Introduction to Multiparameter Bayesian Models

Rajitha M. Silva, PhD

Introduction to Multiparameter Models

This chapter focuses on Bayesian models with **multiple unknown parameters**. Unlike simpler Bayesian models where we estimate just one parameter (such as the mean of a normal distribution), multiparameter models involve estimating multiple parameters simultaneously.

A few key ideas covered in this chapter:

- **Simulation methods**: Instead of finding exact solutions, we generate samples from the posterior distribution.
- **Transformations**: Once we have simulated samples of the parameters, we can transform them to understand different functions of the parameters.
- **Grid approximation**: When the posterior distribution is complex (e.g., logistic regression), we approximate it using a fine grid of points.
- **Comparing proportions**: A Bayesian approach to comparing two proportions is useful in scenarios such as medical studies or election polling.

Normal Data with Both Parameters Unknown

A classical problem in Bayesian statistics is estimating both the **mean** (μ) and **variance** (σ^2) of a normal population when both are unknown.

Problem Setup

We assume a dataset from 20 runners who participated in the New York Marathon. Their completion times $y_1, y_2, \ldots, y_{20}$ in minutes are modeled as a random sample from a normal distribution:

$$Y_i \sim N(\mu, \sigma^2)$$

where:

- μ is the true mean completion time.
- σ^2 is the true variance of completion times.

Since we do not know μ and σ^2, we must estimate them using Bayesian methods.

Prior Distribution

A standard **noninformative prior** (also called an improper prior) is assumed:

$$g(\mu, \sigma^2) \propto \frac{1}{\sigma^2}$$

This prior expresses *no strong prior knowledge* about μ and σ^2, allowing the data to dominate the inference.

Posterior Distribution

Using **Bayes' Theorem**, we derive the posterior distribution for μ and σ^2:

Starting with Bayes' Theorem:

$$g(\mu, \sigma^2 | y) \propto g(y|\mu, \sigma^2) g(\mu, \sigma^2)$$

where: - $g(y|\mu, \sigma^2)$ is the likelihood function. - $g(\mu, \sigma^2)$ is the prior.

The likelihood function for a normal sample is:

$$g(y|\mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - \mu)^2\right)$$

Substituting the noninformative prior $g(\mu, \sigma^2) \propto \frac{1}{\sigma^2}$, we get:

$$g(\mu, \sigma^2 | y) \propto \frac{1}{(\sigma^2)^{n/2+1}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - \mu)^2\right)$$

To further simplify the exponent term, we use the identity:

$$\sum_{i=1}^{n}(y_i - \mu)^2 = \sum_{i=1}^{n}(y_i - \bar{y} + \bar{y} - \mu)^2$$

Expanding this expression:

$$\sum_{i=1}^{n}(y_i - \mu)^2 = \sum_{i=1}^{n}(y_i - \bar{y})^2 + 2(\bar{y} - \mu)\sum_{i=1}^{n}(y_i - \bar{y}) + n(\bar{y} - \mu)^2$$

Since $\sum_{i=1}^{n}(y_i - \bar{y}) = 0$, the middle term vanishes, simplifying to:

$$\sum_{i=1}^{n}(y_i - \mu)^2 = S + n(\mu - \bar{y})^2$$

Thus, the posterior distribution becomes:

$$g(\mu, \sigma^2 | y) \propto \frac{1}{(\sigma^2)^{n/2+1}} \exp\left(-\frac{1}{2\sigma^2}\left(S + n(\mu - \bar{y})^2\right)\right)$$

where:

- $n = 20$ (sample size),
- $\bar{y}$ is the sample mean (average marathon completion time),
- S is the sum of squared deviations from the mean:

$$S = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

This equation represents the **joint posterior distribution** of μ and σ^2.

Understanding the Posterior Distribution

Conditional Posterior of μ Given σ^2

We rewrite the joint posterior:

$$g(\mu, \sigma^2|y) \propto (\sigma^2)^{-n/2-1} \exp\left(-\frac{1}{2\sigma^2}\left(S + n(\mu - \bar{y})^2\right)\right)$$

Focusing on terms involving μ:

$$\exp\left(-\frac{n}{2\sigma^2}(\mu - \bar{y})^2\right)$$

which is recognized as the kernel of a normal distribution. Therefore, the conditional posterior of μ given σ^2 is:

$$\mu|\sigma^2, y \sim N\left(\bar{y}, \frac{\sigma}{\sqrt{n}}\right)$$

This means that, given the variance σ^2, the mean μ follows a normal distribution centered at $\bar{y}$ with standard deviation $\frac{\sigma}{\sqrt{n}}$.

Marginal Posterior of σ^2

Integrating out μ, we obtain:

$$\sigma^2|y \sim S \cdot \chi^{-2}_{n-1}$$

where χ^{-2}_{n-1} denotes an **inverse chi-square distribution** with $n-1$ degrees of freedom.

Summary of Bayesian Approach

1. **Define the prior**: Here, we use a noninformative prior $g(\mu, \sigma^2) \propto \frac{1}{\sigma^2}$.
2. **Compute the posterior**: We use Bayes' theorem to derive the posterior distributions of μ and σ^2.
3. **Characterize the posterior**:
   - The conditional posterior of μ follows a normal distribution.
   - The marginal posterior of σ^2 follows an inverse chi-square distribution.

Computing the Posterior in R

We now compute and visualize the posterior using the dataset from the book (`marathontimes`).

```
library(LearnBayes)
data(marathontimes)
attach(marathontimes)
time <- marathontimes$time
```

Simulating from the Posterior

```
n <- length(time)
y_bar <- mean(time)
S <- sum((time - y_bar)^2)
df <- n - 1
```

```
set.seed(123)
sigma2_samples <- S / rchisq(1000, df)
mu_samples <- rnorm(1000, mean = y_bar, sd = sqrt(sigma2_samples) / sqrt(n))
```

