STAT8310 - Bayesian Data Analysis

Chi-Kuang Yeh

2026-05-03

Table of contents

Preface

Thank you

This course has been **successfully completed**! Thank you everyone for the participation and attention. Hope you all have learned something from this course, and have fun. Wish everyone has a good summer vacation!

Description

This course will cover the topics in the theory and practice of *Bayesian statistical inference*, ranging from a review of fundamentals to questions of current research interest. Motivation for the Bayesian approach. Bayesian computation, Monte Carlo methods, asymptotics. Model checking and comparison. A selection of examples and issues in modelling and data analysis. Discussion of advantages and difficulties of the Bayesian approach. This course will be computationally intensive through analysis of data sets using the R statistical computing language.

Prerequisites

MATH 4752/6752 – Mathematical Statistics II or equivalent, and the ability to program in a high-level language.

Instructor

Chi-Kuang Yeh, Assistant Professor in the Department of Mathematics and Statistics, Georgia State University.

- Office: Suite 1407, 25 Park Place.
- Email: cyeh@gsu.edu.

Office Hour

10:00–13:00 on Monday, or by appointment.

Grade Distribution

- Homework – 50%
- Exam – 30%
- Final – 20%

Assignment

- ☒ A1, due on Jan 29, 2026
- ☒ A2, due on Feb 13, 2026
- ☒ A3, due on Mar 01, 2026
- ☒ A4, due on Mar 25, 2026
- ☒ A5, due on Apr 20, 2026

Midterm

- ☒ March 5, 2026

Final Project

- ☒ Due on May 1, 2026

More information can be found on the project page

Topics and Corresponding Lectures

Those chapters are based on the lecture notes. This part will be updated frequently.

Status	Chapter	Topic	Lecture
	Ch. 1	Welcome and Overview	1
	—	Intro to R Programming	2

Status	Chapter	Topic	Lecture
	Ch. 2	Probability and Exchangeability	3–5
	Ch. 3	One Parameter Models	6–9
	Ch. 4	Monte Carlo	10– 13
	Ch.5	Gibbs Sampler	14,17–18, 20
		Midterm Review	15
		Midterm Exam	16
		Spring Break	
	-	Intro to JAGS and BUGS	19
	Ch.6	MCMC Diagnosis	20-22
	Ch.7	Multivariate Gaussian	22– 26
	Ch. 8	Select topic: Bayesian Machine Learning:	27
	Ch. 9	Select topic: Modern Bayesian Computation	28

Recommended Textbooks

- Gelman, A., Carlin, J., Stern, H., Rubin, D., Dunson, D., and Vehtari, A. (2021). Bayesian Data Analysis, CRC Press, 3rd Ed.
- Hoff, P.D. (2009). A First Course in Bayesian Statistical Methods, Springer.
- McElreath, R. (2018). Statistical Rethinking: A Bayesian Course with Examples in R and Stan, CRC Press.

Side Readings

- Just Another Gibbs Sampler Page
- BUGS and Winbugs from MRC Biostatistics unit Page

1 Quick Overview

The posterior distribution is obtained from the prior distribution and sampling model via *Bayes' rule*:

$$p(\theta \mid y) = \frac{p(y \mid \theta)p(\theta)}{\int_{\Theta} p(y \mid \theta')p(\theta')d\theta'}.$$

1.1 Why Bayesian?

- **Intuitive probability interpretation**: Directly quantifies uncertainty about parameters as probability distributions
- **Incorporates prior knowledge**: Systematically combines domain expertise with data through the prior distribution
- **Principled inference**: Bayes' rule provides a coherent framework for updating beliefs based on evidence
- **Natural handling of uncertainty**: Posterior distributions capture full uncertainty, not just point estimates
- **Sequential analysis**: Easily updates beliefs as new data arrives (posterior becomes new prior)
- **Small sample inference**: Performs well with limited data by leveraging prior information
- **Prediction with uncertainty**: Generates predictive distributions that quantify uncertainty in future observations
- **Decision-making**: Naturally incorporates loss functions for optimal decision rules
- **Model comparison**: Bayes factors provide a principled approach to comparing competing models

---

1.2 Some Bayesian Topics and their Computational Focus

Table 1.1: Some of the Bayesian Topics and its computational related focuses.

Topics	**Key Concepts / Readings**	**Computing Focus**
Introduction to Bayesian Thinking	Bayesian vs. Frequentist paradigms; Prior, likelihood, posterior	Review of R basics and reproducible workflows
Bayesian Inference for Simple Models	Conjugate priors, Beta-Binomial, Normal-Normal, Poisson-Gamma	Simulating posteriors, visualization
Prior Elicitation and Sensitivity	Informative vs. noninformative priors, Jeffreys prior	Prior sensitivity plots
Monte Carlo Integration	Law of large numbers, sampling-based inference	Random sampling and Monte Carlo approximation
Markov Chain Monte Carlo (MCMC)	Metropolis-Hastings, Gibbs sampler	Implementing MCMC in R
Convergence Diagnostics	Trace plots, autocorrelation, Gelman–Rubin statistic	`coda`, `rstan`, and `bayesplot` packages
Hierarchical Bayesian Models	Partial pooling, shrinkage, multilevel structures	`rstanarm` / `brms`
Midterm Project: Bayesian Linear Regression	Posterior inference for regression, model selection	`brms`, `rstanarm`, custom Gibbs samplers
Bayesian Model Comparison	Bayes factors, BIC, DIC, WAIC, LOO	Practical comparison via cross-validation
Model Checking and Diagnostics	Posterior predictive checks, residual analysis	`pp_check` in `brms`
Advanced Computation	Hamiltonian Monte Carlo (HMC), Variational Inference	Using `Stan` and `CmdStanR`
Bayesian Decision Theory	Utility functions, decision rules, loss minimization	Simple decision problems in R
Modern Bayesian Methods	Approximate Bayesian computation (ABC), Bayesian neural networks	Examples via `rstan` or `tensorflow-probability`
Student Project Presentations	Applications and case studies	Full workflow demonstration in R

1.3 Interesting Article:

- Goligher, E.C., Harhay, M.O. (2023). What Is the Point of Bayesian Analysis?, American Journal of Respiratory and Critical Care Medicine, 209, 485–487.

2 Belief function and Probability Review

Leading objectives:

be familiar with the following concepts

- Belief Functions
- Probability
- Bayes' Rule
- Random Variables
- Exchangeability

2.1 Belief functions

Probability is a way to express rational beliefs.

A **belief function** Be$(\cdot)$ is a function that assigns number to statements such that the large the number, the higher the degree of belief.

Let F, G, and H be three possibly overlapping statements about the world.

For example:

- F = { a person owns a smartphone }
- G = { a person uses social media daily }
- H = { a person works remotely at least part of the time }

or

- F = { a person has a graduate degree }
- G = { a person works in a STEM field }
- H = { a person is employed in the private sector }

The perference over bets involving these statements can be used to define a belief function

- Be$(F) >$ Be(G) means you prefer a bet F is true over that G is true.

Also, we want Be$(\cdot)$ to describe our beliefs under certain conditions

- $\text{Be}(F \mid H) > \text{Be}(G \mid H)$ means that if we knew that H were true, then we would perfer to bet that F is also true over G is also true.
- $\text{Be}(F \mid G) > \text{Be}(F \mid H)$ means that if we bet on F, we would perfer to do it under the condition that G is true rather than H is true.

Some more notations:

- Let $\neg$ denote negation. That is, $\neg F$ is the statement that F is not true.
- Let $F \vee G$ denote the disjunction (or) of statements F and G, meaning that at least one of F or G is true.
- Let $F \wedge G$ denote the conjunction (and) of statements F and G, meaning that both F and G are true.

It has been argued by many that any function that is to numerically represent our beliefs should have the following properties:

- **B1**: $\text{Be}(\neg H \mid H) \leq \text{Be}(F \mid H) \leq \text{Be}(H \mid H)$
- **B2**: $\text{Be}(F \vee G \mid H) \geq \max\{\text{Be}(F \mid H), \text{Be}(G \mid H)\}$
- **B3**: $\text{Be}(F \wedge G \mid H)$ can be derived from $\text{Be}(G \mid H)$ and $\text{Be}(F \mid G \wedge H)$.

How should we interpret these properties, and do they make sense?

- B1 means that the number we assign to $\text{Be}(F \mid H)$, our conditional belief in F given H, is bounded below and above by the numbers we assign to complete disbelief $\text{Be}(\neg H \mid H)$ and complete belief $\text{Be}(H \mid H)$.
- B2 says that our belief that the truth lies in a given set of possibilities should not decrease as we add to the set of possibilities.
- B3 is a bit trickier. To see why it makes sense, imagine you have to decide whether or not F and G are true, knowing that H is true. You could do this by first deciding whether or not G is true given H, and if so, then deciding whether or not F is true given G and H.

Recall the notation from (elementary) probability that, $F \cup G$ means F or G, and $F \cap G$ means F and G, and $\emptyset$ is the empty set.

- **P1**:
$$0 = \Pr(\neg H \mid H) \leq \Pr(F \mid H) \leq \Pr(H \mid H) = 1$$
- **P2**:
$$\Pr(F \cup G \mid H) = \Pr(F \mid H) + \Pr(G \mid H), \quad \text{if} \quad F \cap G = \emptyset$$
- **P3**:
$$\Pr(F \cap G \mid H) = \Pr(G \mid H)\Pr(F \mid G \cap H)$$

2.1.1 Conclusion

You can see that, a probability function satisfy P1–P3 also satisfies B1–B3. Therefore, probability functions are a special case of belief functions, and we can use it to describe our belief.

2.2 Events, Partitions and Bayes' Rule

A collectiion of sets $\{H_1, \dots, H_K\}$ is a partition of another set $\mathcal{H}$ if

1. $H_i \cap H_j = \emptyset$ for all $i \neq j$ (mutually exclusive);
2. $\bigcup_{i=1}^{K} H_i = \mathcal{H}$ (collectively exhaustive).

In the context of indetifying which of several statements is true, if $\mathcal{H}$ is the set of all possible truths and $\{H_1, \dots, H_K\}$ is a partition of $\mathcal{H}$, then exactly one set H_j contains the truth.

Let $\mathcal{H}$ be the status of a statistical model.

