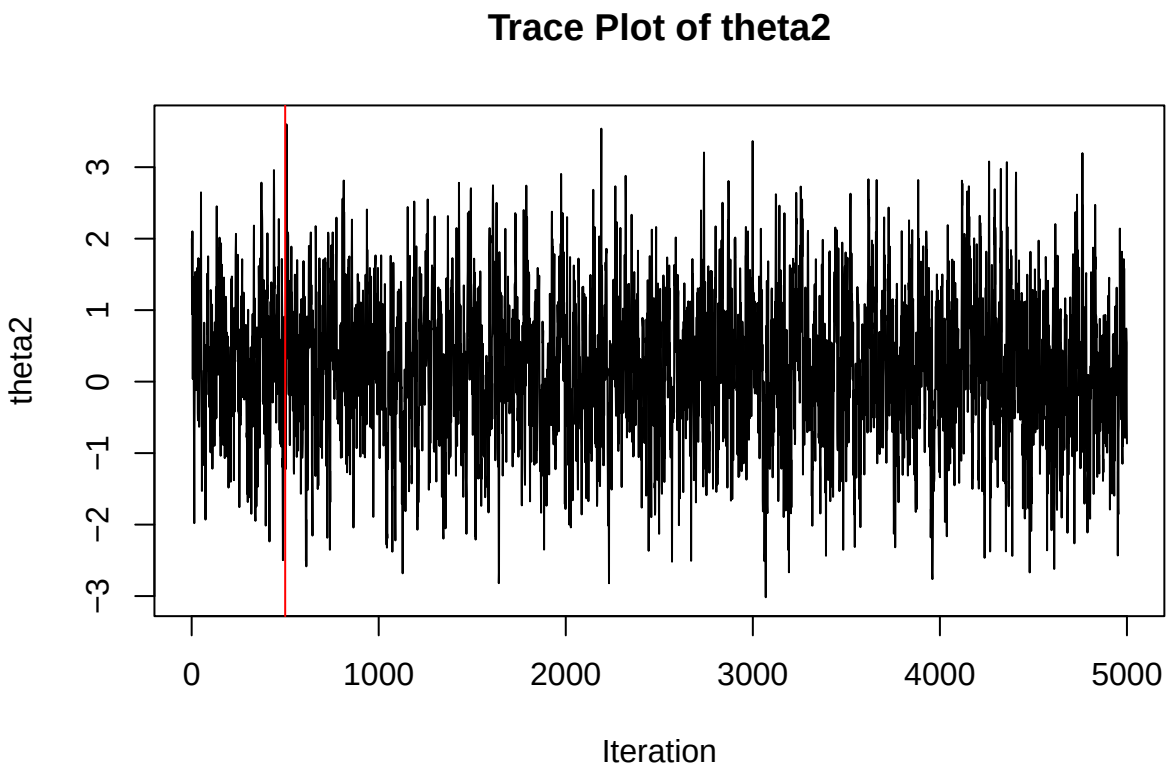
```
## Expected loss: 0.147408
```

Problem 3

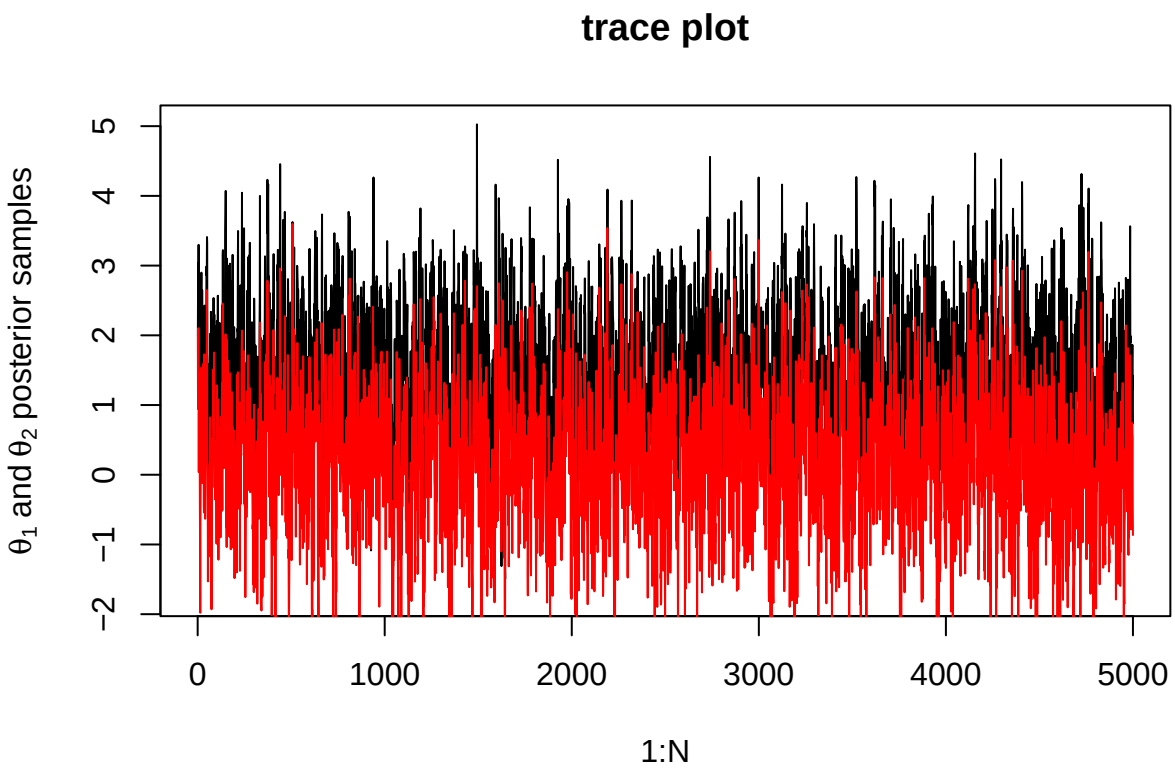
```r
set.seed(123)
N <- 5000
burn_in <- 500

# Initialize
theta1 <- numeric(N)
theta2 <- numeric(N)
theta1[1] <- 2
theta2[1] <- 2

y1 <- 1.5
y2 <- 0.2
rho <- 0.8
var_cond <- 1 - rho^2

for (i in 2:N) {
  mu1 <- y1 + rho * (theta2[i - 1] - y2)
  theta1[i] <- rnorm(1, mean = mu1, sd = sqrt(var_cond))