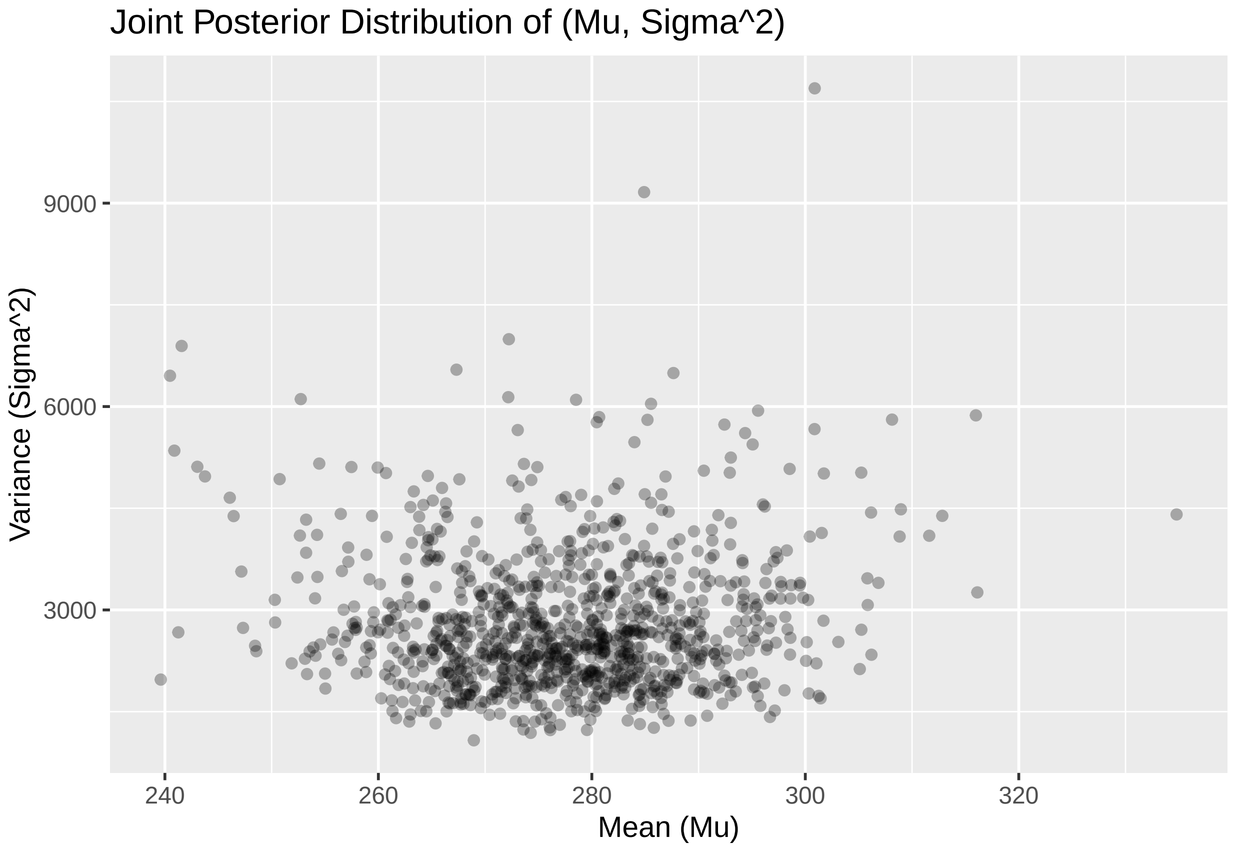
Compute 95% Credible Intervals

```
ci_mu <- quantile(mu_samples, c(0.025, 0.975))
ci_sigma <- quantile(sqrt(sigma2_samples), c(0.025, 0.975))

list(mean_ci = ci_mu, sigma_ci = ci_sigma)
```

```
## $mean_ci
##     2.5%    97.5%
## 253.9023 300.4377
##
## $sigma_ci
##     2.5%    97.5%
## 38.28686 71.84067
```

Plotting the Joint Posterior Distribution

```
library(ggplot2)
data_posterior <- data.frame(mu = mu_samples, sigma2 = sigma2_samples)

ggplot(data_posterior, aes(x = mu, y = sigma2)) +
  geom_point(alpha = 0.3) +
  labs(title = "Joint Posterior Distribution of (Mu, Sigma^2)",
       x = "Mean (Mu)", y = "Variance (Sigma^2)")
```



We now use the functions from the **LearnBayes** package.

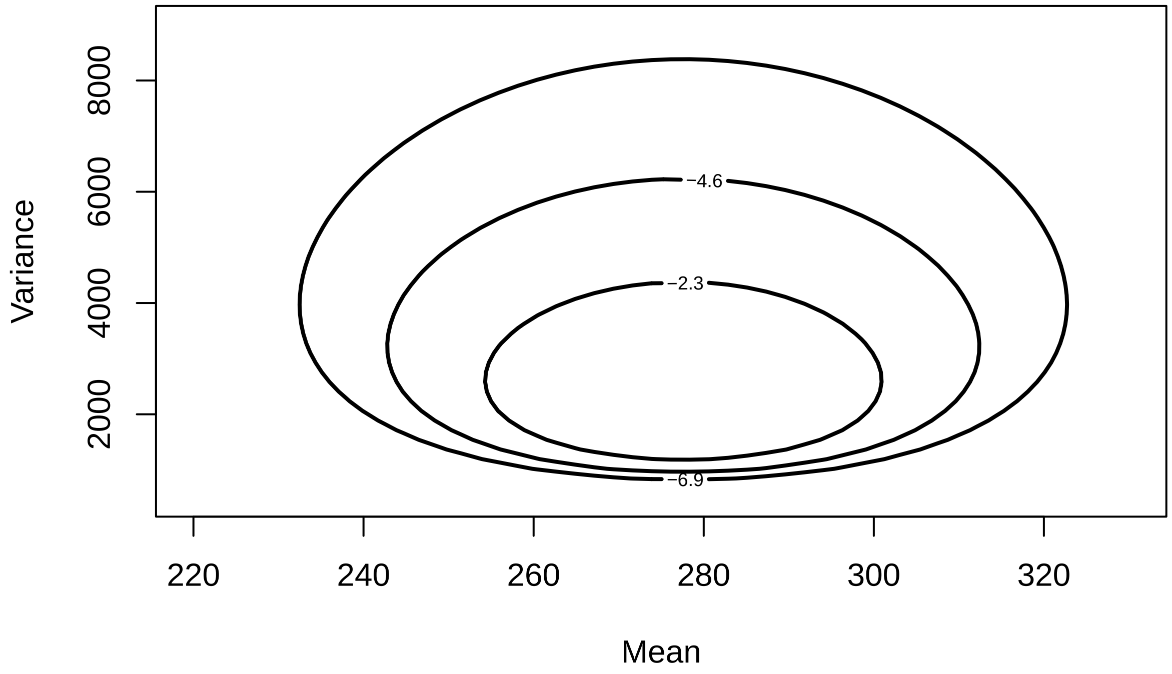
Load Data and Required Functions

```
library(LearnBayes)
data(marathontimes)
attach(marathontimes)
time <- marathontimes$time
```

Contour Plot of Joint Posterior

We use the `normchi2post` function to compute the log posterior density and the `mycontour` function to plot the posterior distribution.

```
library(LearnBayes)

# Define limits for the contour plot
x_limits <- c(220, 330, 500, 9000)

# Plot contour for posterior distribution
mycontour(normchi2post, x_limits, time, xlab="Mean", ylab="Variance")
```



Simulating from the Posterior using `normpostsim`

The `normpostsim` function simulates from the **joint posterior distribution**.

```
library(LearnBayes)

# Simulate from the joint posterior
draws <- normpostsim(time)

# Extract mu and sigma^2 samples
mu_samples <- draws$mu
sigma2_samples <- draws$sigma2
```