Valid partitions include:

- {correctly specified, misspecified}
- {underfitting, well-specified, overfitting}

2.2.1 Partition and Probability

Suppose $\{H_1, \dots, H_K\}$ is a partition of $\mathcal{H}$,$\Pr(\mathcal{H}) = 1$ and E is some specific event. Then, by the axioms of probability, we have

- Law of total probability

$$\sum_{k=1}^{K} \Pr(H_k) = \Pr\left(\bigcup_{k=1}^{K} H_k\right) = \Pr(\mathcal{H}) = 1$$

- Law of marginal probability

$$\Pr(E) = \sum_{k=1}^{K} \Pr(E \cap H_k) = \sum_{k=1}^{K} \Pr(E \mid H_k)\Pr(H_k)$$

- Bayes' rule

$$\Pr(H_j \mid E) = \frac{\Pr(E \mid H_j)\Pr(H_j)}{\Pr(E)} = \frac{\Pr(E \mid H_j)\Pr(H_j)}{\sum_{k=1}^{K} \Pr(E \mid H_k)\Pr(H_k)}$$

A subset of the 1996 General Social Survey includes data on the education level and income for a sample of males over 30 years of age. Let {H1,H2,H3,H4} be the events that a randomly selected person in this sample is in, respectively, the lower 25th percentile, the second 25th percentile, the third 25th percentile and the upper 25th percentile in terms of income. By definition,

$$\{\Pr(H_1), \Pr(H_2), \Pr(H_3), \Pr(H_4)\} = \{.25, .25, .25, .25\}.$$

Note that $\{H1, H2, H3, H4\}$ is a partition and so these probabilities sum to 1. Let E be the event that a randomly sampled person from the survey has a college education. From the survey data, we have

$$\{\Pr(E \mid H_1), \Pr(E \mid H_2), \Pr(E \mid H_3), \Pr(E \mid H_4)\} = \{.11, .19, .31, .53\}.$$

These probabilities do not sum to 1 - they represent the proportions of people with college degrees in the four different income subpopulations H_1, H_2, H_3 and H_4. Now let's consider the income distribution of the college-educated population. Using Bayes' rule we can obtain

$\{\Pr(H_1 \mid E), \Pr(H_2 \mid E), \Pr(H_3 \mid E), \Pr(H_4 \mid E)\} = \{0.09, 0.17, 0.27, 0.47\}$, and we see that the income distribution for people in the college-educated population differs markedly from $\{0.25, 0.25, 0.25, 0.25\}$, the distribution for the general population. Note that these probabilities do sum to 1 - they are the conditional probabilities of the events in the partition, given E.

In Bayesian inference, $H_1, ..., H_K$ often refer to disjoint hypotheses or states of nature and E refers to the outcome of a survey, study or experiment. To compare hypotheses *post-experimentally*, we often calculate the following ratio:

$$\begin{aligned}
\frac{\Pr(H_i \mid E)}{\Pr(H_j \mid E)} &= \frac{\Pr(E \mid H_i)\Pr(H_i) / \Pr(E)}{\Pr(E \mid H_j)\Pr(H_j) / \Pr(E)} \\
&= \frac{\Pr(E \mid H_i)\Pr(H_i)}{\Pr(E \mid H_j)\Pr(H_j)} \\
&= \frac{\Pr(E \mid H_i)}{\Pr(E \mid H_j)} \times \frac{\Pr(H_i)}{\Pr(H_j)} \\
&= \text{ "Bayes factor" } \times \text{ "prior beliefs".}
\end{aligned}$$

This calculation reminds us that Bayes' rule does not determine what our *beliefs should be* after seeing the data, it only tells us how they *should change after seeing the data.*

2.3 Independence

Two events F and G are conditionally independent, if given H, we have $\Pr(F \cap G \mid H) = \Pr(F \mid H)\Pr(G \mid H)$.

How do we interpret conditional independence? By Axiom P3, the following is always true:

$$\begin{aligned}
\Pr(G \mid H)\Pr(F \mid H \cap G) &\overset{\text{always}}{=} \Pr(F \cap G \mid H) &&\overset{\text{independence}}{=} \Pr(F \mid H)\Pr(G \mid H) \\
\Pr(G \mid H)\Pr(F \mid H \cap G) &= && \Pr(F \mid H)\Pr(G \mid H) \\
\Pr(F \mid H \cap G) &= && \Pr(F \mid H).
\end{aligned}$$

Thus, conditional independence implies that $\Pr(F \mid H \cap G) = \Pr(F \mid H)$. In other words, if we know H is true, and F and G are conditionally independent given H, then knowing G does not change our belief about F.

Let's consider the conditional dependence of F and G when H is assumed to be true in the following two situations:

Siutation 1:

- F = { a hospital patient is a smoker }
- G = { a hospital patient has lung cancer }
- H = { smoking causes lung cancer}

Situation 2:

- F = { a student studies regularly for an exam }
- G = { a student receives a high exam score }
- H = { studying improves exam performance }

Think: In both of these situations, H being true implies a relationship between F and G. What about when H is not true?

2.4 Random Variables

In Bayesian inference a random variable is defined as an unknown numerical quantity about which we make probability statements. For example, the quantitative outcome of a survey, experiment or study is a random variable before the study is performed. Additionally, a fixed but unknown population parameter is also a random variable

2.4.1 Discrete Ramdon variables

Let Y be a random variable and let $\mathcal{Y}$ be the set of all possible values that Y can take. If $\mathcal{Y}$ is countable, meaning that $\mathcal{Y} = \{y_1, y_2, ...\}$, then Y is a discrete random variable.

The event that the outcome Y of our survey has the value Y is expressed as $\{Y = y\}$. For each $y \in \mathcal{Y}$, the shorthand notation for $\Pr(Y = y)$ is $p(y)$, and $p(\cdot)$ is called the **probability mass function** of Y, and with two properties

1. $0 \leq p(y) \leq 1$ for all $y \in \mathcal{Y}$,
2. $\sum_{y \in \mathcal{Y}} p(y) = 1$.

General probability statements about Y can be derived from the pdf/pmf, for example, for any subset $A \subseteq \mathcal{Y}$, we have $\Pr(Y \in A) = \sum_{y \in A} p(y)$. When we have two disjoint subsets A and B of $\mathcal{Y}$, we have

$$\Pr(Y \in A \cup B) = \Pr(Y \in A) + \Pr(Y \in B) = \sum_{y \in A} p(y) + \sum_{y \in B} p(y).$$

Let Y be the number of successes in n independent Bernoulli trials, each with probability of success θ. Then, Y follows a Binomial distribution with parameters n and θ, denoted as $Y \sim \text{Binomial}(n, p)$. The probability mass function of Y is given by

$$p(y) = \Pr(Y = y) = \binom{n}{y} \theta^y (1-\theta)^{n-y}, \quad y = 0, 1, 2, \ldots, n.$$

If $\theta = 0.3$ and $n = 3$, then the probability of observing exactly 2 successes is

$$p(2) = \Pr(Y = 2 \mid \theta = 0.3) = \binom{3}{2} (0.3)^2 (0.7)^1 = 3 \cdot 0.09 \cdot 0.7 = 0.189.$$

2.4.2 Continuous random variables

If $\mathcal{Y}$ is uncountable, for example, $\mathcal{Y} = \mathbb{R}$ or $\mathcal{Y} = (0, 1)$, then Y is a continuous random variable. In this case, we cannot list all possible values of Y and assign probabilities to each value. Instead, we use a probability distribution to describe the distribution of Y. That is, the cummulative distribution function (cdf) defined as follows.

The **cumulative distribution function** (cdf) of a continuous random variable Y is defined as

$$F(y) = \Pr(Y \leq y), \quad y \in \mathcal{Y}.$$

Note that, for the cdf $F(y)$, we have the following properties:

- $0 \leq F(y) \leq 1$ for all $y \in \mathcal{Y}$,
- $F(y)$ is non-decreasing, meaning that if $y_1 < y_2$, then $F(y_1) \leq F(y_2)$,
- $\lim_{y \to -\infty} F(y) = 0$
- $\lim_{y \to \infty} F(y) = 1$.

Probability of various events can be derived from the cdf. For example, for any interval $A = (a, b] \subseteq \mathcal{Y}$, we have

$$\Pr(Y \in A) = \Pr(a < Y \leq b) = F(b) - F(a).$$

Also, $\Pr(Y \leq a) = F(a)$ and $\Pr(Y > a) = 1 - F(a)$.

2.4.3 Description of distributions through quantiles and moments

In this subsection, we discuss a few ways to describe probability distributions: quantiles and moments. They are used to describe the behaviour of the distribution compressing them into summary statistics.

The **expectation** or **mean** of a random variable Y can be thought as the centre of mass or the location of the distribution, which is defined as

- For discrete random variable:
$$E(Y) = \sum_{y \in \mathcal{Y}} yp(y).$$
- For continuous random variable:
$$E(Y) = \int_{y} yf(y)dy.$$

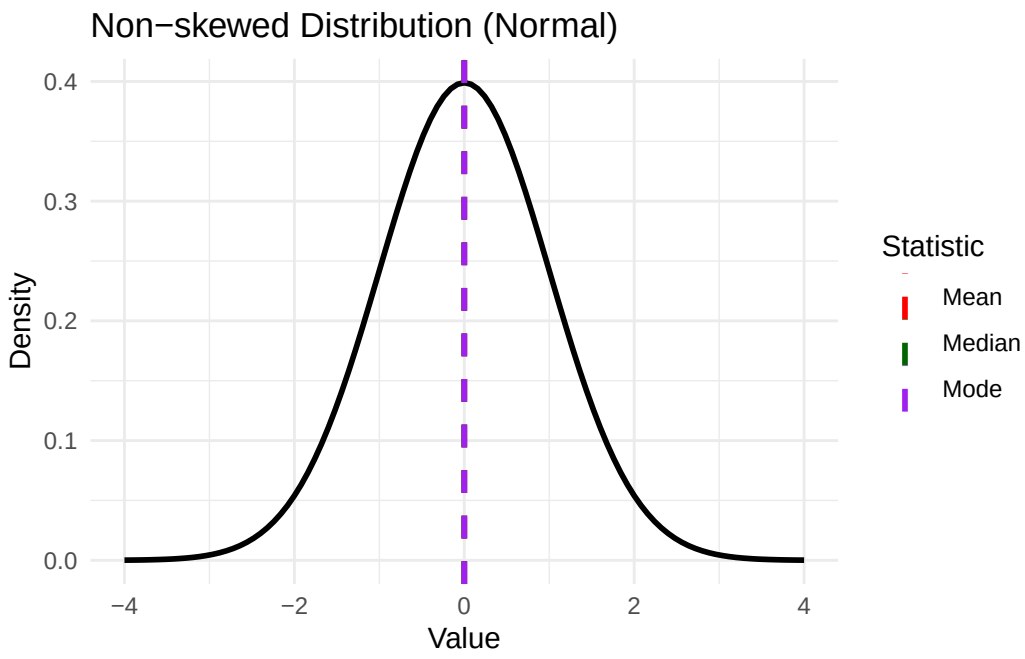
> **Difference between mean, mode and median**
>
> - Mean: the centre of mass of the distribution
> - Mode: The most probable value of Y
> - Median: The value of Y in the middle of the distribution.

In skewed distribution, the three will not equal to each other.

