Homework 1

Alan Wu

imports

```
library(ggplot2)
library(tidyr)
```

Question 2: Interpreting the Posterior

We need to simulate a beta-binomial model with different parameters based on the data. The data shows that there are 3 out of 4 successful trials. We need to adjust and plot for the following alpha and beta values:

- (alpha, beta) = (0,0)
- (alpha, beta) = (1,1)
- (alpha, beta) = (1/2, 1/2)
- (alpha, beta) = (1, 2)
- (alpha, beta) = (2, 1)

We use 1,2 for the values where alpha > beta or when beta > alpha.

```
x <- seq(0, 1, length.out = 500)
priors <- list(
  "(0,0)" = c(0,0),
  "(1,1)" = c(1,1),
  "(1/2,1/2)" = c(0.5, 0.5),
  "(1,2)" = c(1,2),
  "(2,1)" = c(2,1)
)
s_low <- 3
f_low <- 1
s_hi <- 75
f_hi <- 25
```

```
# for the 3 successes out of 4 trials
posterior_3_4 = data.frame()

for (name in names(priors)) {
  p <- priors[[name]]
  alpha_prior <- p[1]
  beta_prior  <- p[2]

  alpha_post <- alpha_prior + s_low
  beta_post  <- beta_prior + f_low

  y <- dbeta(x, alpha_post, beta_post)

  temp_df <- data.frame(
    prob_val = x,
```

```
    density = y,
    prior_type = name
  )


  posterior_3_4 <- rbind(posterior_3_4, temp_df)
}
```

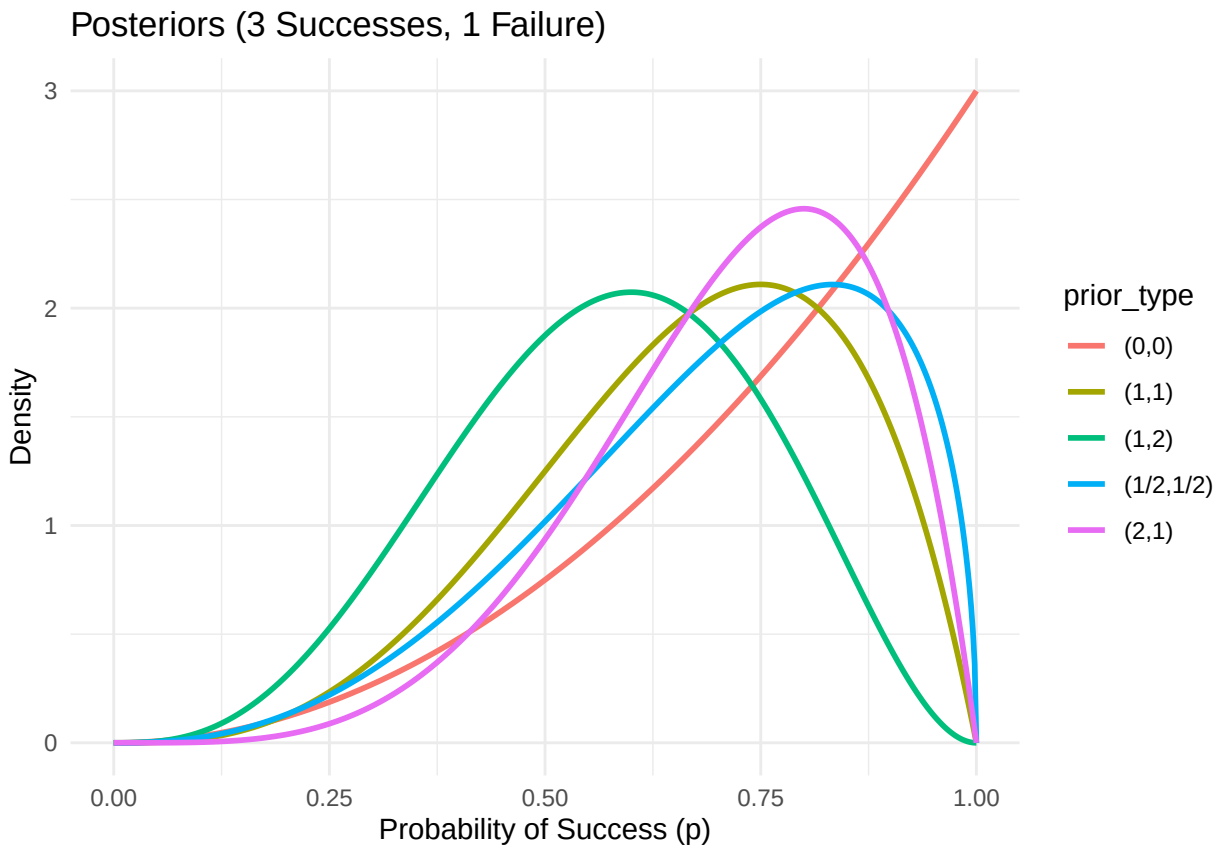
Function to help us define the full process better:

```
generate_posterior_df_2 <- function(s,f){
  x <- seq(0, 1, length.out = 500)
  priors <- list(
    "(0,0)" = c(0,0),
    "(1,1)" = c(1,1),
    "(1/2,1/2)" = c(0.5, 0.5),
    "(1,2)" = c(1,2),
    "(2,1)" = c(2,1)
  )

  posteriors = data.frame()

  for (name in names(priors)) {
    p <- priors[[name]]
    alpha_prior <- p[1]
    beta_prior  <- p[2]

    alpha_post <- alpha_prior + s
    beta_post  <- beta_prior + f


    y <- dbeta(x, alpha_post, beta_post)

    temp_df <- data.frame(
      prob_val = x,
      density = y,
      prior_type = name
    )

    posteriors <- rbind(posteriors, temp_df)


  }

  return(posteriors)
}
```