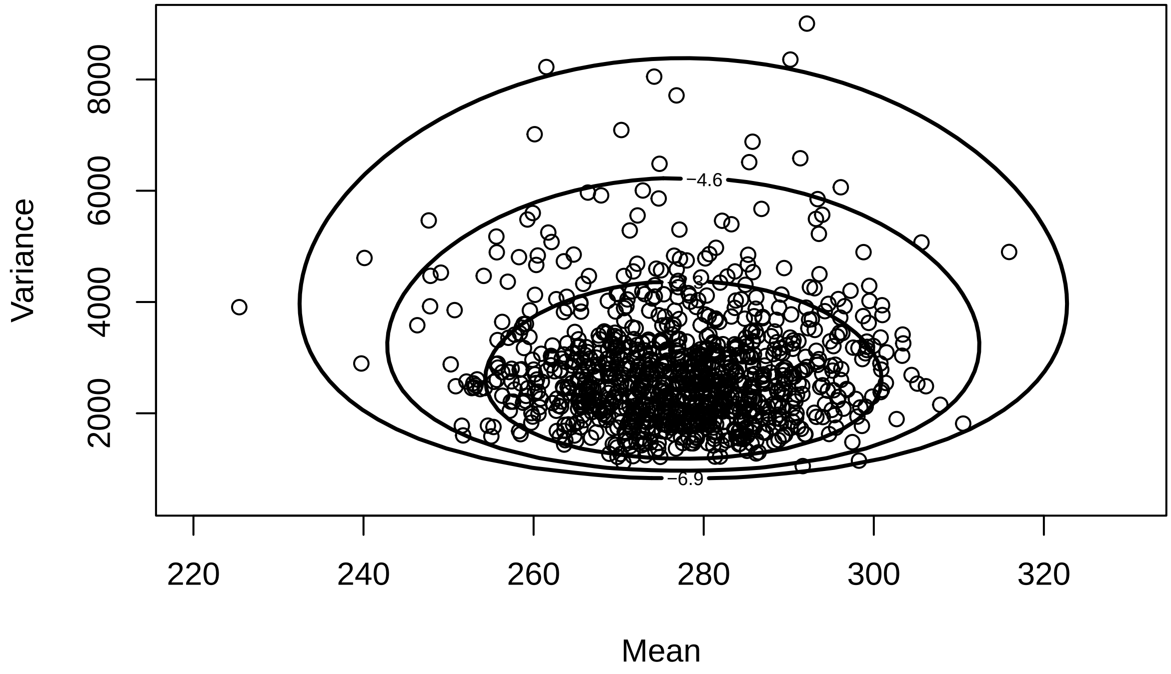
Compute 95% Credible Intervals

```
ci_mu <- quantile(mu_samples, c(0.025, 0.975))
ci_sigma <- quantile(sqrt(sigma2_samples), c(0.025, 0.975))

list(mean_ci = ci_mu, sigma_ci = ci_sigma)
```

```
## $mean_ci
##     2.5%    97.5%
## 255.3355 299.2160
##
## $sigma_ci
##     2.5%    97.5%
## 38.03738 73.89407
```

Plotting the Simulated Posterior Samples on Contour Plot

```r
mycontour(normchi2post, x_limits, time, xlab="Mean", ylab="Variance")
points(mu_samples, sigma2_samples)
```



Introduction to Bayesian Multinomial Models

In this section, we explore Bayesian inference for **multinomial models**, which generalize binomial models to more than two categories. The **multinomial distribution** models situations where we have multiple possible outcomes for a single trial, such as survey responses, dice rolls, or categorical data classification.

Definition of the Multinomial Model

A **multinomial model** is an extension of the binomial model, where each observation falls into one of k categories. Given n trials, we define:

- Let $Y = (Y_1, Y_2, ..., Y_k)$ represent the counts for each of the k categories.
- Each observation falls into category i with probability θ_i.
- The probabilities satisfy:

$$\sum_{i=1}^{k} \theta_i = 1$$

- The likelihood function for a multinomial model is:

$$P(Y_1, Y_2, ..., Y_k | \theta_1, \theta_2, ..., \theta_k) = \frac{n!}{Y_1!Y_2!...Y_k!} \theta_1^{Y_1} \theta_2^{Y_2} ... \theta_k^{Y_k}$$

  where Y_i is the observed count for category i, and $n = \sum_{i=1}^{k} Y_i$ is the total number of observations.

Choosing a Prior Distribution

A natural conjugate prior for the multinomial model is the **Dirichlet distribution**, which is the multivariate generalization of the beta distribution:

$$p(\theta_1, ..., \theta_k) \propto \theta_1^{\alpha_1 - 1} \theta_2^{\alpha_2 - 1} ... \theta_k^{\alpha_k - 1}$$

where $\alpha = (\alpha_1, \alpha_2, ..., \alpha_k)$ are the prior parameters. When $\alpha_1 = \alpha_2 = ... = \alpha_k = 1$, the prior is **uniform** over the probability simplex.

Posterior Distribution

Using **Bayes' theorem**, we update the prior with the observed data:

$$p(\theta | Y) \propto p(Y | \theta) p(\theta)$$

Since the **Dirichlet distribution is conjugate** to the multinomial, the posterior distribution remains **Dirichlet** with updated parameters:

$$\theta | Y \sim Dirichlet(\alpha_1 + Y_1, \alpha_2 + Y_2, ..., \alpha_k + Y_k)$$

Thus, **posterior inference** is straightforward: we simply add the observed counts to the prior parameters.

Computing the Posterior in R

We will now compute and visualize the posterior distribution for a multinomial dataset using R.

Load Required Packages

```r
library(LearnBayes)
library(ggplot2)
```

Define Observed Data

We assume we have a survey where respondents can choose among 4 categories:

```r
k <- 4  # Number of categories
Y <- c(50, 30, 15, 5)  # Observed counts for each category
n <- sum(Y)  # Total sample size
```

Set Prior Parameters (Uniform Prior)

We use a **noninformative prior** where all Dirichlet parameters are set to 1:

```r
alpha_prior <- rep(1, k)  # Dirichlet(1,1,1,1) prior
```

Compute Posterior Parameters

The posterior parameters are simply:

```r
alpha_posterior <- alpha_prior + Y
```

Simulating from the Posterior

We generate 5000 samples from the **Dirichlet posterior distribution**:

```r
library(MCMCpack)
```

```
## Loading required package: coda

## Loading required package: MASS

## ##
## ## Markov Chain Monte Carlo Package (MCMCpack)

## ## Copyright (C) 2003-2025 Andrew D. Martin, Kevin M. Quinn, and Jong Hee Park

## ##
## ## Support provided by the U.S. National Science Foundation

## ## (Grants SES-0350646 and SES-0350613)
## ##

##
## Attaching package: 'MCMCpack'

## The following object is masked from 'package:LearnBayes':
##
##     rdirichlet
```

```r
posterior_samples <- rdirichlet(5000, alpha_posterior)
colnames(posterior_samples) <- paste("Category", 1:k)
```

