Part 1

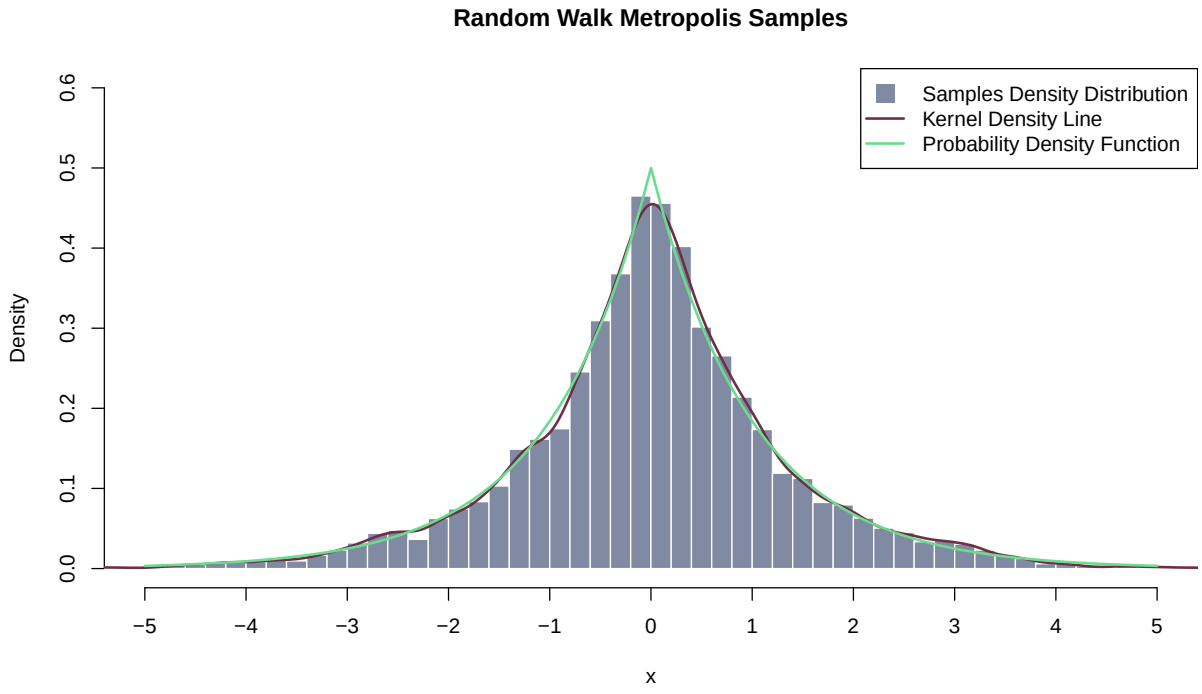
2025-03-12

```
library(ggplot2)
library(rmarkdown)
library(tinytex)
```

Exercise A

```
#parameters
x0 <- 0
N <- 10000
s <- 1

#pdf given
pdf <- function(x) {
  return(0.5 * exp(-abs(x)))
}

#random walk metropolis given
metropolis <- function(x0, N, s) {
  x <- numeric(N)
  x[1] <- x0

  for (i in 2:N) {
    random_number <- rnorm(1, mean = x[i-1], sd = s)
    r <- pdf(random_number) / pdf(x[i-1])

    if (log(runif(1)) < log(r)) {
      x[i] <- random_number
    } else {
      x[i] <- x[i-1]
    }
  }

  return(x)
}

#get samples and kernel density
samples <- metropolis(x0, N, s)
kernel_density <- density(samples)

#Plots
hist(samples,
     probability = TRUE,
```

```
     breaks = 50,
     main = "Random Walk Metropolis Samples",
     xlab = "x",
     col = "#818aa3",
     border = "white",
     xlim = (c(-5, 5)),
     ylim = (c(0.0, 0.6)),
     xaxt = "n")
axis(1, at = seq(-5, 5, by = 1))
lines(kernel_density, col = "#6d2e46", lwd = 2)
curve(pdf(x), col = "#6adc8e", lwd = 2, add = TRUE)
legend(x="topright",
       c("Samples Density Distribution", "Kernel Density Line", "Probability Density Function"),
       col=c("#818aa3","#6d2e46", "#6adc8e"),
       lty=c(NA,1,1), lwd = c(NA, 2, 2),
       pch = c(15,NA,NA), pt.cex = 2)
```




```
#Monte Carlo estimates
mean_estimate <- mean(samples)
sd_estimate <- sd(samples)

cat("Monte Carlo Mean Estimate:", mean_estimate)
```

```
## Monte Carlo Mean Estimate: 0.01436874
```

```
cat("Monte Carlo Standard Deviation Estimate:", sd_estimate)
```

```
## Monte Carlo Standard Deviation Estimate: 1.33171
```