Plot results for 3 successes out of 4

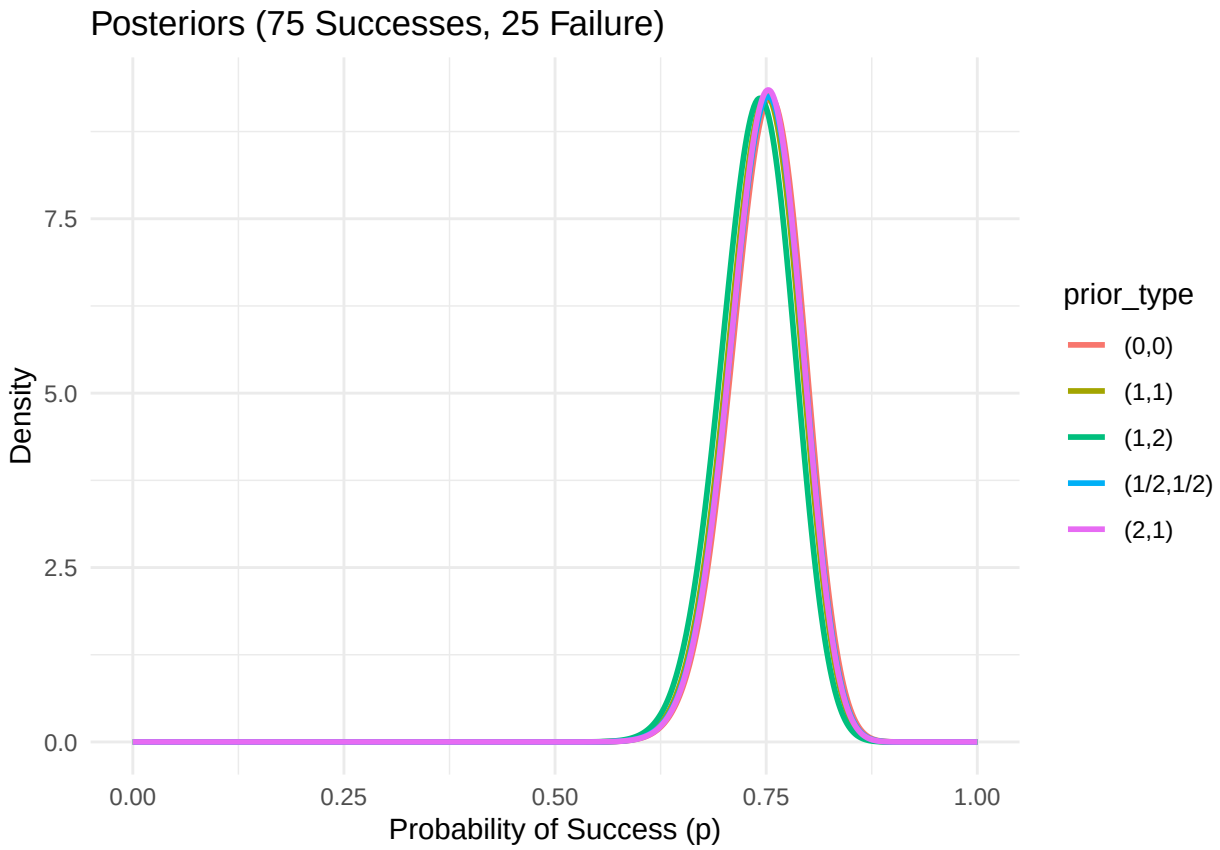
```
posteriors_3_4 <- generate_posterior_df_2(3,1)
ggplot(posteriors_3_4, aes(x = prob_val, y = density, color = prior_type)) +
  geom_line(linewidth = 1) +
  labs(title = "Posteriors (3 Successes, 1 Failure)",
```

```
       x = "Probability of Success (p)",
       y = "Density") +
  theme_minimal()
```