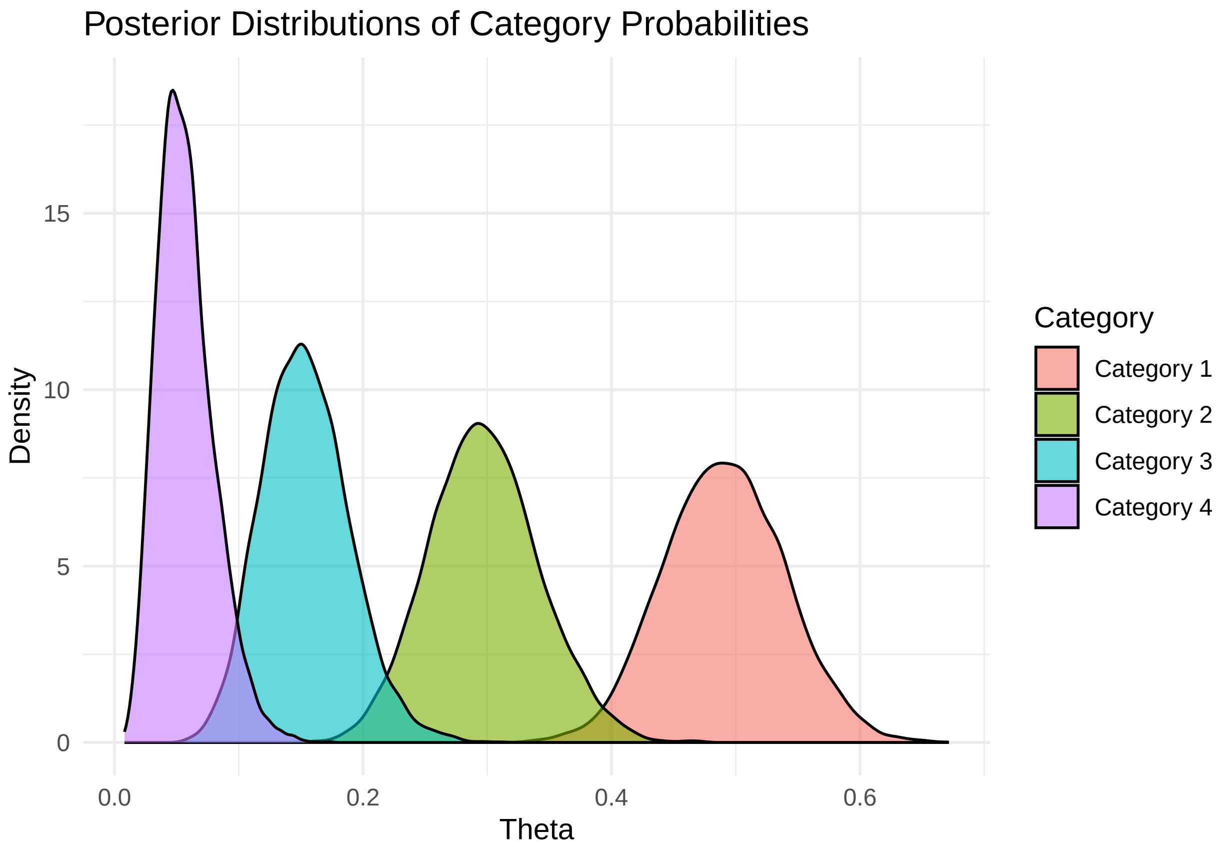
Visualizing the Posterior Distributions

We convert the samples into a **tidy format** for ggplot visualization:

```r
library(tidyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:MASS':
##
##     select

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
data_posterior <- as.data.frame(posterior_samples) %>%
  pivot_longer(cols = everything(), names_to = "Category", values_to = "Theta")

# Plot posterior distributions

ggplot(data_posterior, aes(x = Theta, fill = Category)) +
  geom_density(alpha = 0.6) +
  labs(title = "Posterior Distributions of Category Probabilities",
       x = "Theta", y = "Density") +
  theme_minimal()
```



Computing Credible Intervals for Each Category

```
credible_intervals <- apply(posterior_samples, 2, function(x) quantile(x, c(0.025, 0.975)))
credible_intervals
```

```
##       Category 1 Category 2 Category 3 Category 4
## 2.5%   0.3996537  0.2145412  0.0912194 0.02185714
## 97.5%  0.5865033  0.3864778  0.2284322 0.10863438
```

Cricket Example: Outcome of a Ball Bowled

In a cricket match, when a bowler delivers a ball, it can have multiple possible outcomes. Let's categorize these outcomes into:

1. **Dot Ball** (No runs scored)

2. **Single Run**

3. **Two Runs**

4. **Three Runs**

5. **Four Runs (Boundary)**

6. **Six Runs (Maximum Boundary)**

7. **Wicket Taken**

We assume we have observed data from a particular bowler's over (6 balls per over, across multiple overs). We will analyze the probability distribution of these outcomes using a **Bayesian multinomial model**.

Definition of the Multinomial Model

A **multinomial model** extends the binomial model, allowing multiple categories. Given n trials (balls bowled), we define:

- $Y = (Y_1, Y_2, ..., Y_k)$ as the counts of each outcome category.
- Each outcome occurs with probability θ_i.
- The probabilities satisfy:

$$\sum_{i=1}^{k} \theta_i = 1$$

- The likelihood function for the multinomial model is:

$$P(Y_1, Y_2, ..., Y_k | \theta_1, \theta_2, ..., \theta_k) = \frac{n!}{Y_1!Y_2!...Y_k!} \theta_1^{Y_1} \theta_2^{Y_2} ... \theta_k^{Y_k}$$

Choosing a Prior Distribution

A natural conjugate prior for the multinomial model is the **Dirichlet distribution**, defined as:

$$p(\theta_1, ..., \theta_k) \propto \theta_1^{\alpha_1 - 1} \theta_2^{\alpha_2 - 1} ... \theta_k^{\alpha_k - 1}$$

where $\alpha = (\alpha_1, \alpha_2, ..., \alpha_k)$ are the prior parameters. If $\alpha_1 = \alpha_2 = ... = \alpha_k = 1$, we assume a **uniform prior**.

Posterior Distribution

Using **Bayes' theorem**, we update the prior with observed data:

$$p(\theta|Y) \propto p(Y|\theta)p(\theta)$$

Since the **Dirichlet distribution is conjugate** to the multinomial likelihood, the posterior remains **Dirichlet**:

$$\theta|Y \sim Dirichlet(\alpha_1 + Y_1, \alpha_2 + Y_2, ..., \alpha_k + Y_k)$$

Thus, **posterior inference** is straightforward: we simply add the observed counts to the prior parameters.

Computing the Posterior in R for Cricket Data

We will compute and visualize the posterior distribution for **cricket ball outcomes** using R.

Load Required Packages

```
library(LearnBayes)
library(MCMCpack)
library(ggplot2)
library(tidyr)
library(dplyr)
```

Define Observed Data

We collect data from **analyzing 60 balls (10 overs) from a bowler's spell**:

```
k <- 7  # Number of outcome categories
Y <- c(30, 12, 6, 2, 5, 3, 2)  # Observed counts: (Dot, 1, 2, 3, 4, 6, Wicket)
n <- sum(Y)  # Total number of balls observed
outcome_labels <- c("Dot Ball", "Single Run", "Two Runs", "Three Runs", "Four Runs", "Six Runs", "Wicke
```

Set Prior Parameters (Uniform Prior)

Assuming a **noninformative prior**, we set all Dirichlet parameters to 1:

```
alpha_prior <- rep(1, k)  # Dirichlet(1,1,1,1,1,1,1) prior
```

Compute Posterior Parameters

The posterior parameters are:

```
alpha_posterior <- alpha_prior + Y
```

Simulating from the Posterior

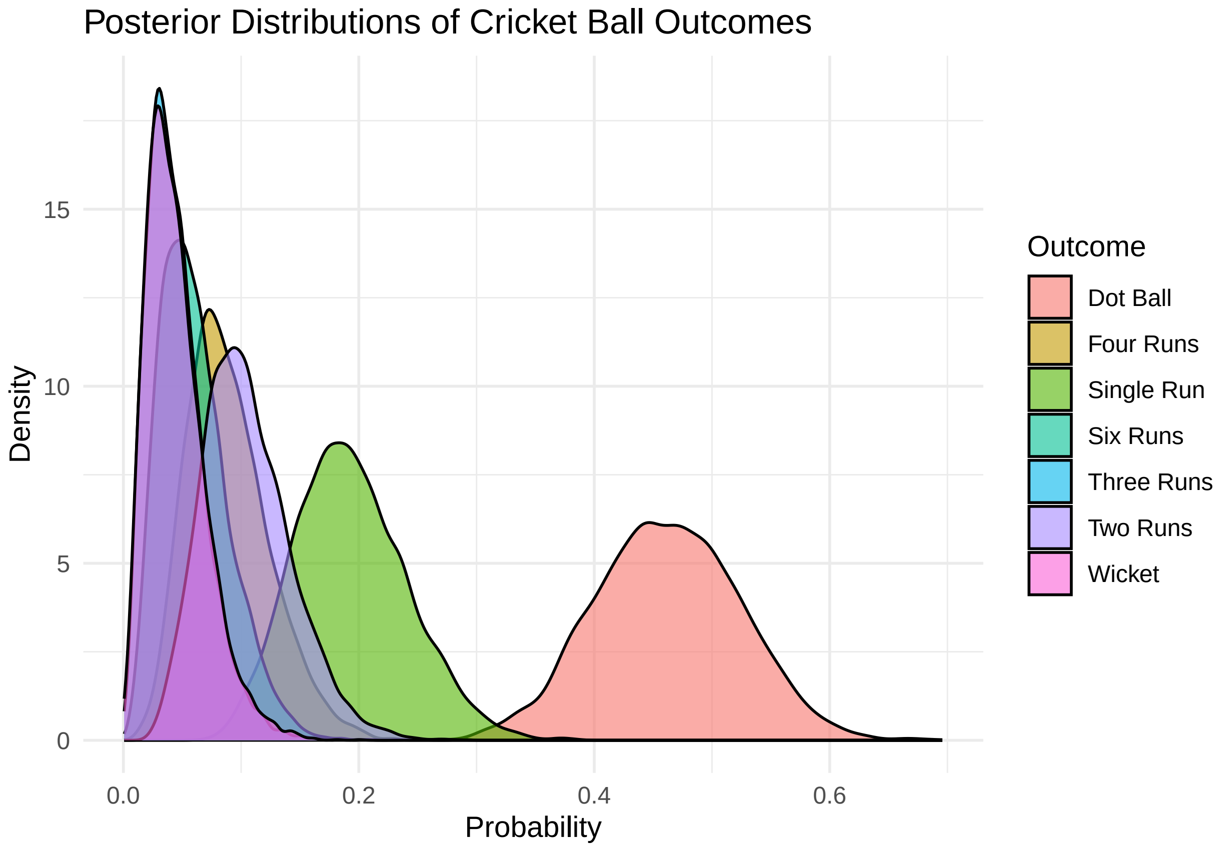
We generate 5000 samples from the **Dirichlet posterior distribution**:

```
posterior_samples <- rdirichlet(5000, alpha_posterior)
colnames(posterior_samples) <- outcome_labels
```