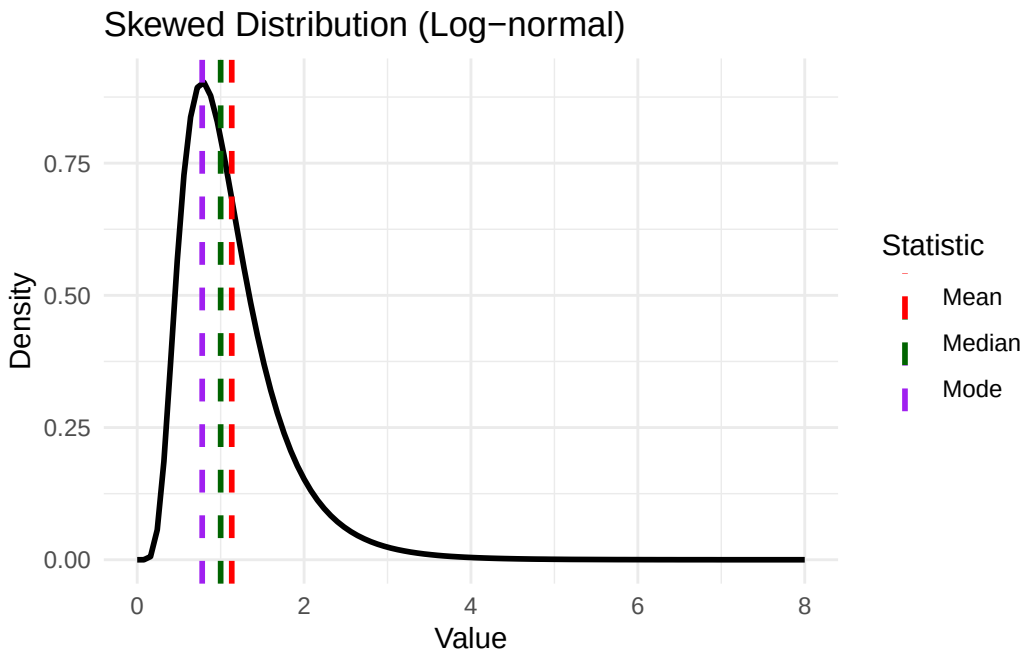
```
library(ggplot2)

# ----------------------------
# Theoretical reference lines
# ----------------------------
lines_normal <- data.frame(
  value = c(0, 0, 0),
  Statistic = c("Mean", "Median", "Mode")
)

lines_lognormal <- data.frame(
  value = c(exp(1/8), 1, exp(-1/4)),
  Statistic = c("Mean", "Median", "Mode")
)

cols <- c("Mean" = "red", "Median" = "darkgreen", "Mode" = "purple")
```

```r
# ----------------------------
# Normal distribution
# ----------------------------
p1 <- ggplot() +
  stat_function(fun = dnorm, size = 1, color = "black") +
  geom_vline(
    data = lines_normal,
    aes(xintercept = value, color = Statistic),
    linetype = "dashed",
    size = 1
  ) +
  scale_color_manual(values = cols) +
  scale_x_continuous(limits = c(-4, 4)) +
  labs(
    title = "Non-skewed Distribution (Normal)",
    x = "Value", y = "Density", color = "Statistic"
  ) +
  theme_minimal()

# ----------------------------
# Log-normal distribution: LN(0, 0.5)
# ----------------------------
p2 <- ggplot() +
  stat_function(
    fun = function(x) dlnorm(x, meanlog = 0, sdlog = 0.5),
    size = 1,
    color = "black"
  ) +
  geom_vline(
    data = lines_lognormal,
    aes(xintercept = value, color = Statistic),
    linetype = "dashed",
    size = 1
  ) +
  scale_color_manual(values = cols) +
  scale_x_continuous(limits = c(0, 8)) +
  labs(
    title = "Skewed Distribution (Log-normal)",
    x = "Value", y = "Density", color = "Statistic"
  ) +
  theme_minimal()
```

```
p1
```


Non−skewed Distribution (Normal)
Density
0.4
0.3
0.2
0.1
0.0
−4
−2
0
2
4
Value
Statistic
Mean
Median
Mode

```
p2
```


Skewed Distribution (Log−normal)
Density
0.75
0.50
0.25
0.00
0
2
4
6
8
Value
Statistic
Mean
Median
Mode

Why use mean?

The mean is widely used in statistics and data analysis for several reasons:

1. Mathematical properties: The mean has desirable mathematical properties, such as linearity, which makes it easier to work with in various statistical analyses and models.
2. Sensitivity to all values: The mean takes into account all values in the dataset, providing a comprehensive measure of central tendency. It is also a scaled version of the total, which is often an interest
3. Foundation for other statistical measures: The mean serves as the basis for many other statistical measures, such as variance and standard deviation, which are essential for understanding the spread and variability of data.
4. Mean minimizes the sum of squared deviations: The mean is the value that minimizes the sum of squared deviations (i.e., the expected penalty by choosing one value) from itself, making it a natural choice for summarizing data.
5. May contains full information: In some distributions (e.g., bernoulli distribution), the mean contains all the information about the distribution, making it a sufficient statistic for inference.

The **variance** of a random variable Y measures the spread or dispersion of the distribution, and is defined as

$$\mathrm{Var}(Y) = E\left[(Y - E(Y))^2\right] = E[Y^2] - E^2[Y].$$

The standard deviation is the square root of the variance, denoted as $\mathrm{SD}(Y) = \sqrt{\mathrm{Var}(Y)}$.

The **quantile** of order α of a random variable Y is defined as the value y_α such that

$$\Pr(Y \leq y_\alpha) = F(y_\alpha) = \alpha$$

for $0 < \alpha < 1$.

For example, the median is the quantile of order 0.5, denoted as $y_{0.5}$, which satisfies $\Pr(Y \leq y_{0.5}) = 0.5$. Also, $(y_{0.025}, y_{0.975})$ and $(y_{0.25}, y_{0.75})$ contains 95% and 50% of the mass of the distribution, respectively.

2.5 Joint Disitrubiton

2.5.1 Discrete random variables

Let Y_1 and Y_2 be two random variables with possible values in $\mathcal{Y}_1$ and $\mathcal{Y}_2$, respectively. The **joint distribution** of Y_1 and Y_2 describes the probability of various combinations of values

that (Y_1, Y_2) can take.

Joint beliefs about Y_1 and Y_2 can be represented with probabilities. For example, for subsets $A \subset \mathcal{Y}_1$ and $B \subset \mathcal{Y}_2$, $\Pr(\{Y_1 \in A\} \cap \{Y_2 \in B\})$ represents our belief that Y_1 takes a value in A and Y_2 takes a value in B. The *joint pdf* or *joint density* of Y_1 and Y_2 is defined as

$$p_{Y_1Y_2}(y_1, y_2) = \Pr(\{Y_1 = y_1\} \cap \{Y_2 = y_2\}), \text{ for } y_1 \in \mathcal{Y}_1, y_2 \in \mathcal{Y}_2.$$

The *marginal density* of Y_1 can be computed from the joint density:

$$\begin{aligned} p_{Y_1}(y_1) &\equiv \Pr(Y_1 = y_1) \\ &= \sum_{y_2 \in \mathcal{Y}_2} \Pr(\{Y_1 = y_1\} \cap \{Y_2 = y_2\}) \\ &\equiv \sum_{y_2 \in \mathcal{Y}_2} p_{Y_1Y_2}(y_1, y_2) \end{aligned}$$

The *conditional density* of Y_2 given $\{Y_1 = y_1\}$ can be computed from the joint density and the marginal density:

$$\begin{aligned} p_{Y_2|Y_1}(y_2 \mid y_1) &= \frac{\Pr(\{Y_1 = y_1\} \cap \{Y_2 = y_2\})}{\Pr(Y_1 = y_1)} \\ &= \frac{p_{Y_1Y_2}(y_1, y_2)}{p_{Y_1}(y_1)}. \end{aligned}$$

You should be able to see that

- $\left\{p_{Y_1}, p_{Y_2|Y_1}\right\}$ can be derived from $p_{Y_1Y_2}$,
- $\left\{p_{Y_2}, p_{Y_1|Y_2}\right\}$ can be derived from $p_{Y_1Y_2}$
- $p_{Y_1Y_2}$ can be derived from $\left\{p_{Y_1}, p_{Y_2|Y_1}\right\}$
- $p_{Y_1Y_2}$ can be derived from $\left\{p_{Y_2}, p_{Y_1|Y_2}\right\}$

BUT

- $p_{Y_1Y_2}$ cannot be derived from $\left\{p_{Y_1}, p_{Y_2}\right\}$.

The subscripts of density functions are often dropped, in which case the type of density function is determined by the arguments. For example,

- $p(y_1, y_2) = p_{Y_1Y_2}(y_1, y_2)$ is the joint density of Y_1 and Y_2,
- $p(y_1) = p_{Y_1}(y_1)$ is the marginal density of Y_1
- $p(y_2 \mid y_1) = p_{Y_2|Y_1}(y_2 \mid y_1)$ is the conditional density of Y_2 given $\{Y_1 = y_1\}$, and so on.

Suppose a sociological study reports the following joint distribution of parents' education level and children's income level in a population.

Joint distribution of education and income Suppose a sociological study reports the following **joint distribution** of **parents' education level** and **children's income level** in a population as shown in the Table below

Parent \ Child	Low Income	Middle Income	High Income
High School or Less	0.18	0.22	0.10
College	0.08	0.20	0.12
Graduate School	0.04	0.06	0.10

Suppose we randomly sample a **parent–child pair** from this population.

Let
- Y_1 be the parent's education level
- Y_2 be the child's income level

We are interested in the conditional probability that the child has **high income**, given that the parent has a **college education**.

We may answer this question using the conditional probability formula:

$$\Pr(Y_2 = \text{High Income} \mid Y_1 = \text{College}) = \frac{\Pr(Y_2 = \text{High Income} \cap Y_1 = \text{College})}{\Pr(Y_1 = \text{College})}$$

From the table,

$$\Pr(Y_2 = \text{High Income} \cap Y_1 = \text{College}) = 0.12$$

$$\Pr(Y_1 = \text{College}) = 0.08 + 0.20 + 0.12 = 0.40$$

Therefore,

$$\Pr(Y_2 = \text{High Income} \mid Y_1 = \text{College}) = \frac{0.12}{0.40} = 0.30$$

Thus, our conclusion from the table is, among children whose parents have a college education, **30%** attain high income.

2.5.2 Continuous random variables

Let Y_1 and Y_2 be two continuous random variables with possible values in $\mathcal{Y}_1$ and $\mathcal{Y}_2$, respectively. The **joint distribution** of Y_1 and Y_2 describes the probability of various combinations of values that (Y_1, Y_2) can take. We again work with the cumulative distribution function (cdf). The definition is given as follows.

Given a continuous joint cdf $F_{Y_1Y_2}(y_1, y_2)$, there is a function p_{Y_1,Y_2} such that

$$F_{Y_1,Y_2}(a, b) = \int_{-\infty}^{a} \int_{-\infty}^{b} p_{Y_1,Y_2}(y_1, y_2) dy_2 dy_1,$$

and $p_{Y_1,Y_2}(y_1, y_2)$ is called the *joint density function* of Y_1 and Y_2.

Similar to the discrete case, we can derive marginal and conditional densities from the joint density as

- Marginal density of Y_1:
$$p_{Y_1}(y_1) = \int_{y_2} p_{Y_1,Y_2}(y_1, y_2) dy_2,$$
- Conditional density of Y_2 given $\{Y_1 = y_1\}$:
$$p_{Y_2|Y_1}(y_2 \mid y_1) = \frac{p_{Y_1,Y_2}(y_1, y_2)}{p_{Y_1}(y_1)}.$$

Think about why $p_{Y_2|Y_1}(y_2 \mid y_1)$ is an actual pdf.

2.5.3 Mixed continuous and discrete variables

It is possible to have joint distributions involving both discrete and continuous random variables. For example, let Y_1 be a discrete random variable taking values in $\mathcal{Y}_1$ and Y_2 be a continuous random variable taking values in $\mathcal{Y}_2$. The joint distribution of Y_1 and Y_2 can be described by the joint density function $p_{Y_1,Y_2}(y_1, y_2)$, which gives the probability that Y_1 takes the value y_1 and Y_2 takes a value in an infinitesimal interval around y_2. One such as example is that Y_1 is a binary variable indicating the presence or absence of a disease, and Y_2 is a continuous variable representing the severity of symptoms. Suppose we define

- Marginal density p_{Y_1} from our belief $\Pr(Y_1 = y_1)$
- a conditional density $p_{Y_2|Y_1}$ from $\Pr(Y_2 \leq y_2 \mid Y_1 = y_1) \doteq F_{Y_2|Y_1}(y_2 \mid y_1)$.