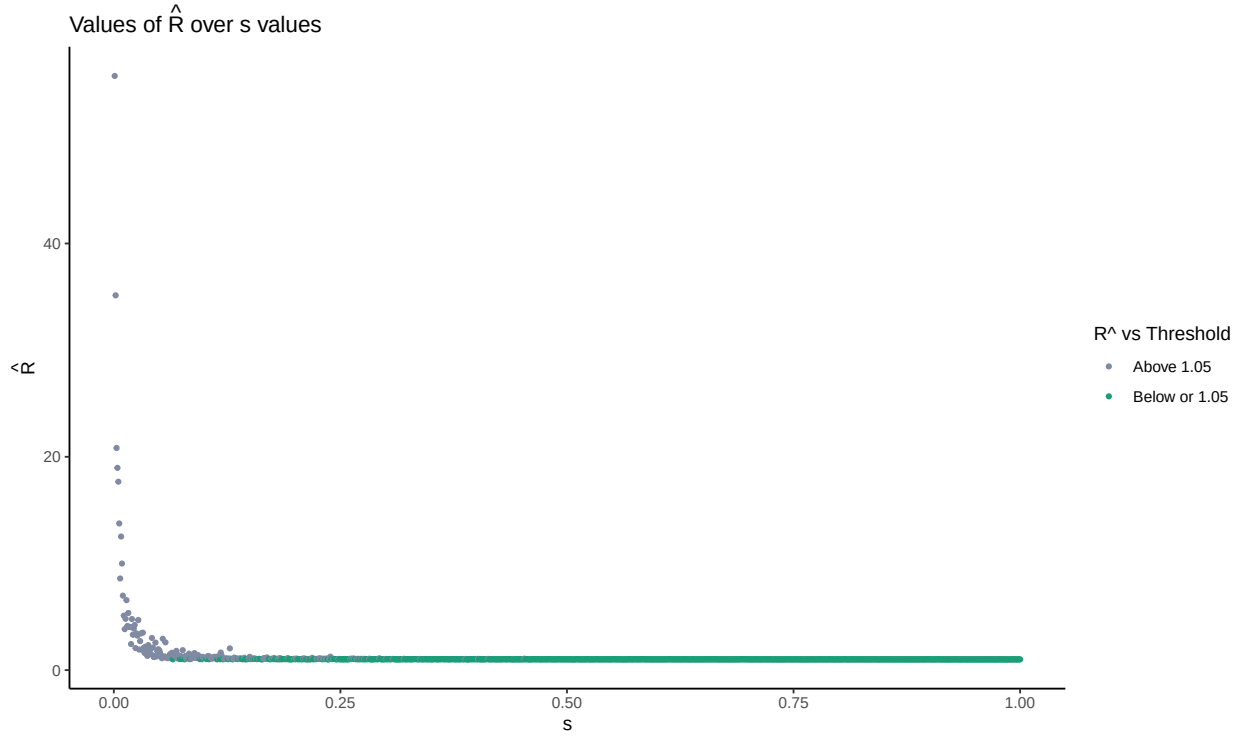
Exercise B

```r
#function to calculate Rhat
calculate_R_hat <- function(chains) {
  J <- length(chains)
  N <- length(chains[[1]])

  means <- sapply(chains, mean)
  variances <- sapply(chains, var)

  W <- mean(variances)
  M <- mean(means)
  B <- mean((means - M)^2)

  R_hat <- sqrt((B + W) / W)
  return(R_hat)
}

#values of N and s given
N <- 2000
s <- 0.001
x0_initial_values <- c(0, 1, 2, 3) #considered 4 chains with different x0 initial values

chains <- lapply(x0_initial_values, function(x0) metropolis(x0, N, s))
R_hat <- calculate_R_hat(chains)
cat("R-hat for s = 0.001:", R_hat)
```

```
## R-hat for s = 0.001: 78.32588
```