Visualizing the Posterior Distributions

We convert the samples into a **tidy format** for ggplot visualization:

```
data_posterior <- as.data.frame(posterior_samples) %>%
  pivot_longer(cols = everything(), names_to = "Outcome", values_to = "Theta")

# Plot posterior distributions

ggplot(data_posterior, aes(x = Theta, fill = Outcome)) +
  geom_density(alpha = 0.6) +
  labs(title = "Posterior Distributions of Cricket Ball Outcomes",
       x = "Probability", y = "Density") +
  theme_minimal()
```



Computing Credible Intervals for Each Outcome

```
credible_intervals <- apply(posterior_samples, 2, function(x) quantile(x, c(0.025, 0.975)))
credible_intervals
```

```
##        Dot Ball Single Run   Two Runs  Three Runs  Four Runs   Six Runs
## 2.5%  0.3418971  0.1092026 0.04302332 0.009849869 0.03410111 0.01678173
## 97.5% 0.5813864  0.2949915 0.18887107 0.103825383 0.16801824 0.12751486
##            Wicket
## 2.5%  0.009273467
## 97.5% 0.106620558
```

A Bioassay Experiment

In drug development, bioassay experiments are performed on animals to evaluate the effect of various dose levels of a compound. In this example (taken from Gelman et al., 2003), we administer different doses (recorded on a log-scale) to groups of animals and record a binary outcome (death or survival). Our goal is to analyze these data using a logistic regression model in a Bayesian framework. We will derive the likelihood, incorporate prior information through a *conditional means prior*, and finally summarize the posterior distribution by simulation.

Data Description

The experiment records the following information:

Dose (log g/ml)	Deaths	Sample Size
-0.86	0	5
-0.30	1	5
-0.05	3	5
0.73	5	5

Let
- x_i denote the dose (in log g/ml) for group i,
- n_i the number of animals in group i, and
- y_i the number of deaths observed at dose x_i.

We assume that

$$y_i \sim \text{Binomial}(n_i, p_i),$$

where p_i is the probability of death at dose x_i.

The Logistic Regression Model

We model the probability of death using a logistic regression. That is, we assume the log-odds (logit) of death is a linear function of the dose:

$$\text{logit}(p_i) = \log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 x_i.$$

Derivation of p_i

To express p_i in terms of β_0 and β_1, solve the equation:

$$\log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 x_i.$$

Exponentiate both sides:

$$\frac{p_i}{1-p_i} = \exp(\beta_0 + \beta_1 x_i).$$

Solving for p_i gives:

$$p_i = \frac{\exp(\beta_0 + \beta_1 x_i)}{1 + \exp(\beta_0 + \beta_1 x_i)}.$$

Likelihood Function

Since each y_i is binomial, the likelihood for the four groups is:

$$L(\beta_0, \beta_1) \propto \prod_{i=1}^{4} p_i^{y_i}(1-p_i)^{n_i - y_i},$$

with

$$p_i = \frac{\exp(\beta_0 + \beta_1 x_i)}{1 + \exp(\beta_0 + \beta_1 x_i)}.$$

Classical Analysis Using GLM

Before entering the Bayesian framework, one might analyze these data using classical maximum likelihood via the R function `glm()`. For example:

```
# Define the data vectors
x <- c(-0.86, -0.3, -0.05, 0.73)
n <- rep(5, 4)
y <- c(0, 1, 3, 5)

# Create the response matrix (deaths and survivors)
response <- cbind(y, n - y)

# Fit the logistic regression model
results <- glm(response ~ x, family = binomial)
summary(results)
```

```
##
## Call:
## glm(formula = response ~ x, family = binomial)
##
## Deviance Residuals:
##        1         2         3         4
## -0.17236   0.08133  -0.05869   0.12237
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.8466     1.0191   0.831    0.406
## x             7.7488     4.8728   1.590    0.112
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 15.791412  on 3  degrees of freedom
## Residual deviance:  0.054742  on 2  degrees of freedom
## AIC: 7.9648
##
## Number of Fisher Scoring iterations: 7
```

This output gives the estimates and standard errors for β_0 and β_1. However, our focus here is to use prior beliefs in a Bayesian analysis.

Incorporating Prior Information via a Conditional Means Prior

Suppose the experimenter has prior beliefs about the probability of death at two specific dose levels: - At $x_L = -0.7$, the prior median of p_L is 0.2 and the 90th percentile is 0.5. - At $x_H = 0.6$, the prior median of p_H is 0.8 and the 90th percentile is 0.98.

Using the `beta.select` function (which matches a beta distribution to percentile information), we obtain:

```
# person believes the median(p=0.5) of the prior is 0.2
# and the 90th percentile (p=0.9) of the prior is 0.5
# Matching prior information to beta distributions
beta.select(list(p = 0.5, x = 0.2), list(p = 0.9, x = 0.5))  # pL ~ Beta(1.12, 3.56)
```

```
## [1] 1.12 3.56
```

```
beta.select(list(p = 0.5, x = 0.8), list(p = 0.9, x = 0.98)) # pH ~ Beta(2.10, 0.74)
```

```
## [1] 2.10 0.74
```

Bayesian Posterior Distribution

We now combine prior and observed data to form the posterior:

```
# Observed data
x <- c(-0.86, -0.3, -0.05, 0.73)
n <- rep(5, 4)
y <- c(0, 1, 3, 5)
data <- cbind(x, n, y)

# Prior "data" from conditional means prior
prior <- rbind(c(-0.7, 4.68, 1.12),
               c(0.6, 2.84, 2.10))

# Combine the observed and prior data:
data.new <- rbind(data, prior)
```

Understanding the Prior Data Matrix

The **Beta(a, b)** prior can be interpreted as if it came from an earlier, hypothetical experiment where: - a represents the number of deaths. - b represents the number of survivals. - The total number of trials is $a + b$.