Plot results for 75 successes out of 100

```
posteriors_75_100 <- generate_posterior_df_2(75, 25)
ggplot(posteriors_75_100, aes(x = prob_val, y = density, color = prior_type)) +
  geom_line(linewidth = 1) +
  labs(title = "Posteriors (75 Successes, 25 Failure)",
       x = "Probability of Success (p)",
       y = "Density") +
  theme_minimal()
```



The overall trend that we realize is that when we introduce greater data, the posterior distribution tends to form to the parameters of the actual data. For the situation with 3 successes and 1 failure, at most we have a parameter of 2, either alpha or beta. Therefore, the prior has greater influence on the shape of the posterior distribution. However, when we increase the number of samples, despite the proportion of the number of successes being the same (75%), the posterior distribution shifts in favor of the data.

Question 4: Posterior Distribution on Real Data

Developing a model on the manchester data: want to predict the number of goals that the club would be expected to score in a future match.

The primary model we can use to model this is the gamma-poisson conjugate model. This is because we are dealing with count data. We will focus on the distribution of the team goals to find an expected number of goals.

Since we do not really have a good idea of the number of goals that manchester city would score per game, then we will use a relatively non-informative prior for the gamma distribution.

Based on the gamma-poisson conjugacy, we know that the posterior distribution will take on a gamma distribution with alpha = alpha + sum(y) and beta = beta + n, where n is the sample size.

```
manchester <- read.csv('./data/manchester_city_2024.csv')
head(manchester)
```

```
##   competition_name            team             opp team_goals opp_goals
## 1   Premier League Manchester City         Burnley          3         0
## 2   Premier League Manchester City   Newcastle Utd          1         0
```

```
## 3   Premier League Manchester City   Sheffield Utd          2         1
## 4   Premier League Manchester City          Fulham          5         1
## 5   Premier League Manchester City        West Ham          3         1
## 6   Premier League Manchester City Nott'ham Forest          2         0
##   attendance          venue        referee
## 1      21572      Turf Moor   Craig Pawson
## 2      53419 Etihad Stadium   Robert Jones
## 3      31336   Bramall Lane Jarred Gillett
## 4      52899 Etihad Stadium Michael Oliver
## 5      62475 London Stadium    Andy Madley
## 6      53413 Etihad Stadium Anthony Taylor
```