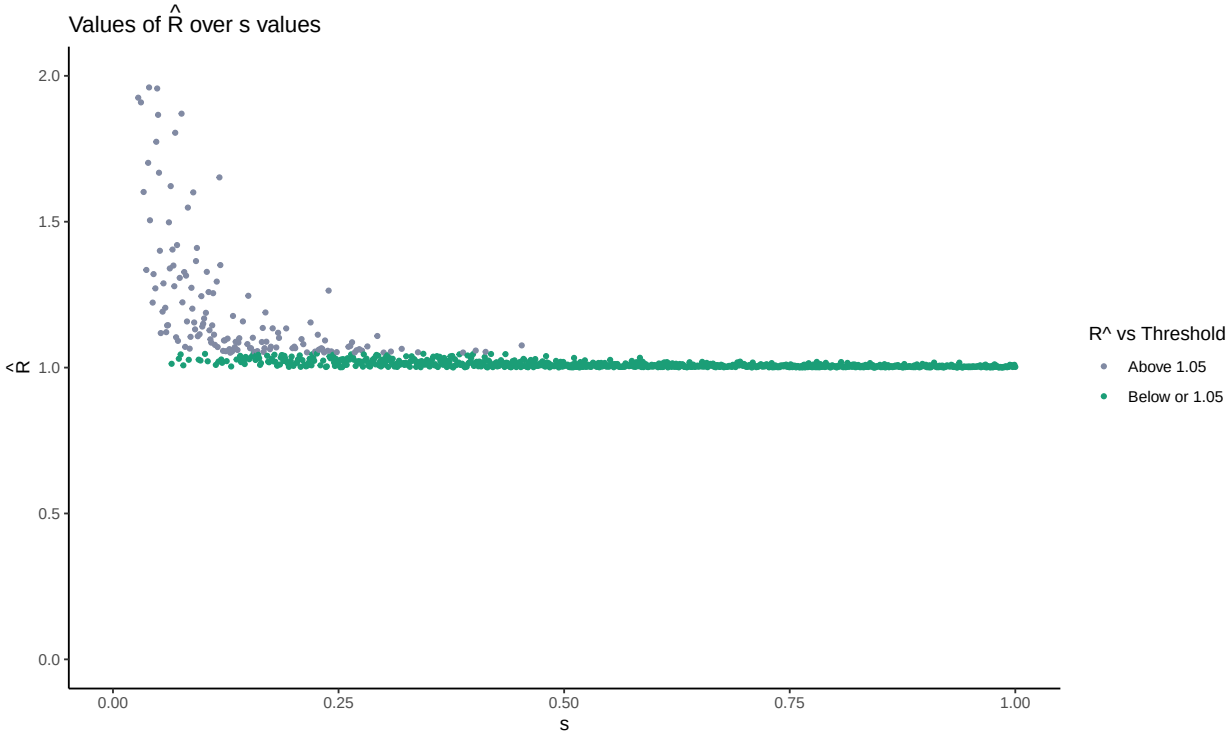
```r
#create plot for values of Rhat over grid of s values
s_values <- seq(0.001, 1, by = 0.001)
R_hats <- numeric(length(s_values))

for (i in seq_along(s_values)) {
  s <- s_values[i]
  chains <- lapply(x0_initial_values, function(x0) metropolis(x0, N, s))
  R_hats[i] <- calculate_R_hat(chains)
}

r_hat_df <- data.frame(
  s = s_values,
  R_hat = R_hats
)
r_hat_df$threshold_group <- ifelse(r_hat_df$R_hat > 1.05, "Above 1.05", "Below or 1.05")

ggplot(r_hat_df, aes(x = s, y = R_hat, color = threshold_group)) +
  geom_point(size = 1) +
  labs(
    title = expression(paste("Values of ", hat(R), " over s values")),
    x = "s",
    y = expression(hat(R)),
    color = "R vs Threshold"
```

```
  ) +
  scale_color_manual(
    values = c("Above 1.05" = "#818aa3", "Below or 1.05" = "#1b9e77")
  ) +
  theme(
      panel.grid.major = element_blank(),
      panel.grid.minor = element_blank(),
      panel.background = element_rect(fill = "white"),
      axis.line = element_line(color = "black")
    )
```




```
#to see better
ggplot(r_hat_df, aes(x = s, y = R_hat, color = threshold_group)) +
  geom_point(size = 1) +
  labs(
    title = expression(paste("Values of ", hat(R), " over s values")),
    x = "s",
    y = expression(hat(R)),
    color = "R vs Threshold"
  ) +
  ylim(0, 2) +
  scale_color_manual(
    values = c("Above 1.05" = "#818aa3", "Below or 1.05" = "#1b9e77")
  ) +
  theme(
      panel.grid.major = element_blank(),
      panel.grid.minor = element_blank(),
      panel.background = element_rect(fill = "white"),
      axis.line = element_line(color = "black")
```

```
)
```


Values of R̂ over s values
2.0
1.5
1.0
0.5
0.0
<R̂
0.00
0.25
0.50
0.75
1.00
s
R^ vs Threshold
Above 1.05
Below or 1.05