Thus, in the prior data matrix:

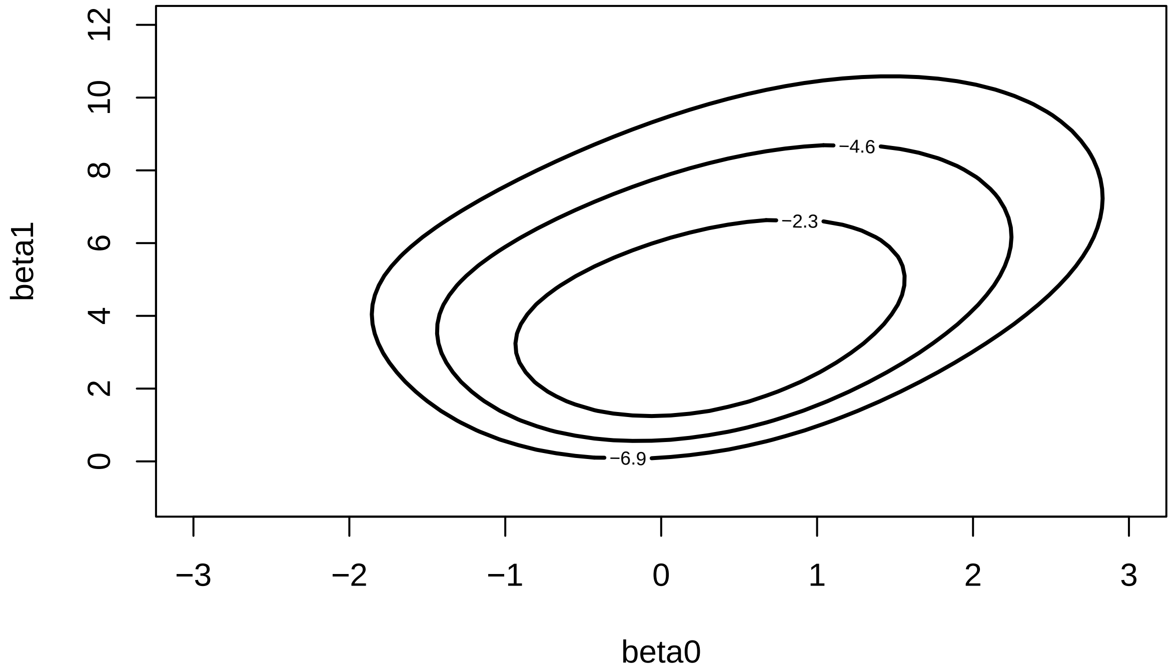
```
prior <- rbind(c(-0.7, 4.68, 1.12),  # Dose = -0.7, n = 4.68, deaths = 1.12
               c(0.6, 2.84, 2.10))    # Dose = 0.6, n = 2.84, deaths = 2.10
```

- For dose $x_L = -0.7$, the prior corresponds to $n = 4.68$, deaths $= 1.12$, and survivals $= 3.56$.
- For dose $x_H = 0.6$, the prior corresponds to $n = 2.84$, deaths $= 2.10$, and survivals $= 0.74$.

These values represent pseudo-data derived from prior beliefs and are incorporated into the Bayesian analysis as additional observations.

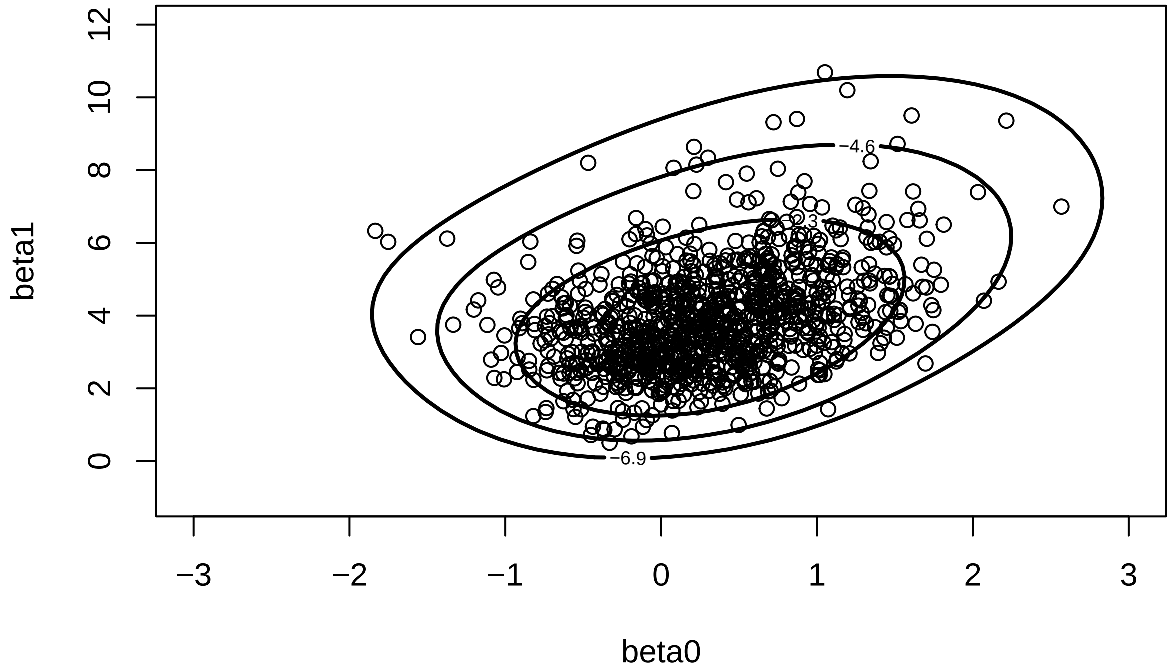
Posterior Contour Plot

```
# Load necessary package
library(LearnBayes)

# Contour plot of posterior
data.new <- as.matrix(data.new)
mycontour(logisticpost, c(-3, 3, -1, 12), data.new,
          xlab = "beta0", ylab = "beta1")
```



Simulating from the Posterior

```
# Simulate samples from posterior
mycontour(logisticpost, c(-3, 3, -1, 12), data.new,
          xlab = "beta0", ylab = "beta1")
s <- simcontour(logisticpost, c(-2, 3, -1, 11), data.new, 1000)
points(s)
```



Derivation of LD-50

The LD-50 (lethal dose for 50% of subjects) is the dose level x at which the probability of death is exactly 50%, i.e.,

$$\Pr(X = x) = 0.5$$

Using the logistic regression model,

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x.$$

1. **Set probability to 0.5**:
$$\log\left(\frac{0.5}{1-0.5}\right) = \beta_0 + \beta_1 x.$$