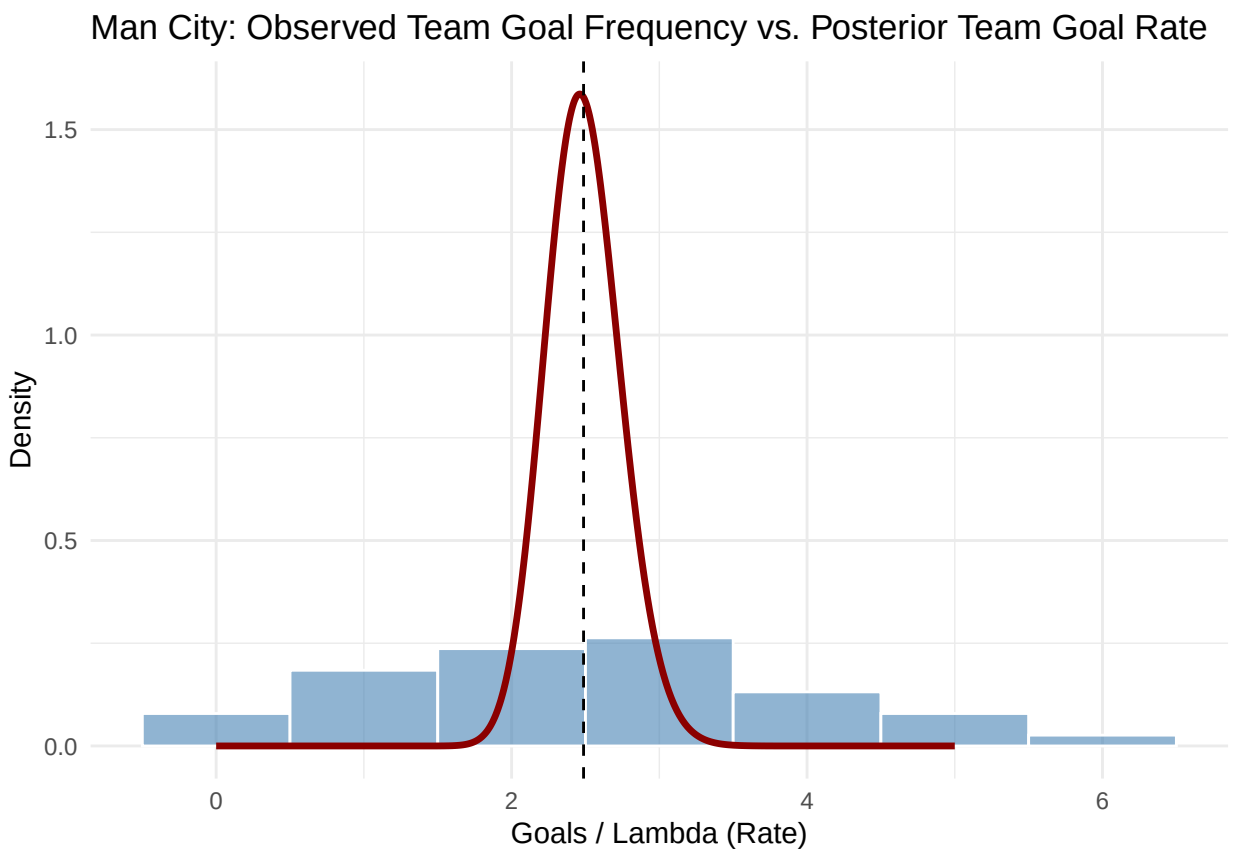
Computing the posterior values

```
alpha <- 1
beta <- 1
n <- nrow(manchester)
sum_y = sum(manchester$team_goals)

alpha_post_team <- alpha + sum_y
beta_post_team <- beta + n

lambdas <- seq(0, 5, length.out = 1000)
density_vals <- dgamma(lambdas, shape = alpha_post_team, rate = beta_post_team)
posterior_curve_team <- data.frame(
  lambda = lambdas,
  density = density_vals
)
```