  mu2 <- y2 + rho * (theta1[i] - y1)
  theta2[i] <- rnorm(1, mean = mu2, sd = sqrt(var_cond))
}

plot(theta1, type = "l", main = "Trace Plot of theta1", ylab = "theta1", xlab = "Iteration")
abline(v = burn_in, col = "red")
```



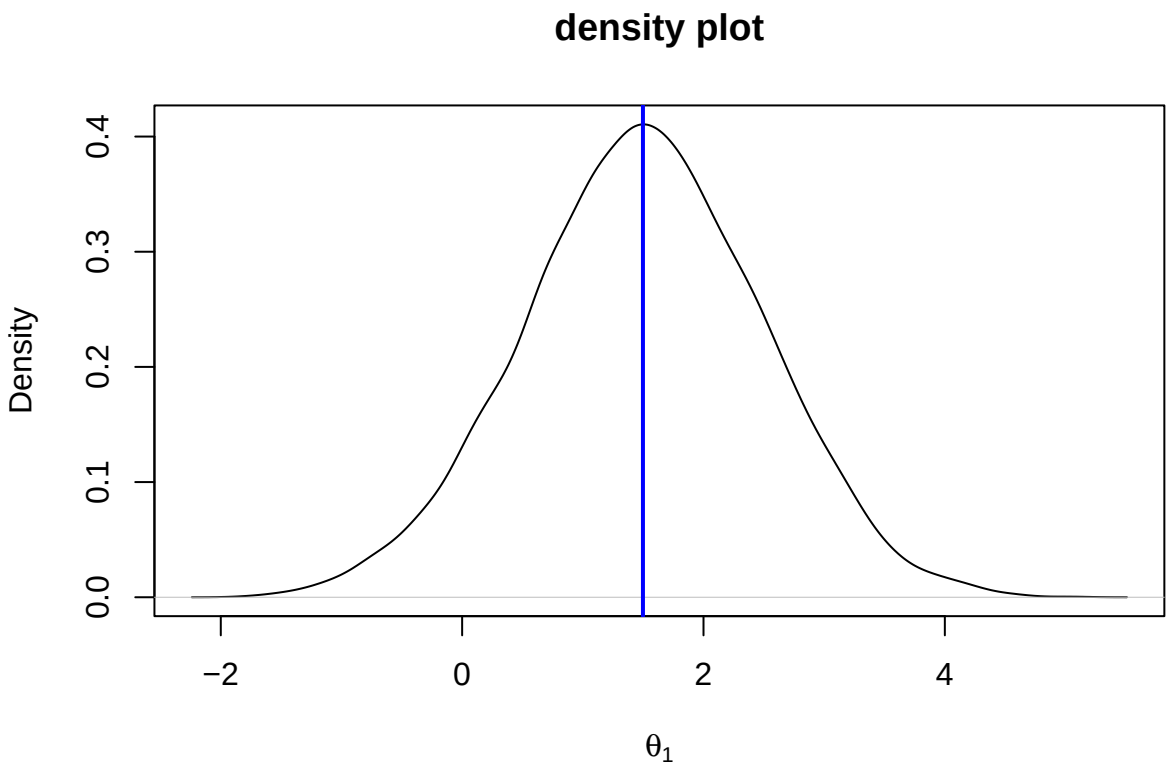

```r
plot(theta2, type = "l", main = "Trace Plot of theta2", ylab = "theta2", xlab = "Iteration")
abline(v = burn_in, col = "red")
```