2. **Simplify the left-hand side**:
$$\log(1) = 0.$$

   So, the equation reduces to:
$$0 = \beta_0 + \beta_1 x.$$

3. **Solve for x (LD-50)**:
$$x = -\frac{\beta_0}{\beta_1}.$$

Thus, the **LD-50** is:

$$\theta = -\frac{\beta_0}{\beta_1}.$$

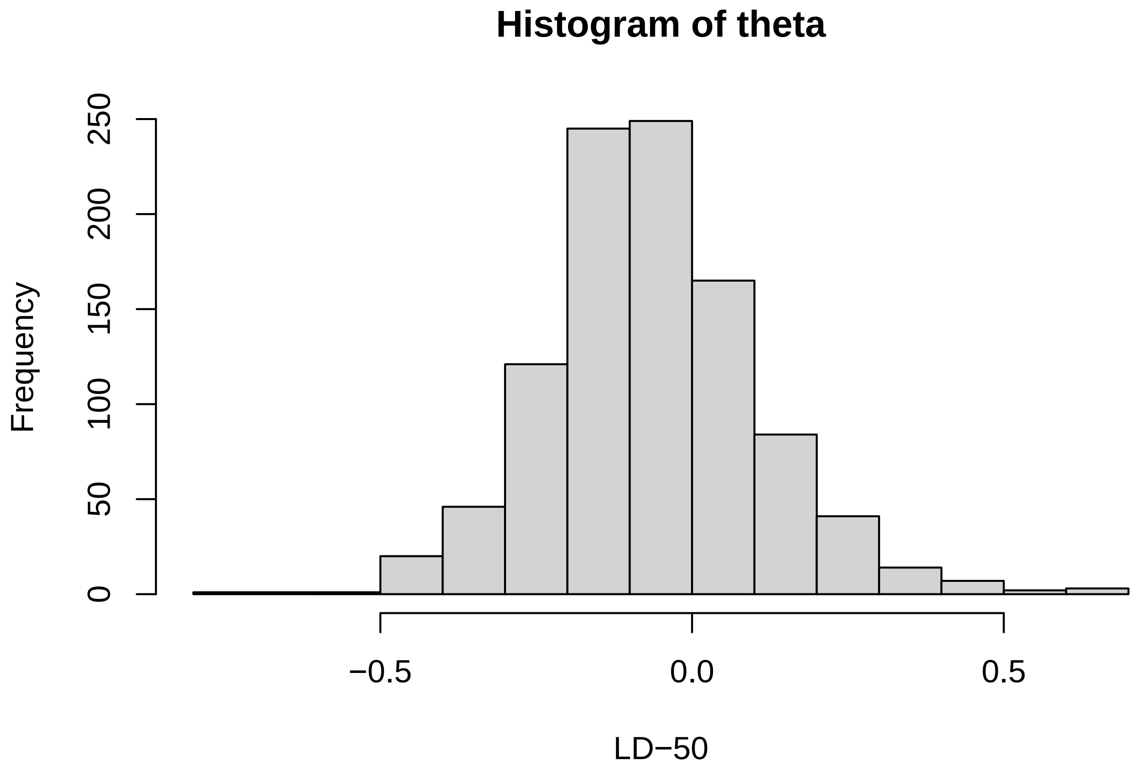
This means that LD-50 represents the dose at which there is a 50% probability of death, providing an important measure in toxicity studies.

Estimating LD-50

The LD-50 (lethal dose for 50% of subjects) is calculated as:

$$\theta = -\frac{\beta_0}{\beta_1}.$$

```r
theta <- -s$x / s$y
hist(theta, xlab = "LD-50", breaks = 20)
```



```r
quantile(theta,c(.025,.975))
```

```
##       2.5%      97.5% 
## -0.3944688  0.3075908
```

Comparing Two Proportions

In many real-world scenarios, we are interested in comparing two proportions to determine whether there is a significant difference between them. A typical example arises in sports, where we may wish to compare the success rates of two teams or players under similar conditions. In this section, we will explore the Bayesian approach to comparing two proportions using a sports-related example.

Problem Setup: Comparing Free Throw Success Rates

Consider a basketball scenario where we compare the free throw success rates of two players: **Player A** and **Player B**. Suppose we have observed the following data:

```r
library(knitr)
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'

## The following object is masked from 'package:dplyr':
##
##     group_rows
```

```r
# Creating a centered table for better visibility
data <- data.frame(
  Player = c("Player A", "Player B"),
  Successful_Free_Throws = c(85, 78),
  Total_Attempts = c(100, 100)
)
kable(data, caption = "Observed Free Throw Success Data", align = "c") %>%
  kable_styling(full_width = FALSE, position = "center")
```

Table 2: Observed Free Throw Success Data

Player	Successful_Free_Throws	Total_Attempts
Player A	85	100
Player B	78	100

We want to estimate the success rates of both players and determine the probability that Player A is better than Player B.

Bayesian Model

Let p_A and p_B be the probabilities of making a free throw for Player A and Player B, respectively. We assume independent **Binomial** distributions for the observed data:

$$Y_A \sim \text{Binomial}(n_A, p_A), \quad Y_B \sim \text{Binomial}(n_B, p_B),$$

where: - $Y_A = 85$ successful free throws out of $n_A = 100$ attempts for Player A. - $Y_B = 78$ successful free throws out of $n_B = 100$ attempts for Player B.

Prior Distributions

We assign independent **Beta priors** to the success probabilities:

$$p_A \sim \text{Beta}(2, 2), \quad p_B \sim \text{Beta}(2, 2).$$

This choice represents a weakly informative prior, assuming each player has an initial probability centered around 0.5 but allowing for variation.

Posterior Distribution

Using **Bayes' Theorem**, the posterior distributions are also Beta-distributed:

$$p_A | Y_A \sim \text{Beta}(85 + 2, 100 - 85 + 2),$$
$$p_B | Y_B \sim \text{Beta}(78 + 2, 100 - 78 + 2).$$

These distributions incorporate both prior beliefs and observed data.

Simulation from Posterior

We approximate the posterior distributions using Monte Carlo simulation:

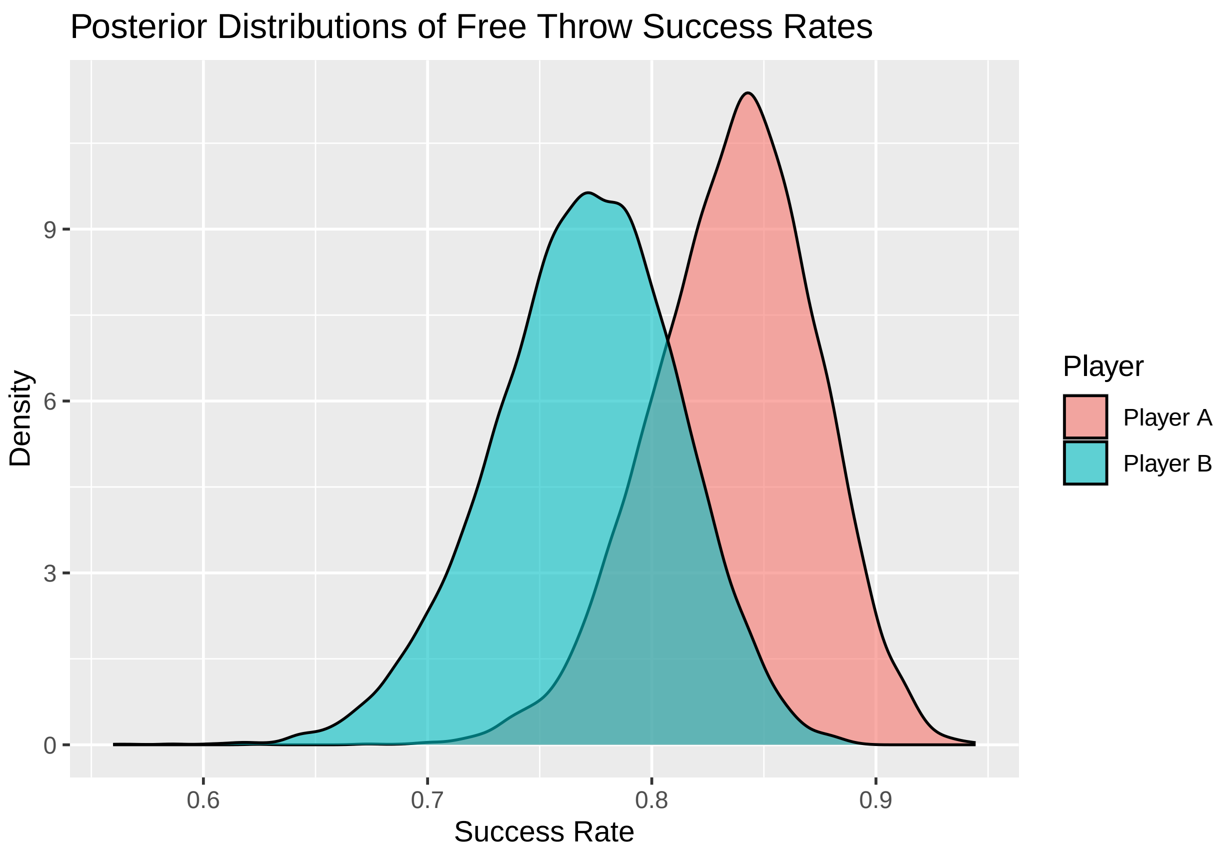
```r
set.seed(123)
n_samples <- 10000

# Simulating posterior samples for p_A and p_B
p_A_samples <- rbeta(n_samples, 85 + 2, 100 - 85 + 2)
p_B_samples <- rbeta(n_samples, 78 + 2, 100 - 78 + 2)

# Compute probability that p_A > p_B
prob_A_better <- mean(p_A_samples > p_B_samples)

# Display result
print(paste("Probability that Player A is better than Player B:", round(prob_A_better, 4)))
```

```
## [1] "Probability that Player A is better than Player B: 0.8918"
```