Plotting the posterior distribution vs the actual data

```
ggplot() +
  # The Histogram: y = after_stat(density) is key to align scales
  geom_histogram(data = manchester, aes(x = team_goals, y = after_stat(density)),
                 binwidth = 1, fill = "steelblue", color = "white", alpha = 0.6) +

  # The Posterior Line: Using our manual data frame
  geom_line(data = posterior_curve_team, aes(x = lambda, y = density),
            color = "darkred", size = 1.2) +

  # Adding a vertical line for the mean to show where it centers
  geom_vline(xintercept = alpha_post_team/beta_post_team, linetype = "dashed", color = "black") +

  labs(title = "Man City: Observed Team Goal Frequency vs. Posterior Team Goal Rate",
       x = "Goals / Lambda (Rate)",
       y = "Density") +
  theme_minimal()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




```
cat(sprintf("Observed Mean:     %.2f\n", mean(manchester$team_goals)))
```

```
## Observed Mean:     2.53
```

```
cat(sprintf("Observed Variance: %.2f\n", var(manchester$team_goals)))
```

```
## Observed Variance: 2.20
```

```
cat(sprintf("Posterior Mean: %.2f\n", alpha_post_team / beta_post_team))
```

```
## Posterior Mean: 2.49
```

```
cat(sprintf("Posterior variance: %.2f\n", alpha_post_team / (beta_post_team*beta_post_team)))
```

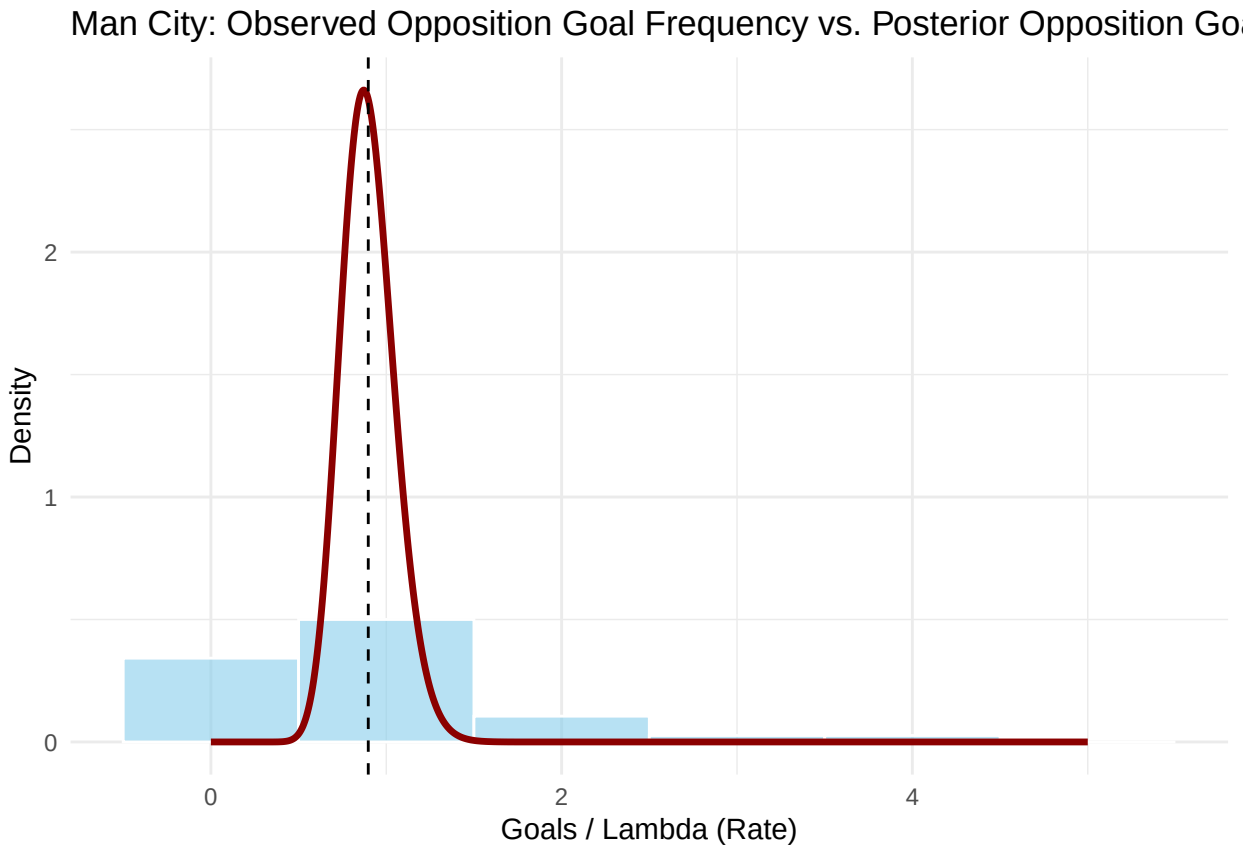
```
## Posterior variance: 0.06
```

Question 5: Posterior Distribution on Real Data

For this part, we can use the same method as before, with gamma-poisson conjugacy. To define the prior, we will use alpha = 1 and beta = 1, since we have no prior knowledge regarding the outcomes of opposition goals scored against manchester. We have no clue on the defensive capabilities of manchester city and will let the data take over.