Then, the joint density can be derived as

$$p_{Y_1,Y_2}(y_1, y_2) = p_{Y_1}(y_1)p_{Y_2|Y_1}(y_2 \mid y_1),$$

and the probability can be calculated as

$$\Pr(Y_1 \in A, Y_2 \in B) = \int_{y_2 \in B} \left\{ \sum_{y_1 \in A} p_{Y_1,Y_2}(y_1, y_2) \right\} dy_2.$$

2.5.4 Bayes' rule and parameter estimation

Let

- θ: proportion of people in a large population who have a certain charactersitic.
- Y: number of people in a small random sample from the population who have the charactersitic

Then, in this case, we may threat θ as continuous random variable taking values in $\Theta = (0, 1)$, and Y as a discrete random variable taking values in $\mathcal{Y} = \{0, 1, 2, ..., n\}$, where n is the sample size. *Bayesian estimation of the parameter* θ derives from the calculate of $p(\theta \mid y)$ where y is the observed value of Y. In Bayesian, this calculation first requires that we have a joint density $p(y, \theta)$ representing our belief about θ and the survey outcome Y. Often, it is natural to construct this joint density from

- $p(\theta)$: our prior belief about θ before seeing the data, and
- $p(y \mid \theta)$: belief about Y given θ, often called the likelihood function.

Once we observed $\{Y = y\}$, we need to compute our updated belief about θ, represented by the **posterior density** $p(\theta \mid y)$ as

$$p(\theta \mid y) = \frac{p(\theta, y)}{p(y)} = \frac{p(y \mid \theta)p(\theta)}{p(y)} = \frac{p(y \mid \theta)p(\theta)}{\int_\Theta p(y \mid \theta)p(\theta)d\theta}.$$

If we have two values θ_1 and θ_2 in Θ that may be true, then the ratio of their posterior densities is given by

$$\frac{p(\theta_1 \mid y)}{p(\theta_2 \mid y)} = \frac{p(y \mid \theta_1)p(\theta_1)/p(y)}{p(y \mid \theta_2)p(\theta_2)/p(y)} = \frac{p(y \mid \theta_1)p(\theta_1)}{p(y \mid \theta_2)p(\theta_2)}.$$

> **Note**
>
> From this calculation, we notice when we are calculating the relative posterior probability between two parameter values **we do not need** calculate $p(y)$ out.

Another way to think about this is, for a function of θ,

$$p(\theta \mid y) \propto p(y \mid \theta)p(\theta).$$

> **Note**
>
> We will see that the numerator is the important part, while the denominator is just a normalizing constant to make sure the posterior density integrates to 1.

2.6 Independence Random Variables

Let $Y_1, \dots, Y_n$ be random variables with joint density $p(y_1, \dots, y_n)$, and θ is the parameter describe the conditions under which the random variables are generated. We say that $Y_1, \dots, Y_n$ are conditionally independent given θ if every collection of n sets $\{A_1, \dots, A_n\}$ satisfies

$$\Pr(Y_1 \in A_1, \dots, Y_n \in A_n \mid \theta) = \prod_{i=1}^{n} \Pr(Y_i \in A_i \mid \theta).$$

If we have independence property, then

$$\Pr(Y_i \in A_i \mid \theta, Y_j \in A_j) = \Pr(Y_i \in A_i \mid \theta),$$

so the conditional indpenddence can be interpreted as meaning that Y_j gives no additional information about Y_i once we know θ. Also, under independence, the joint density can be factorized as

$$p(y_1, \dots, y_n \mid \theta) = \prod_{i=1}^{n} p_{Y_i}(y_i \mid \theta).$$

If the samples are also identically distributed, meaning that each Y_i has the same marginal density $p_Y(y \mid \theta)$, then the joint density can be further simplified as

$$p(y_1, \dots, y_n \mid \theta) = \prod_{i=1}^{n} p_Y(y_i \mid \theta).$$

In this case , we say that $Y_1, \dots, Y_n$ are **independent and identically distributed** (i.i.d.) given θ, with notation

$$Y_1, \dots, Y_n \mid \theta \overset{i.i.d.}{\sim} p_Y(y \mid \theta).$$

2.7 Exchangeability

A sequence of random variables $Y_1, Y_2, \ldots, Y_n$ is **exchangeable** if for any permutation π of the indices $\{1, 2, \ldots, n\}$, we have

$$p(y_1, y_2, \ldots, y_n) = p(y_{\pi(1)}, y_{\pi(2)}, \cdots, y_{\pi(n)}).$$

In other words, the joint density of an exchangeable sequence is invariant to the order of the random variables. That is, the labels contains no information about the outcome.

Suppose a factory produces a large batch of items. Each item may be either **defective** or **non-defective**.

Let

$$Y_i = \begin{cases} 1, & \text{if the } i\text{th inspected item is defective,} \\ 0, & \text{otherwise.} \end{cases}$$

We inspect $n = 10$ items chosen at random from the batch and record $Y_1, Y_2, \ldots, Y_{10}$.

Consider the following three observed sequences:

1. $p(1, 0, 1, 0, 1, 0, 0, 1, 0, 1)$
2. $p(0, 1, 0, 1, 0, 1, 1, 0, 0, 1)$
3. $p(1, 1, 0, 0, 1, 0, 1, 0, 0, 1)$

Each sequence contains **5 defective items** and **5 non-defective items**.

Question: Is there a reason to assign these three sequences *different probabilities*?

If the inspection order conveys no additional information about quality, then *only the number of defective items matters*, not their positions in the sequence. This motivates the concept of exchangeability.

2.7.1 Independence versus dependence

Consider the probability assignments

$$\begin{cases} \Pr(Y_{10} = 1) = a, \\ \Pr(Y_{10} = 1 \mid Y_1 = \cdots = Y_9 = 1) = b. \end{cases}$$

If $a \neq b$, then Y_{10} is **not independent** of $Y_1, \ldots, Y_9$.

However, lack of independence does **not** imply lack of exchangeability.

Question: should we have $a = b$, $a > b$ or $a < b$?

2.7.2 A latent-parameter model

Suppose the defect rate θ of the factory is unknown.

Conditional on θ,

$$Y_1, \dots, Y_{10} \mid \theta \sim \text{i.i.d. Bernoulli}(\theta).$$

Then

$$\Pr(Y_1 = y_1, \dots, Y_{10} = y_{10} \mid \theta) = \theta^{\sum y_i}(1-\theta)^{10-\sum y_i}.$$

If our uncertainty about θ is described by a prior distribution $p(\theta)$, the marginal joint distribution is

$$p(y_1, \dots, y_{10}) = \int \theta^{\sum y_i}(1-\theta)^{10-\sum y_i} p(\theta)\, d\theta.$$

This probability depends **only on the number of defective items**, not their order.

Thus, we have exchangeability, even though the Y_i are not independent under this model of belief.

Conditional i.i.d. given a latent parameter implies marginal exchangeability. That is, if $\theta \sim p(\theta)$ and $Y_1, \dots, Y_n$ are conditionally i.i.d. given θ, then $Y_1, \dots, Y_n$ (i.e., unconditional on θ) are exchangeable.

For the Proof, see page 28 in Hopf (2009).

2.8 de Finetti's Theorem

As of now, we have seen that conditional i.i.d. given a latent parameter implies marginal exchangeability. For example,

$$\begin{cases} Y_1, \dots, Y_n \mid \theta \overset{i.i.d.}{\sim} \\ \theta \sim p(\theta) \end{cases} \implies Y_1, \dots, Y_n \text{ are exchangeable.}$$

The converse is also true, as stated in de Finetti's theorem.

Let $Y_i \in \mathcal{Y}$ for all $i \in \{1, 2, \dots, n\}$ be an exchangeable sequence of random variables. Then, there exists a parameter space Θ and a prior distribution $p(\theta)$ on Θ such that the joint distribution of $Y_1, \dots, Y_n$ can be represented as

$$p(y_1, \dots, y_n) = \int_\Theta \left\{ \prod_{i=1}^{n} p_Y(y_i \mid \theta) \right\} p(\theta) d\theta,$$

where $p_Y(y \mid \theta)$ is a probability density function on $\mathcal{Y}$ for each $\theta \in \Theta$. The prior and sampling model depend on the form of the belief model $p(y_1, \dots, y_n)$.

The probability distribution $p(\theta)$ represents our belief about the outcomes $\{Y_1, Y_2, \dots, Y_n\}$, induced by our belief model $p(y_1, \dots, y_n)$. That is,

- $p(\theta)$ represents our belief about $\lim_{n\to\infty} \sum Y_i/n$ in the binary sense
- $p(\theta)$ represents our belief about $\lim_{n\to\infty} \sum(Y_i \le c)/n$ for each c in the general case.

The main idea of this and the previous section is as follows

$$Y_1, \dots, Y_n \mid \theta \overset{\text{i.i.d.}}{\sim} p(\cdot \mid \theta), \quad \theta \sim p(\theta) \iff Y_1, \dots, Y_n \text{ are exchangeable for all } n.$$

Question: When is the condition of "exchangeability for all n" reasonable?

- Have exchaneability and repeatability
  - Exchangeability holds if the labels convey no information
  - repeatability hold includes the follows
    1. $Y_1, \dots, Y_n$ are outcomes of a repeartable experiment
    2. $Y_1, \dots, Y_n$ are sampled from a finite population **with replacement**
    3. $Y_1, \dots, Y_n$ are sampled from an infinite population without replacement.

> **i** In large finite population
>
> Note, if $Y_1, \dots, Y_n$ are exchangeable and sampled from a finite population of size N that is way bigger than n without replacement, then they can be modelled as *approximate* being conditional i.i.d.

This Chapter follows closely with Chapter 2 in Hoff (2009).

3 Bayesian Inference for single parameter models

Leading objectives:

Understand how to perform Bayesian inference on a single parameter model.

- Binomial model with given n
- Poission model
- Exponential family

Recall the important ingredients of Bayesian inference:

1. **Prior distribution:** $\pi(\theta)$
2. **Likelihood function:** $p(y \mid \theta)$
3. **Posterior distribution:** $p(\theta \mid y) \propto p(y \mid \theta)\pi(\theta)$

3.1 Three basic ingredients of Bayesian inference

3.1.1 Prior

The prior distribution encodes our beliefs about the parameter θ *before* conduct any experiments.

> **Prior and Data are independent**
>
> Note that, the prior distribution is independent of the data. It represents our knowledge or beliefs about the parameter before seeing the data.

How do we choose a prior?

1. **Informative priors:** Based on previous studies or expert knowledge
2. **Weakly informative priors:** Provide some regularization without dominating the data
3. **Non-informative priors:** Attempt to be "objective" (e.g., uniform, Jeffreys prior)

3.1.2 Likelihood

The likelihood function represents the probability of observing the data given the parameter θ. It can be derived from the assumed statistical model for the data or experiment, i.e., $y \sim p(y \mid \theta)$, or we can estimate this non-parametrically (i.e., without assuming the underlying distribution is the one we know.).