Visualization of Posterior Distributions

We plot the posterior distributions of p_A and p_B to visualize the uncertainty:

```r
library(ggplot2)

data <- data.frame(
  Probability = c(p_A_samples, p_B_samples),
  Player = rep(c("Player A", "Player B"), each = n_samples)
)

ggplot(data, aes(x = Probability, fill = Player)) +
  geom_density(alpha = 0.6) +
  labs(title = "Posterior Distributions of Free Throw Success Rates",
       x = "Success Rate", y = "Density")
```



Posterior Probability of Superior Performance

The key Bayesian question is: **What is the probability that Player A is better than Player B?**

Using the simulated posterior samples:

```r
prob_A_better <- mean(p_A_samples > p_B_samples)
cat("The probability that Player A is better than Player B:", round(prob_A_better, 4), "\n")
```

```
## The probability that Player A is better than Player B: 0.8918
```

This probability provides a direct Bayesian answer to our question, without reliance on p-values.

Dependent Prior for Comparing Two Proportions

Howard (1998) introduced a **dependent prior** to model the relationship between two proportions. Instead of assuming p_A and p_B are independent, we introduce a correlation between them using a transformation:

$$\theta_A = \log\left(\frac{p_A}{1-p_A}\right), \quad \theta_B = \log\left(\frac{p_B}{1-p_B}\right)$$

We assume that, given θ_A, the logit of p_B follows a normal distribution:

$$\theta_B|\theta_A \sim N(\theta_A, \sigma^2)$$

where σ controls the level of dependence.

Derivation of the Dependent Prior

The dependent prior takes the form:

$$g(p_A, p_B) \propto e^{-\frac{1}{2}u^2} p_A^{\alpha-1}(1-p_A)^{\beta-1} p_B^{\gamma-1}(1-p_B)^{\delta-1},$$

where:

$$u = \frac{1}{\sigma}(\theta_A - \theta_B).$$

Derivation of the Dependent Prior

The dependent prior takes the form:

$$g(p_A, p_B) \propto e^{-\frac{1}{2}u^2} p_A^{\alpha-1}(1-p_A)^{\beta-1} p_B^{\gamma-1}(1-p_B)^{\delta-1},$$

where:

$$u = \frac{1}{\sigma}(\theta_A - \theta_B).$$

Step-by-Step Derivation Using Logit Transformation

1. We assume that the transformed logit parameters θ_A and θ_B are related via a normal distribution:

   $$\theta_B | \theta_A \sim N(\theta_A, \sigma^2)$$

   This means that the conditional density of θ_B given θ_A is:

   $$f(\theta_B|\theta_A) \propto e^{-\frac{1}{2}(\theta_B - \theta_A)^2/\sigma^2}.$$

2. Using the logit transformation:

   $$\theta_B = \log\left(\frac{p_B}{1-p_B}\right),$$

   we need to express this in terms of p_B.

3. Compute the Jacobian determinant for the transformation:

   $$J = \left|\frac{d\theta_B}{dp_B}\right| = \frac{1}{p_B(1-p_B)}.$$

4. Apply the change of variables:

   $$f(p_B|p_A) = f(\theta_B|\theta_A) \times J.$$

   Substituting:

   $$f(p_B|p_A) \propto e^{-\frac{1}{2}u^2} \times \frac{1}{p_B(1-p_B)}.$$

5. Incorporating the individual Beta priors:

$$\begin{aligned} f(p_A) &\propto p_A^{\alpha-1}(1-p_A)^{\beta-1}, \\ f(p_B) &\propto p_B^{\gamma-1}(1-p_B)^{\delta-1}. \end{aligned}$$

6. The final joint prior is:

$$g(p_A, p_B) \propto e^{-\frac{1}{2}u^2} p_A^{\alpha-1}(1-p_A)^{\beta-1} p_B^{\gamma-1}(1-p_B)^{\delta-1}.$$

This prior incorporates both individual beliefs about p_A and p_B as well as the dependence structure controlled by σ.

This class of priors is indexed by parameters $(\alpha, \beta, \gamma, \delta, \sigma)$, which reflect our prior beliefs about the individual proportions and their dependence.

Simulation Using a Sports Example

We assume that the players' success rates are dependent. Suppose that knowing Player A has a high free throw percentage makes us believe Player B is also likely to be good.

We set priors:

```
alpha <- 2  # Prior success count for Player A
beta <- 2   # Prior failure count for Player A
gamma <- 2  # Prior success count for Player B
delta <- 2  # Prior failure count for Player B
sigma <- 1  # Strength of dependence

# Define function for Howard's dependent prior
howard_prior <- function(p_A, p_B, alpha, beta, gamma, delta, sigma) {
  theta_A <- log(p_A / (1 - p_A))
  theta_B <- log(p_B / (1 - p_B))
  u <- (theta_A - theta_B) / sigma
  exp(-0.5 * u^2) * p_A^(alpha-1) * (1 - p_A)^(beta-1) * p_B^(gamma-1) * (1 - p_B)^(delta-1)
}
```

We then simulate posterior samples under the dependent prior and compare the results:

```
n_samples <- 10000

# Simulate independent priors
p_A_samples <- rbeta(n_samples, 85 + alpha, 100 - 85 + beta)
p_B_samples <- rbeta(n_samples, 78 + gamma, 100 - 78 + delta)

# Compute dependent prior weights
prior_weights <- howard_prior(p_A_samples, p_B_samples, alpha, beta, gamma, delta, sigma)

# Normalize weights
prior_weights <- prior_weights / sum(prior_weights)

# Sample from posterior under dependent prior
idx <- sample(1:n_samples, n_samples, replace = TRUE, prob = prior_weights)
p_A_dep <- p_A_samples[idx]
p_B_dep <- p_B_samples[idx]
```

```r
# Compute probability that Player A is better than Player B
prob_A_better_dep <- mean(p_A_dep > p_B_dep)

print(paste("Probability that Player A is better than Player B (dependent prior):", round(prob_A_better_
```

```
## [1] "Probability that Player A is better than Player B (dependent prior): 0.8698"
```