```r
alpha <- 1
beta <- 1
n <- nrow(manchester)
sum_y = sum(manchester$opp_goals)

alpha_post_opp <- alpha + sum_y
beta_post_opp <- beta + n

lambdas <- seq(0, 5, length.out = 1000)
density_vals <- dgamma(lambdas, shape = alpha_post_opp, rate = beta_post_opp)
posterior_curve_opp <- data.frame(
  lambda = lambdas,
  density = density_vals
)
```

```r
ggplot() +
  # The Histogram: y = after_stat(density) is key to align scales
  geom_histogram(data = manchester, aes(x = opp_goals, y = after_stat(density)),
                 binwidth = 1, fill = "skyblue", color = "white", alpha = 0.6) +

  # The Posterior Line: Using our manual data frame
  geom_line(data = posterior_curve_opp, aes(x = lambda, y = density),
            color = "darkred", size = 1.2) +

  geom_vline(xintercept = alpha_post_opp/beta_post_opp, linetype = "dashed", color = "black") +

  labs(title = "Man City: Observed Opposition Goal Frequency vs. Posterior Opposition Goal Rate",
       x = "Goals / Lambda (Rate)",
       y = "Density") +
  theme_minimal()
```




```
cat(sprintf("Observed Opp Goals Mean:     %.2f\n", mean(manchester$opp_goals)))
```

```
## Observed Opp Goals Mean:     0.89
```

```
cat(sprintf("Observed Opp Goals Variance: %.2f\n", var(manchester$opp_goals)))
```

```
## Observed Opp Goals Variance: 0.80
```

```
cat(sprintf("Posterior Opp Goals Mean: %.2f\n", alpha_post_opp / beta_post_opp))
```

```
## Posterior Opp Goals Mean: 0.90
```

```
cat(sprintf("Posterior Opp Goals variance: %.2f\n", alpha_post_opp / (beta_post_opp*beta_post_opp)))
```

```
## Posterior Opp Goals variance: 0.02
```

Question 6: Posterior Predictive

Our goal is to use 1000 simulations from the above parts to describe the number of games that manchester city will win. The process will be the following to simulate the posterior predictive distribution:

1. sample 1000 lambdas from the posterior distribution for team goals

2. sample 1000 lambdas from the posterior distribution for opposition goals
3. simulate the number of goals scored each game from poisson distribution for both team and opposition
4. compare these two and get the rate for wins, losses and ties for each sim.