> **Likelihood is NOT a probability distribution for θ**
>
> Note that, the likelihood function is not a probability distribution for θ itself. It is a function of θ for fixed data y.

3.1.3 Posterior

The posterior distribution combines the prior and likelihood to update our beliefs about θ after observing the data. It is given by Bayes' theorem:

$$p(\theta \mid y) = \frac{p(y \mid \theta)\pi(\theta)}{p(y)},$$

where $p(y) = \int p(y \mid \theta)\pi(\theta)d\theta$ is the marginal likelihood or evidence.

3.1.4 An simple example

Examples:

- Beta prior + Binomial likelihood → Beta posterior
- Normal prior + Normal likelihood (known variance) → Normal posterior
- Gamma prior + Poisson likelihood → Gamma posterior

Advantages: - Analytical posteriors (no numerical integration needed) - Interpretable parameters - Computationally efficient

Limitations:

- May not reflect true prior beliefs
- Modern computing makes non-conjugate priors feasible

Let's look a simple example to illustrate the convenience of conjugate priors. Consider a Binomial model with unknown success probability θ and known number of trials n. We can use a Beta prior for θ.

Suppose we have a Binomial model with known number of trials n and unknown success probability θ. We can use a Beta prior for θ.

- **Prior:** $\theta \sim \text{Beta}(\alpha, \beta)$
- **Likelihood:** $y \mid \theta \sim \text{Binomial}(n, \theta)$

The derivation of the posterior is as follows:

$$p(y \mid \theta) = \binom{n}{y} \theta^y (1-\theta)^{n-y},$$
$$\pi(\theta) = \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{B(\alpha, \beta)},$$

where $B(\alpha, \beta)$ is the Beta function. Then the posterior is proportional to:

$$p(\theta \mid y) \propto p(y \mid \theta)\pi(\theta) \propto \theta^{y+\alpha-1}(1-\theta)^{n-y+\beta-1}.$$

This is the kernel of a Beta distribution with parameters $(\alpha + y, \beta + n - y)$. Thus, the posterior distribution is:

$$\theta \mid y \sim \text{Beta}(\alpha + y, \beta + n - y).$$

Thus, the **Posterior** is $\theta \mid y \sim \text{Beta}(\alpha + y, \beta + n - y)$.

3.2 Happiness Data – the first example of Bayesian inference procedure

We study Bayesian inference for a binomial proportion θ when the sample size n is fixed. In this example, we want to see what is the procedure of doing Bayesian inference

In the 1998 General Social Survey, each female respondent aged 65 or over was asked whether she was generally happy.

Define the response variable

$$Y_i = \begin{cases} 1, & \text{if respondent } i \text{ reports being generally happy,} \\ 0, & \text{otherwise,} \end{cases} \qquad i = 1, \ldots, n,$$

where $n = 129$.

Because we lack information that distinguishes individuals, it is reasonable to treat the responses as **exchangeable**.
That is, before observing the data, the labels or ordering of respondents carry no information.

Since the sample size n is small relative to the population size N of senior women, results from the previous chapter justify the following modeling approximation.

Modeling Assumptions: Our beliefs about $(Y_1, \ldots, Y_{129})$ are described by:

- **An unknown population proportion**

$$\theta = \frac{1}{N}\sum_{i=1}^{N} Y_i,$$

where θ represents the proportion of generally happy individuals in the population.

- **A sampling model given θ**

Conditional on θ, the responses $Y_1, \dots, Y_{129}$ are independent and identically distributed Bernoulli random variables with

$$\Pr(Y_i = 1 \mid \theta) = \theta.$$

Given the population proportion θ, each respondent independently reports being happy with probability θ.

Likelihood: Under this model, the probability of observing data $\{y_1, \dots, y_{129}\}$ given θ is

$$p(y_1, \dots, y_{129} \mid \theta) = \theta^{\sum_{i=1}^{129} y_i}(1-\theta)^{129-\sum_{i=1}^{129} y_i}.$$

This expression depends on the data only through the sufficient statistic

$$S = \sum_{i=1}^{129} Y_i,$$

the total number of respondents who report being generally happy.

For the happiness data,

$$S = 118,$$

so the likelihood simplifies to

$$p(y_1, \dots, y_{129} \mid \theta) = \theta^{118}(1-\theta)^{11}.$$

Q: Which prior to be used?

A prior distribution is **conjugate** to a likelihood if the posterior distribution belongs to the same family as the prior. For the binomial likelihood, the **Beta distribution** is conjugate. But we have another choice of prior, to use *non-informative prior*.

A Uniform Prior Distribution: Suppose our prior information about θ is very weak, in the sense that all subintervals of $[0, 1]$ with equal length are equally plausible.
Symbolically, for any $0 \le a < b < b + c \le 1$,

$$\Pr(a \le \theta \le b) = \Pr(a + c \le \theta \le b + c).$$

This implies a **uniform prior**:

$$\pi(\theta) = 1, \qquad 0 \leq \theta \leq 1.$$

Posterior Distribution: Bayes' rule gives

$$p(\theta \mid y_1, \dots, y_{129}) = \frac{p(y_1, \dots, y_{129} \mid \theta)\, \pi(\theta)}{p(y_1, \dots, y_{129})}.$$

With a uniform prior, this reduces to

$$p(\theta \mid y_1, \dots, y_{129}) \propto \theta^{118}(1-\theta)^{11}.$$

Key idea: with a uniform prior, the posterior has the **same shape** as the likelihood.

To obtain a proper probability distribution, we must normalize.

Normalizing Constant and the Beta Distribution: Using the identity

$$\int_0^1 \theta^{a-1}(1-\theta)^{b-1}\, d\theta = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)},$$

we find

$$p(y_1, \dots, y_{129}) = \frac{\Gamma(119)\Gamma(12)}{\Gamma(131)}.$$

Therefore, the posterior density is

$$p(\theta \mid y_1, \dots, y_{129}) = \frac{\Gamma(131)}{\Gamma(119)\Gamma(12)}\theta^{119-1}(1-\theta)^{12-1}.$$

That is,

$$\theta \mid y \sim \text{Beta}(119,\, 12).$$

Recall that, a random variable $\theta \sim \text{Beta}(a, b)$ distribution if

$$\pi(\theta) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\theta^{a-1}(1-\theta)^{b-1}.$$

For $\theta \sim \text{Beta}(a, b)$, the expectation (i.e., mean or the first moment) is $\mathbb{E}(\theta) = \frac{a}{a+b}$, and the variance is $\text{Var}(\theta) = \frac{ab}{(a+b)^2(a+b+1)}$.

In our example, the happiness data, the posterior distribution is

$$\theta \mid y \sim \text{Beta}(119, 12).$$

Thus, the posterior mean is $\mathbb{E}(\theta \mid y) = 0.915$, and the posterior standard deviation is $\text{sd}(\theta \mid y) = 0.025$.

These summaries quantify both our **best estimate** of the population proportion and our **remaining uncertainty** after observing the data.

3.2.1 Inference about exchangeable binary data

Posterior Inference under a Uniform Prior

Suppose $Y_1, \ldots, Y_n \mid \theta \overset{\text{i.i.d.}}{\sim} \text{Bernoulli}(\theta)$, and we place a uniform prior on θ. The posterior distribution of θ given the observed data $y_1, \ldots, y_n$ is proportional to

$$\begin{aligned} p(\theta \mid y_1, \ldots, y_n) &= \frac{p(y_1, \ldots, y_n \mid \theta)\pi(\theta)}{p(y_1, \ldots, y_n)} \\ &= \theta^{\sum_i y_i}(1-\theta)^{n-\sum_i y_i} \times \frac{\pi(\theta)}{p(y_1, \ldots, y_n)} \\ &\propto \theta^{\sum_i y_i}(1-\theta)^{n-\sum_i y_i}. \end{aligned}$$

Consider two parameter values θ_a and θ_b. The ratio of their posterior densities is

$$\begin{aligned} \frac{p(\theta_a \mid y_1, \ldots, y_n)}{p(\theta_b \mid y_1, \ldots, y_n)} &= \frac{\theta_a^{\sum y_i}\left(1-\theta_a\right)^{n-\sum y_i} \times p\left(\theta_a\right) / p\left(y_1, \ldots, y_n\right)}{\theta_b^{\sum y_i}\left(1-\theta_b\right)^{n-\sum y_i} \times p\left(\theta_b\right) / p\left(y_1, \ldots, y_n\right)} \\ &= \left(\frac{\theta_a}{\theta_b}\right)^{\sum_i y_i}\left(\frac{1-\theta_a}{1-\theta_b}\right)^{n-\sum_i y_i} \frac{p(\theta_a)}{p(\theta_b)}. \end{aligned}$$

This expression shows that the data affect the posterior distribution **only through the sum of the data** $\sum_{i=1}^n y_i$ based on the relative probability density at θ_a to θ_b.

As a result, for any set A, one can show that

$$\Pr(\theta \in A \mid Y_1 = y_1, \ldots, Y_n = y_n) = \Pr\left(\theta \in A \mid \sum_{i=1}^n Y_i = \sum_{i=1}^n y_i\right).$$

This means that $\sum_{i=1}^n Y_i$ contains **all the information** in the data relevant for inference about θ. We therefore say that $Y = \sum_{i=1}^n Y_i$ is a **sufficient statistic** for θ. The term *sufficient* is used because knowing $\sum_{i=1}^n Y_i$ is sufficient to carry out inference about θ; no additional information from the individual observations $Y_1, \ldots, Y_n$ is required.

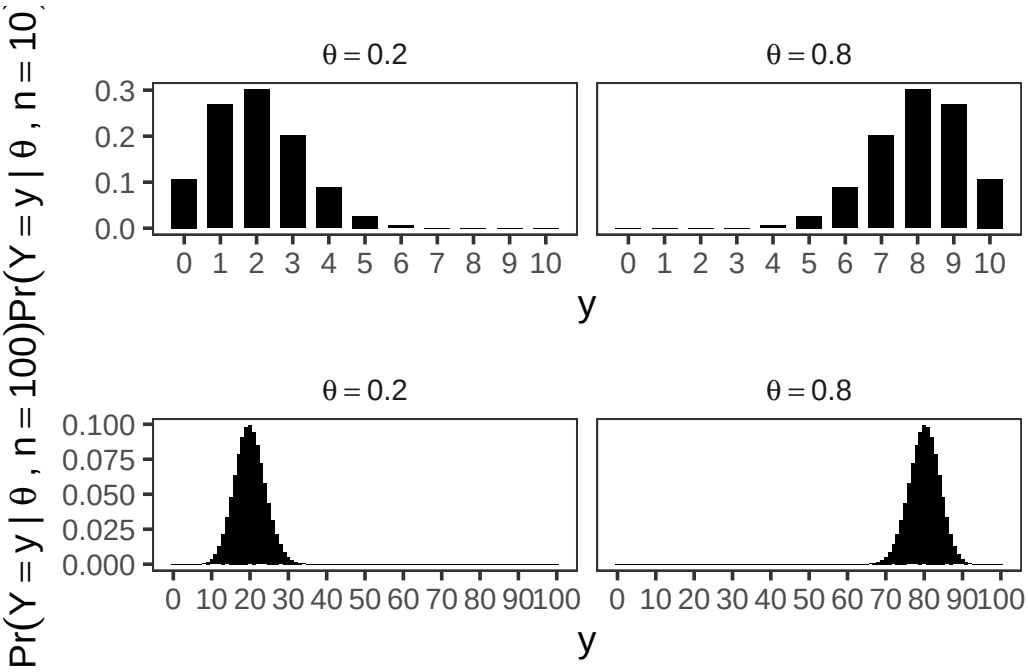
In the case where $Y_1, \ldots, Y_n \mid \theta$ are i.i.d. Bernoulli(θ) random variables, the sufficient statistic $Y = \sum_{i=1}^n Y_i$ follows a **binomial distribution** with parameters (n, θ).