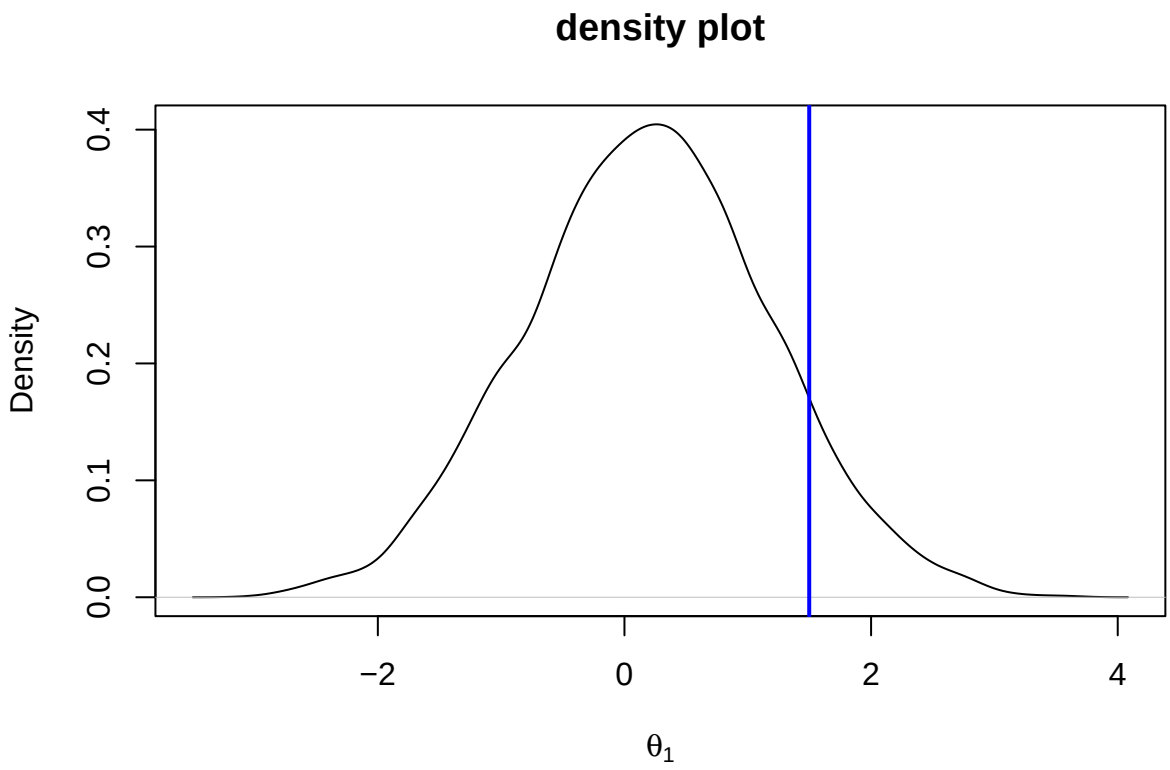

```
plot(theta1, type = "l", col = "black", ylab = expression(theta[1]~"and"~theta[2]~"posterior samples"),
     main = "trace plot", xlab = "1:N")
lines(theta2, col = "red")
```