```
set.seed(42)
posterior_team <- rgamma(1000, shape=alpha_post_team, rate=beta_post_team)
posterior_opp <- rgamma(1000, shape=alpha_post_opp, rate=beta_post_opp)

sim_team_goals <- rpois(1000, lambda = posterior_team)
sim_opp_goals  <- rpois(1000, lambda = posterior_opp)

results <- data.frame(
  team = sim_team_goals,
  opp = sim_opp_goals
)

results$outcome <- ifelse(results$team > results$opp, "Win",
                          ifelse(results$team == results$opp, "Tie", "Loss"))

prob_table <- table(results$outcome) / 1000 * 100.0
print(prob_table)
```

```
##
## Loss  Tie  Win
## 13.4 14.9 71.7
```

The results are in percentages.

Question 8: Preview of Gibbs Sampler

Consider a normal model with unknown mean and unknown variance. Using the following steps, describe the process of iteratively sampling from the posterior distribution:

1. Simulate data $y_1, y_2, \ldots, y_n \sim N(10, 3.3)$ where n = 100. Prior to simulation set your seed with set.seed(5). 2.Assign independent conjugate priors:

- $\mu \sim N(0, 50)$,
- $\sigma^2 \sim Inverse - Gamma(2, 2)$

3. Iteratively sample from the posterior 10,000 times where you use your most recent value of μ and σ^2 sampled in the conditional posterior distributions of the other parameter:

- $\mu|\sigma^2, y$
- $\sigma^2|\mu, y$

Report a 95% posterior interval for both μ and σ^2 based on your samples of each. Are your samples of μ and σ^2 correlated with each other?

We choose to use the prior means given to use to use as the initialization of the parameters. We define θ_1 as the mean mu and the θ_2 as the variance of the parameter mu.

We start by developing two vectors of size and start with our initializations.

```r
set.seed(5)

iter <- 10000

y <- rnorm(100, 10, 3.3)
s2_mu <- 50
m_mu <- 0
n <- length(y)
y_bar <- mean(y)
alpha <- 2
beta <- 2

theta1 <- rep(NA,iter)
theta2 <- rep(NA,iter)

theta1[1] <- 0
theta2[1] <- (beta)/(alpha-1)
```

Then, for the gibbs sampler, we sample each next step with respect to the other parameter conditiional distribution. This means:

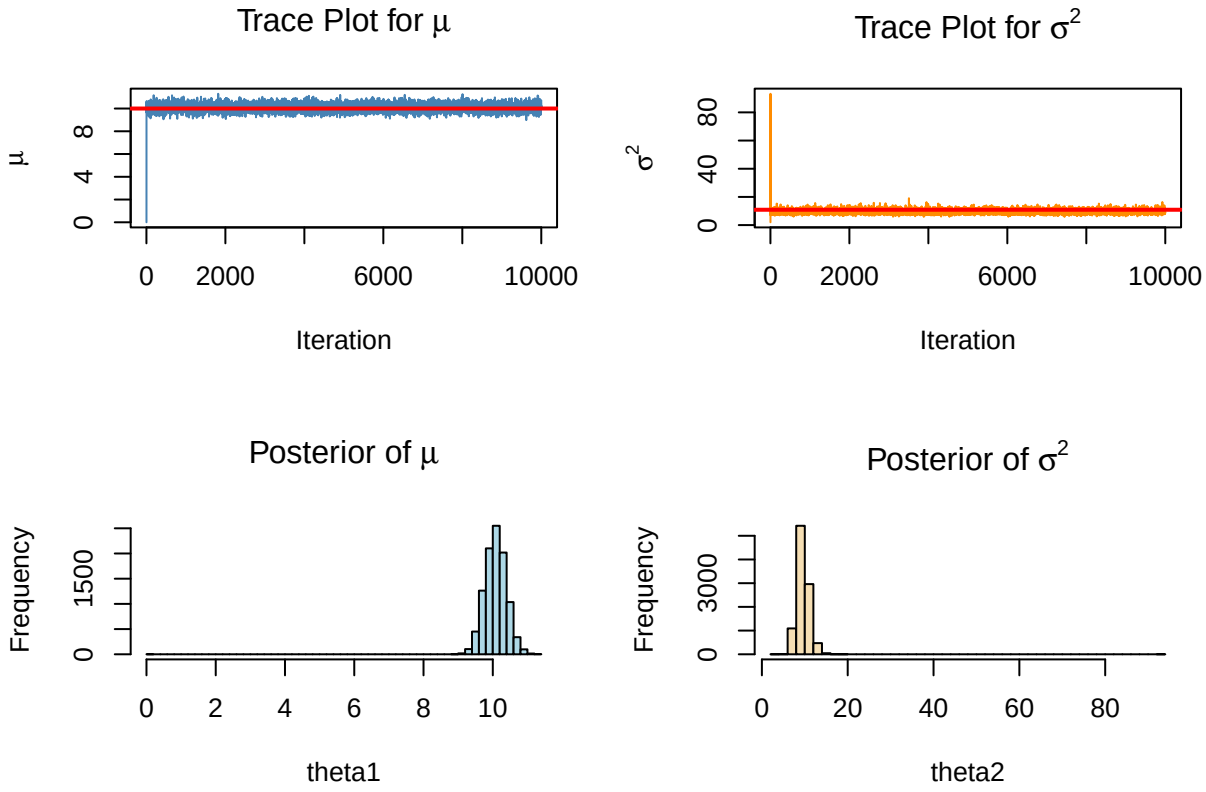
- when we sample for mu, we condition on the current sampled value of σ^2 (e.g., $\sigma^2 = 12.5$ from the previous iteration)
- when we sample for mu_s2, we condition on the current sampled value of μ (e.g., $\mu = 9.8$ that we just sampled in this iteration)