The Binomial Model

Because each Y_i is Bernoulli(θ) and the observations are independent, the sufficient statistic $Y = \sum_{i=1}^n Y_i$ follows a **binomial distribution** with parameters (n, θ).

That is, $\Pr(Y = y \mid \theta) = \binom{n}{y}\theta^y(1-\theta)^{n-y}$, $y = 0, 1, \ldots, n$. For a binomial(n, θ) random variable Y,

- $\mathbb{E}[Y \mid \theta] = n\theta$,
- $\mathrm{Var}(Y \mid \theta) = n\theta(1-\theta)$.



Posterior inference under a uniform prior distribution

Having observed $Y = y$ our task is to obtain the posterior distribution of θ. By Bayes' theorem,

$$p(\theta \mid y) = \frac{p(y \mid \theta), \pi(\theta)}{p(y)}.$$

For a binomial model with $Y \sim \text{Binomial}(n, \theta)$, the likelihood is

$$p(y \mid \theta) = \binom{n}{y} \theta^y (1-\theta)^{n-y}.$$

Therefore,

$$p(\theta \mid y) = \frac{\binom{n}{y}\theta^y(1-\theta)^{n-y}\pi(\theta)}{p(y)} = c(y)\theta^y(1-\theta)^{n-y}\pi(\theta),$$

where $c(y)$ is a normalizing constant that depends only on y, not on θ. When using the uniform

distribution, $\pi(\theta)$, we can calculate $c(y)$ easily as

$$\begin{aligned} 1 &= \int_0^1 c(y)\theta^y(1-\theta)^{n-y}d\theta \\ &= c(y)\int_0^1 \theta^y(1-\theta)^{n-y}d\theta \quad . \\ &= c(y)\frac{\Gamma(y+1)\Gamma(n-y+1)}{\Gamma(n+2)} \end{aligned}$$

Hence, $c(y) = \Gamma(n+2)/\{\Gamma(y+1)\Gamma(n-y+1)\}$, and the posterior distribution is

$$\begin{aligned} p(\theta \mid y) &= \frac{\Gamma(n+2)}{\Gamma(y+1)\Gamma(n-y+1)}\theta^y(1-\theta)^{n-y} \\ &= \frac{\Gamma(n+2)}{\Gamma(y+1)\Gamma(n-y+1)}\theta^{(y+1)-1}(1-\theta)^{(n-y+1)-1}, \end{aligned}$$

Which is exactly the beta$(y+1, n-y+1)$. In the happiness example, we have $n = 129$ and $Y = \sum Y_i = 118$, so the posterior distribution is beta$(119, 12)$, written as

$$n = 129, Y \equiv \sum Y_i = 118 \quad \Rightarrow \quad \theta \mid \{Y = 118\} \sim \text{beta}(119, 12).$$

This confirms the sufficiency result for this model and prior distribution, by showing that if $\sum y_i = y = 118$, $p(\theta \mid y_1, \ldots y_n) = p(\theta \mid y) = \text{beta}(119, 12)$. That is, the information contained in $\{Y_1 = y_1, \ldots, Y_n = y_n\}$ is the same as the information contained in $\{Y = y\}$, where $Y = \sum Y_i$ and $y = \sum y_i$. This show the posterior when we use **uniform prior**. One may ask, what if we use a different prior?

Posterior distributions under beta prior distributions

The uniform prior distribution has $\pi(\theta) = 1$ for all $\theta \in [0, 1]$. This distribution can be thought of as a beta prior distribution with parameters $a = 1, b = 1$

$$\pi(\theta) = \frac{\Gamma(2)}{\Gamma(1)\Gamma(1)}\theta^{1-1}(1-\theta)^{1-1} = \frac{1}{1\times 1}1 \times 1 = 1$$

for all $\theta \in [0, 1]$.

The gamma function is defined as

$$\Gamma(x) = \int_0^\infty t^{x-1}e^{-t}dt, \quad x > 0.$$

It satisfies the following properties:

- $\Gamma(n) = (n-1)!$ for any positive integer n.
- $\Gamma(x+1) = x\Gamma(x)$ for any $x > 0$.

- $\Gamma(1/2) = \sqrt{\pi}$.
- $\Gamma(1) = 1$ by convention.

Now, from the previous part, recall that we have,

$$\text{if } \left\{ \begin{array}{c} \theta \sim \text{beta}(1,1) \text{ (uniform)} \\ Y \sim \text{binomial}(n, \theta) \end{array} \right\}, \text{ then } \{\theta \mid Y = y\} \sim \text{beta}(1 + y, 1 + n - y).$$

To get the posterior distribution under a general beta prior distribution, we just need to add the number of 1's to the α parameter and the number of 0's to the β parameter. To see this, assume $\theta \sim \text{beta}(\alpha, \beta)$, and $Y \mid \theta \sim \text{binomial}(n, \theta)$. Then, once we observed $\{Y = y\}$, by Bayes' theorem, the posterior distribution is

$$\begin{aligned} p(\theta \mid y) &= \frac{\pi(\theta) p(y \mid \theta)}{p(y)} \\ &= \frac{1}{p(y)} \times \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \theta^{a-1}(1-\theta)^{b-1} \times \binom{n}{y} \theta^{y}(1-\theta)^{n-y} \\ &= c(n, y, a, b) \times \theta^{a+y-1}(1-\theta)^{b+n-y-1} \\ &\propto \beta(a + y, b + n - y). \end{aligned}$$

> **One-to-one correspondence between the distribution**
>
> Note that, there is a one-to-one correspondence between the prior distribution parameters and the posterior distribution parameters. Two distributions are said to be the same if
>
> - Their CDFs are the same.
> - Their PDFs are the same.
> - All of their moments are the same. This implies that they are equal if and only if the moment generating function or the probability generating functions are the same.

We have seen the beta-binomial example twice, which is an example of **conjugate prior**, let's definite this formally,

A class $\mathcal{P}$ of prior distribution for θ is said **conjugate** for the likelihood function $p(y \mid \theta)$ if for every prior distribution $\pi(\theta) \in \mathcal{P}$, the corresponding posterior distribution $p(\theta \mid y)$ is also in $\mathcal{P}$, that is

$$\pi(\theta) \in \mathcal{P} \Rightarrow p(\theta \mid y) \in \mathcal{P}.$$

> **Note**
>
> Conjugate priors simplify posterior calculations, but they may not accurately reflect genuine prior beliefs. Still, mixtures of conjugate priors offer substantially greater flexibility while remaining computationally tractable.

If the likelihood $\theta \mid \{Y = y\} \sim beta(a + y, b + n - y)$, recall that

- $\mathrm{E}[\theta \mid y] = \frac{a+y}{a+b+n}$
- $\mathrm{mode}[\theta \mid y] = \frac{a+y-1}{a+b+n-2}$
- $\mathrm{Var}[\theta \mid y] = \frac{\mathrm{E}[\theta|y]\mathrm{E}[1-\theta|y]}{a+b+n+1}$

The posterior mean can be expressed as a weighted average of the prior mean and the maximum likelihood estimate (MLE) of θ:

$$\begin{aligned}
\mathrm{E}[\theta \mid y] &= \frac{a+y}{a+b+n} \\
&= \frac{a+b}{a+b+n} \times \frac{a}{a+b} + \frac{n}{a+b+n} \times \frac{y}{n} \\
&= \frac{a+b}{a+b+n} \times \text{ prior expectation } + \frac{n}{a+b+n} \times \text{ data mean}
\end{aligned}$$

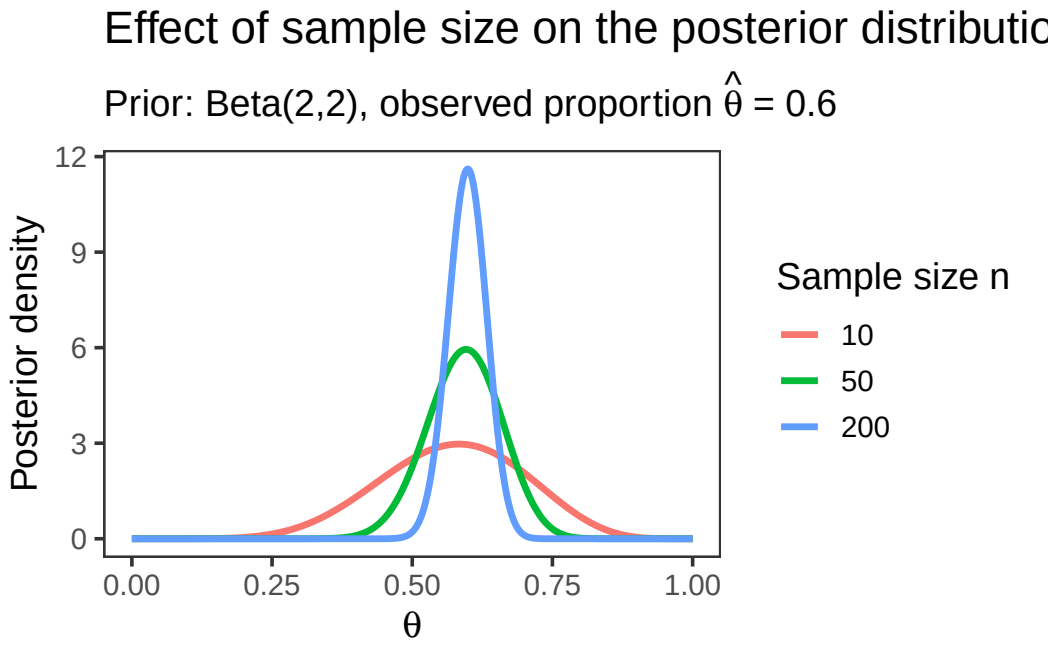
For this model and prior distribution, the posterior expectation (also known as the posterior mean) can be expressed as a weighted average of the prior expectation and the sample mean. The weights are proportional to the prior sample size a + b and the observed sample size n, respectively. This representation leads to a natural interpretation of the Beta prior parameters as prior data:

- $a \approx$ "prior # of 1's,"
- $b \approx$ "prior # of 0's,"
- $a + b \approx$ "prior sample size."

When $n \gg a + b$, it is reasonable to expect that most of the information about θ should come from the data rather than from the prior distribution. This intuition is confirmed mathematically. In particular, when $n \gg a + b$,

- $\frac{a+b}{a+b+n} \approx 0$,
- $\mathbb{E}[\theta \mid y] \approx \frac{y}{n}$,
- $\mathrm{Var}(\theta \mid y) \approx \frac{1}{n} \frac{y}{n} \left(1 - \frac{y}{n}\right)$.

Thus, in large samples, the posterior mean approaches the sample proportion and the posterior variance shrinks at rate $1/n$, reflecting increasing information from the data.