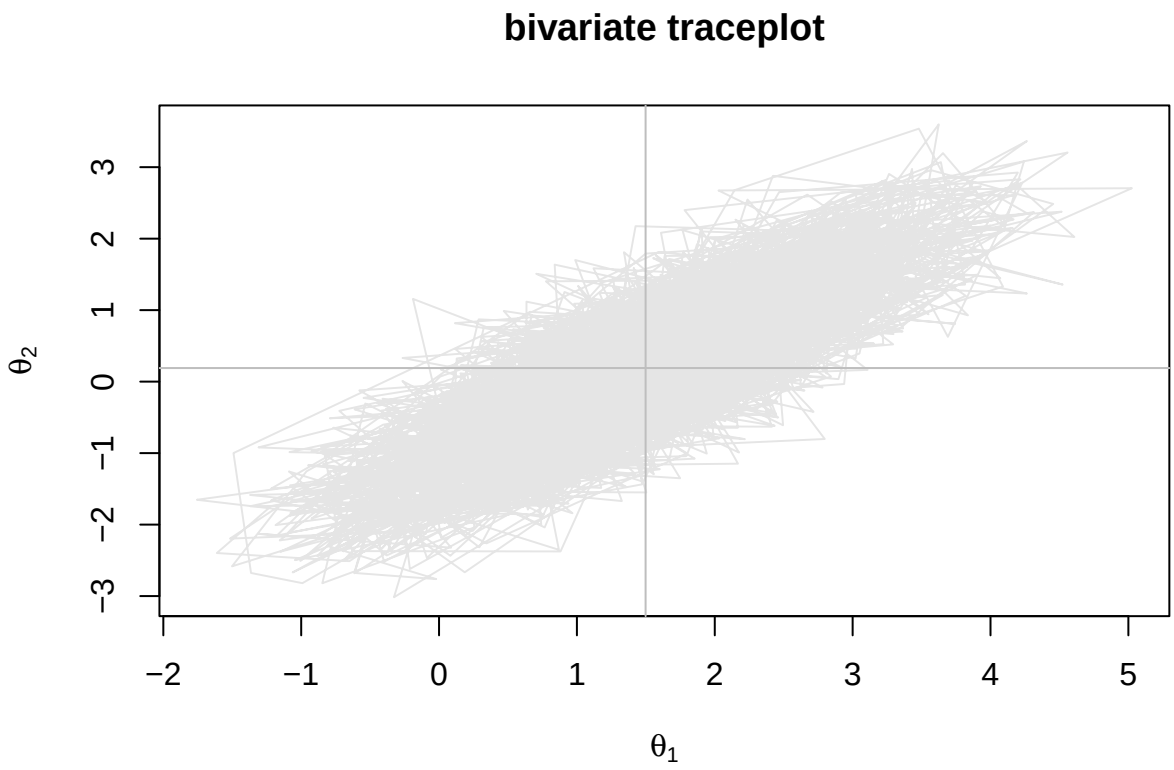

```
plot(density(theta1), main = "density plot", xlab = expression(theta[1]))
abline(v = mean(theta1), col = "blue", lwd = 2)
```




```
plot(density(theta2), main = "density plot", xlab = expression(theta[1]))
abline(v = mean(theta1), col = "blue", lwd = 2)
```




```
plot(theta1, theta2, type = "l",col = rgb(0, 0, 0, alpha = 0.1), main = "bivariate traceplot",
     xlab = expression(theta[1]), ylab = expression(theta[2]))
abline(h = mean(theta2), v = mean(theta1), col = "gray")
```




```
cat("Posterior Mean (r):", mean(theta1),"\n")
```

```
## Posterior Mean (r): 1.498294
```

```
cat("Posterior Mean (h):", mean(theta2),"\n")
```

```
## Posterior Mean (h): 0.189967
```

Problem 4

b

```
x1 <- seq(0, 10, by = 0.01)
x2 <- 10-x1
p1 <- 0.7
p2 <- 0.6

sharp_ratio <- function(x1,x2) {
  r1 <- x1*p1 - x1*(1-p1)
  r2 <- x2*p2 - x2*(1-p2)
  var1 <- p1*(x1-r1)^2 + (1-p1)*(-x1-r1)^2
```

```
  var2 <- p2*(x2-r2)^2 + (1-p2)*(-x2-r2)^2
  s_r <- (r1+r2)/sqrt(var1+var2)
  return(s_r)
}

s_r <- sharp_ratio(x1, x2)
x1[which.max(s_r)]
```

```
## [1] 6.96
```

```
x2[which.max(s_r)]
```

```
## [1] 3.04
```

```
max(s_r)
```

```
## [1] 0.481812
```