```r
for (i in 2:iter){

  curr_sigma2 <- theta2[i-1]
  curr_mu <- theta1[i-1]

  post_mu <- ((1/s2_mu) / (n/curr_sigma2 + 1/s2_mu))*m_mu +
  ((n/curr_sigma2)/(n/curr_sigma2 + 1/s2_mu))*y_bar
  post_s2 <- 1/((1/s2_mu) + (n/curr_sigma2))

  alpha_post <- alpha + n/2
  beta_post <- beta + 0.5 * sum((y - curr_mu)^2)

  #first sample mu
  theta1[i] <- rnorm(1, mean=post_mu, sd=sqrt(post_s2))

  #then sample sigma2
  theta2[i] <- 1/rgamma(1, shape=alpha_post, rate=beta_post)
}
```

MCMC Convergence Evaluation

```r
par(mfrow = c(2, 2)) # Arrange plots in a 2x2 grid

# Trace Plot for Mu
plot(theta1, type = "l", col = "steelblue",
     main = expression(paste("Trace Plot for ", mu)),
```

```
     xlab = "Iteration", ylab = expression(mu))
abline(h = 10, col = "red", lwd = 2)

# Trace Plot for Sigma^2
plot(theta2, type = "l", col = "darkorange",
     main = expression(paste("Trace Plot for ", sigma^2)),
     xlab = "Iteration", ylab = expression(sigma^2))
abline(h = 3.3^2, col = "red", lwd = 2)

# Histograms
hist(theta1, breaks = 50, col = "lightblue", main = expression(paste("Posterior of ", mu)))
hist(theta2, breaks = 50, col = "wheat", main = expression(paste("Posterior of ", sigma^2)))
```



Results

```
mu_interval <- quantile(theta1, probs = c(0.025, 0.975))
sigma2_interval <- quantile(theta2, probs = c(0.025, 0.975))
posterior_corr <- cor(theta1, theta2)

print(list(mu_95_interval = mu_interval,
           sigma2_95_interval = sigma2_interval,
           correlation = posterior_corr))
```

```
## $mu_95_interval
##      2.5%     97.5%
##  9.487491 10.696146
##
## $sigma2_95_interval
```

```
##      2.5%     97.5%
##  7.245949 12.573687
##
## $correlation
## [1] 0.01048906
```