Prediction

An important feature of Bayesian inference is the existence of a **predictive distribution** for new observations.

The posterior predictive distribution for a new observation Y_{new} given the observed data y is obtained by integrating over the posterior distribution of θ.

Returning to our notation for binary data, let $y_1, \ldots, y_n$ be the observed outcomes from a sample of n binary rvs, and let $\tilde{Y} \in \{0, 1\}$ denote a future observation from the same population that has not yet been observed. The **predictive distribution** of $\tilde{Y}$ is defined as the conditional distribution of $\tilde{Y}$ given the observed data $\{Y_1 = y_1, \ldots, Y_n = y_n\}$. For conditionally i.i.d. binary observations, the predictive distribution can be derived by integrating out the unknown parameter θ:

$$\begin{aligned}
\Pr\left(\tilde{Y} = 1 \mid y_1, \ldots, y_n\right) &= \int \Pr\left(\tilde{Y} = 1, \theta \mid y_1, \ldots, y_n\right) d\theta \\
&= \int \Pr\left(\tilde{Y} = 1 \mid \theta, y_1, \ldots, y_n\right) p\left(\theta \mid y_1, \ldots, y_n\right) d\theta \\
&= \int p\left(\theta \mid y_1, \ldots, y_n\right) \theta d\theta \\
&= \mathrm{E}\left[\theta \mid y_1, \ldots, y_n\right] \\
&= \frac{a + \sum_{i=1}^{n} y_i}{a + b + n}.
\end{aligned}$$

Hence, we also have,

$$\Pr\left(\tilde{Y}=0 \mid y_1,\dots,y_n\right) = 1 - \mathrm{E}\left[\theta \mid y_1,\dots,y_n\right] = \frac{b+\sum_{i=1}^{n}(1-y_i)}{a+b+n}.$$

> **i** Properties of the predictive distribution
>
> 1. It **does not depend on any unknown quantities.** If it did, it could not be used to make predictions.
> 2. It **depends on the observed data.** In particular, $\tilde{Y}$ is not independent of $Y_1,\dots,Y_n$, because the observed data provide information about θ, which in turn influences $\tilde{Y}$. If $\tilde{Y}$ were independent of the observed data, learning from data would be impossible.

The uniform prior distribution on [0,1], also known as the Beta$(1,1)$ prior, can be interpreted as containing the same information as a hypothetical prior dataset consisting of one success ("1") and one failure ("0").

Under this prior, the posterior predictive probability of a future success is

$$\Pr(\tilde{Y}=1 \mid Y=y) = \mathbb{E}[\theta \mid Y=y] = \frac{2}{2+n}\cdot\frac{1}{2} + \frac{n}{2+n}\cdot\frac{y}{n}.$$

This expression highlights that the predictive probability is a weighted average of:

- the prior mean 1/2, and
- the sample proportion y/n,

with weights proportional to the prior sample size 2 and the observed sample size n, respectively.

The posterior mode under this prior is

$$\mathrm{mode}(\theta \mid Y=y) = \frac{y}{n},$$

where

$$Y = \sum_{i=1}^{n} Y_i.$$

At first glance, the discrepancy between these two posterior summaries may seem surprising. However, it reflects the fact that different summaries capture different features of the posterior distribution.

To see this clearly, consider the case $Y=0$. In this case,

$$\mathrm{mode}(\theta \mid Y=0) = 0,$$

but the predictive probability remains

$$\Pr(\tilde{Y} = 1 \mid Y = 0) = \frac{1}{2+n}.$$

Thus, even when no successes have been observed, the Bayesian predictive distribution assigns a positive probability to a future success due to the prior information. This illustrates how Bayesian prediction naturally balances prior beliefs with observed data.

3.2.2 Confidence Regions: Bayesian v.s. Frequentist

If it often desirable to identify the regions of the parameter space that are likely to contain the true value of the parameter. To do this, after observing the data $Y = y$, we can construct an interval $[\ell(y), u(y)]$ that is likely to contain the true value of θ, i.e., the probability that $\ell(y) < \theta < u(y)$ is large. There are two different ways to interpret this probability, leading to the concepts of **Bayesian coverage** and **frequentist coverage**.

An interval $[\ell(y), u(y)]$, based on the observed data $Y = y$, has 100(1-α)% Bayesian coverage for θ if

$$\Pr(\ell(y) < \theta < u(y) \mid Y = y) = 1 - \alpha.$$

A random interval $[\ell(Y), u(Y)]$ has 100(1-α)% frequentist coverage for θ if, before the data are gathered,

$$\Pr(\ell(Y) < \theta < u(Y) \mid \theta) = 1 - \alpha.$$

> **Note**
>
> In a sense, the frequentist and Bayesian notions of coverage describe **pre** experimental and **post** experimental perspectives, respectively.

3.3 Frequentist vs Bayesian Coverage

You may recall an important point often emphasized in introductory statistics courses. Suppose we observe data $Y = y$ and compute a frequentist confidence interval

$$[l(y),\ u(y)].$$

Once the data are observed, the parameter θ is **treated as fixed, not random**. Therefore,

$$\Pr(l(y) < \theta < u(y) \mid \theta) = \begin{cases} 1, & \text{if } \theta \in [l(y), u(y)], \\ 0, & \text{if } \theta \notin [l(y), u(y)]. \end{cases}$$

This highlights a key limitation of frequentist confidence intervals:

They do not admit a post-experimental probability interpretation.

After observing the data, it is *not meaningful*, from a frequentist perspective, to say that there is a 95% probability that θ lies in the computed interval.

What Frequentist Coverage Means

Although this interpretation may feel unsatisfying, frequentist coverage is still useful in many situations. Imagine repeatedly running many independent experiments and constructing a confidence interval for each one.
If each interval procedure has 95% frequentist coverage, then:

About 95% of the intervals will contain the true parameter value.

This is a **long-run, repeated-sampling interpretation**, not a statement about any single observed interval.

Can Bayesian and Frequentist Coverage Agree?

A natural question is whether a confidence interval can simultaneously have:

- a Bayesian interpretation, i.e., a 100(1-α)% posterior probability that θ lies in the interval, and
- approximately 100(1-α)% frequentist coverage.

Hartigan (1966) showed that, for the types of intervals considered in Hopf (2009), an interval that has 95% Bayesian coverage additionally has the property that

$$\Pr(l(Y) < \theta < u(Y) \mid \theta) = 0.95 + \varepsilon_n,$$

where the error term satisfies $|\varepsilon_n| < a/n$ for some constant a. This result implies that, an interval with 95% Bayesian coverage, will also have approximately 95% frequentist coverage, at least asymptotically, as the sample size n grows.

In other words, under suitable conditions, **Bayesian credible intervals and frequentist confidence intervals can agree in large samples**, even though their interpretations are fundamentally different. Keep in mind that most non-Bayesian methods of constructing 100(1-α)% confidence intervals also only achieve their nominal coverage probability asymptotically.

> **i** Reminder
>
> This reconciliation is important, but it should not obscure the conceptual distinction:
>
> - frequentist coverage is a *pre-experimental* property of a procedure,
> - Bayesian coverage is a *post-experimental* probability statement about θ given the data.

For further discussion of the similarities between Bayesian and frequentist intervals, see Severini (1991) and Sweeting (2001).

3.4 Posterior Quantile Intervals

One of the simplest ways to construct a Bayesian credible interval is to use **posterior quantiles**. To form a $100(1-\alpha)\%$ credible interval for θ, find numbers $\theta_{\alpha/2} < \theta_{1-\alpha/2}$ such that

1. $\Pr(\theta < \theta_{\alpha/2} \mid Y = y) = \alpha/2$,
2. $\Pr(\theta > \theta_{1-\alpha/2} \mid Y = y) = \alpha/2$,

where $\theta_{\alpha/2}$ and $\theta_{1-\alpha/2}$ are the $\alpha/2$ and $1-\alpha/2$ posterior quantiles of θ. By construction,

$$\begin{aligned}
\Pr\left(\theta \in \left[\theta_{\alpha/2}, \theta_{1-\alpha/2}\right] \mid Y = y\right) &= 1 - \Pr\left(\theta \notin \left[\theta_{\alpha/2}, \theta_{1-\alpha/2}\right] \mid Y = y\right) \\
&= 1 - \left[\Pr\left(\theta < \theta_{\alpha/2} \mid Y = y\right) + \Pr\left(\theta > \theta_{1-\alpha/2} \mid Y = y\right)\right] \\
&= 1 - \alpha.
\end{aligned}$$

Suppose we observe $n = 10$ conditionally independent Bernoulli trials and obtain $Y = 2$ successes. Using a uniform prior for θ, $\theta \sim \text{Beta}(1, 1)$, the posterior distribution is

$$\theta \mid \{Y = 2\} \sim \text{Beta}(1 + 2,\ 1 + 8) = \text{Beta}(3, 9).$$

A 95% posterior confidence interval can be obtained from by 2.5% and 97.5% quantiles of this Beta distribution $[\theta_{0.025}, \theta_{0.975}]$. In this case,

$$\theta_{0.025} \approx 0.06, \qquad \theta_{0.975} \approx 0.52,$$

so

$$\Pr(0.06 \leq \theta \leq 0.52 \mid Y = 2) = 0.95.$$

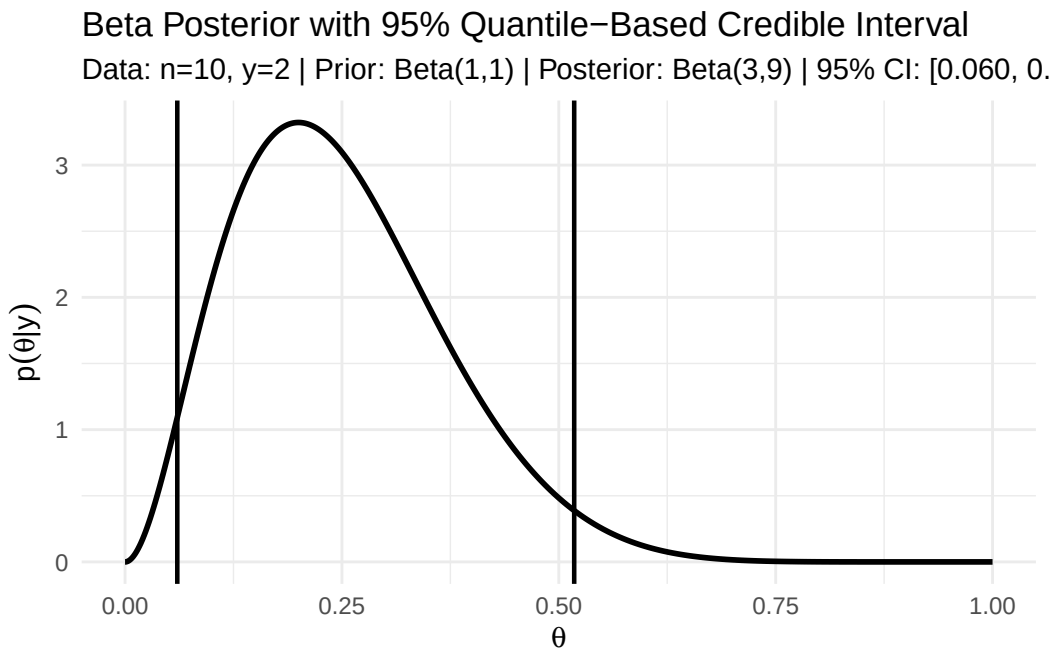
This interval has a direct probabilistic interpretation: **given the observed data**, there is a 95% posterior probability that θ lies in this range.

```
b <- a <- 1 # prior parameter
n <- 10 ; y <- 2 # data
qbeta(c(0.025, 0.975), a + y, b + n - y)
```

```
[1] 0.06021773 0.51775585
```

```
a_post <- a + y
b_post <- b + (n - y)

# 95% quantile-based credible interval
ci <- qbeta(c(0.025, 0.975), a_post, b_post)
```

```r
ci_low <- ci[1]
ci_high <- ci[2]

# Grid for plotting posterior density
theta <- seq(0, 1, length.out = 2000)
df <- data.frame(theta = theta, density = dbeta(theta, a_post, b_post))

# Plot: posterior density curve + two vertical CI bars
ggplot(df, aes(x = theta, y = density)) +
  geom_line(linewidth = 1) +
  geom_vline(xintercept = ci_low, linewidth = 0.8) +
  geom_vline(xintercept = ci_high, linewidth = 0.8) +
  labs(
    title = "Beta Posterior with 95% Quantile-Based Credible Interval",
    subtitle = sprintf(
      "Data: n=%d, y=%d | Prior: Beta(%d,%d) | Posterior: Beta(%d,%d) | 95%% CI: [%.3f, %.3f]
      n, y, a, b, a_post, b_post, ci_low, ci_high
    ),
    x = expression(theta),
    y = expression(p(theta * "|" * y))
  ) +
  theme_minimal()
```



Highest posterior density (HPD) region

The Figure above illustrates the posterior distribution of θ for the binomial example with a uniform prior, together with a 95% quantile-based credible interval. Notice an important feature of the plot:

> There exist values of θ *outside* the quantile-based interval that have *higher posterior density* than some values *inside* the interval.

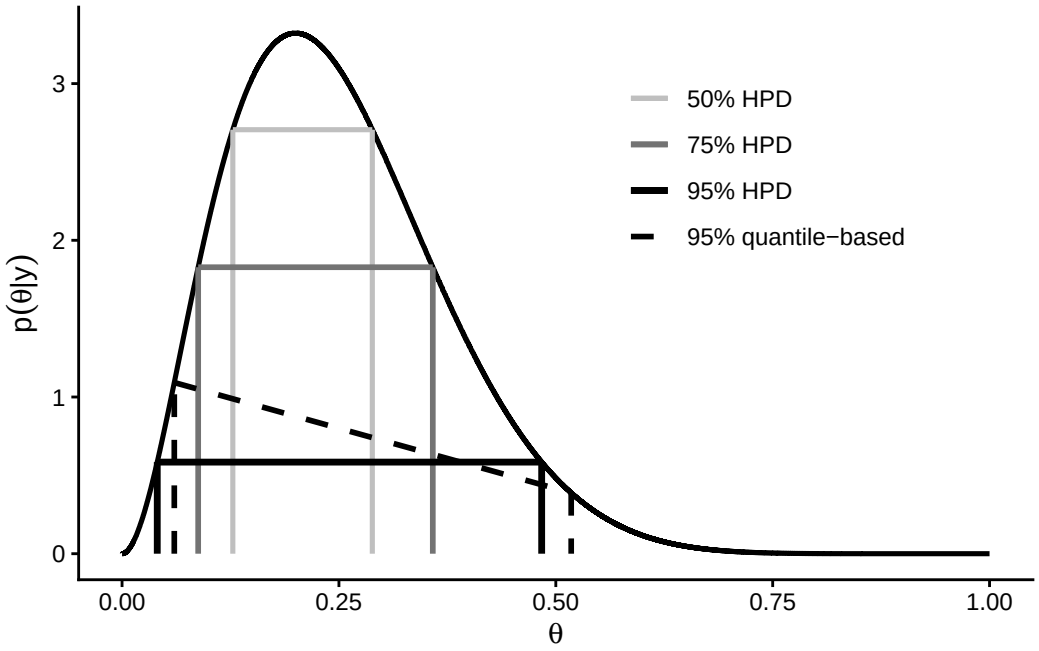
This observation suggests that the quantile-based interval may not be the most efficient way to summarize posterior uncertainty. In particular, it motivates a more restrictive type of credible region that concentrates on the most plausible parameter values.

A $100(1-\alpha)\%$ *HPD region* is a subset of the sample space, $s(y) \subset \Theta$ such that:

1. $\Pr(\theta \in s(y) \mid Y = y) = 1 - \alpha$, and
2. If $\theta_a \in s(y)$ and $\theta_b \notin s(y)$, then $p(\theta_a \mid Y = y) \geq p(\theta_b \mid Y = y)$.

In words, an HPD region contains the parameter values with the *largest posterior density*, subject to containing probability mass $1 - \alpha$.

Observed that, all points inside an HPD region are at least as plausible as any point outside the region, according to the posterior distribution. This property distinguishes HPD regions from quantile-based intervals, which are defined purely by cumulative probability and may include low-density values while excluding higher-density ones.



An HPD region can be constructed conceptually as follows:

> **Algorithm to construct an HPD region**
>
> 1. Begin with a horizontal line above the posterior density curve.
> 2. Gradually lower the line.
> 3. At each height, include all values of θ whose posterior density exceeds the line.
> 4. Stop lowering the line once the total posterior probability of the included region reaches $1 - \alpha$.

This procedure guarantees that the retained region consists of the most probable values of θ.

HPD Regions and Multimodality

If the posterior density is **unimodal**, the HPD region will typically be a single interval. However, if the posterior density is **multimodal** (having multiple peaks), the HPD region need not be an interval; it may consist of several disjoint subsets of the parameter space.

In the binomial example with $n = 10$, $Y = 2$, and a uniform prior, the posterior distribution is $\text{Beta}(3, 9)$.

For this posterior:

- The 95% quantile-based credible interval is approximately [0.06, 0.52].
- The 95% HPD region is approximately [0.04, 0.48].

The HPD region is *narrower*, and therefore more precise, than the quantile-based interval, while still containing 95% of the posterior probability.

Both intervals are valid Bayesian credible intervals, but they summarize posterior uncertainty in different ways.

3.5 The Poisson Model

Another commonly used distribution is the *Poisson*, in this case, the measurement are the integer numbers. Some examples include number of coin tosses, the number of friends they have, or the number of birthday celebrations have a person have. In these situations, the sample space is $\mathcal{Y} = \{0, 1, 2, ...\}$. There are other possible models for those situation, but perhaps the simplest probability model on $\mathcal{Y}$ is the Poisson model.

Poisson distribution

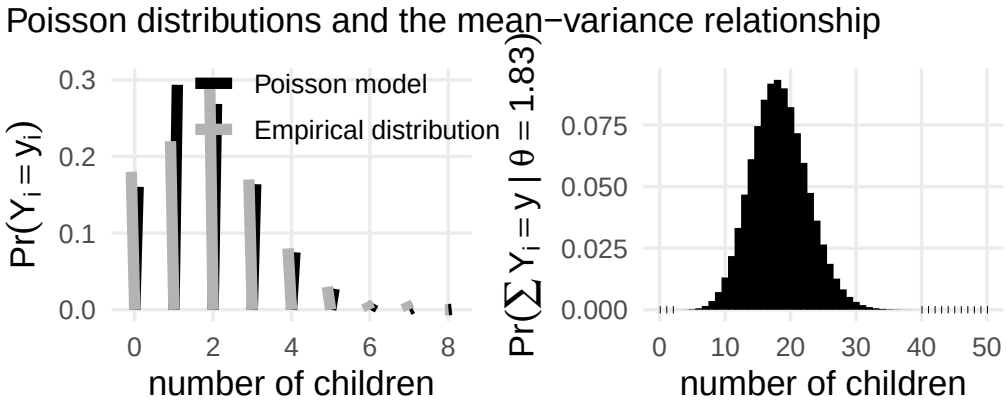
Recall that a random variable Y has a Poisson distribution with mean θ if

$$\Pr(Y = y \mid \theta) = \text{dpois}(y, \theta) = \frac{\theta^y e^{-\theta}}{y!}, \qquad y \in \{0, 1, 2, ...\}.$$

For such a random variable, we have,

- $\mathbb{E}(Y) = \theta$
- $\text{Var}(Y) = \theta$.

People sometimes use the Poisson distribution to model count data because of its simplicity and its ability to model events that occur independently over a fixed interval of time or space. The Poisson distribution is particularly useful when the events being counted are rare or infrequent, and when the average rate of occurrence is known. Note that, in this model, the mean and the variance are the same, which is a property that can be useful in certain applications; One may call this property as "mean-variance relationship".



Left: Poisson pmf with mean . = 1.83 (black) overlaid with an empirical distribution (grey

Right: Distribution of the sum of 10 i.i.d. Poisson(1.83) variables; by additivity this is Pois

The increased spread illustrates the Poisson mean−variance relationship: larger means

3.5.1 Inference on the Posterior for Poisson Model

Suppose we observe data $Y_1, \ldots, Y_n$ and model them as conditionally independent Poisson random variables with common mean θ, i.e.,

$$Y_i \mid \theta \sim \text{Poisson}(\theta), \qquad i = 1, \ldots, n.$$

The joint probability mass function of the data, given θ, is

$$\Pr(Y_1 = y_1, \ldots, Y_n = y_n \mid \theta) = \prod_{i=1}^{n} p(y_i \mid \theta).$$

Using the Poisson pmf,

$$p(y_i \mid \theta) = \frac{\theta^{y_i} e^{-\theta}}{y_i!},$$

we obtain

$$\begin{aligned}\Pr(Y_1 = y_1, \dots, Y_n = y_n \mid \theta) &= \prod_{i=1}^{n} \frac{\theta^{y_i} e^{-\theta}}{y_i!} \\ &= c(y_1, \dots, y_n)\, \theta^{\sum_{i=1}^n y_i} e^{-n\theta},\end{aligned}$$

where

$$c(y_1, \dots, y_n) = \prod_{i=1}^{n} \frac{1}{y_i!},$$

which does not depend on θ. This expression shows that the likelihood depends on the data only through the statistic $S = \sum_{i=1}^n Y_i$.

As in the binomial model, the statistic $S = \sum_{i=1}^n Y_i$ contains all information in the data about θ.
Indeed,

$$\sum_{i=1}^{n} Y_i \mid \theta \sim \text{Poisson}(n\theta),$$

and we therefore say that S is a sufficient statistic for θ.

3.5.2 Comparing posterior beliefs

To compare two values θ_a and θ_b *a posteriori*, consider the posterior odds:

$$\frac{p(\theta_a \mid y_1, \dots, y_n)}{p(\theta_b \mid y_1, \dots, y_n)}.$$

By Bayes' rule,

$$p(\theta \mid y_1, \dots, y_n) \propto \pi(\theta)\, p(y_1, \dots, y_n \mid \theta) \propto \pi(\theta)\, \theta^{\sum_{i=1}^n y_i} e^{-n\theta}.$$

Therefore,

$$\frac{p(\theta_a \mid y)}{p(\theta_b \mid y)} = \frac{\theta_a^{\sum y_i} e^{-n\theta_a} p(\theta_a)}{\theta_b^{\sum y_i} e^{-n\theta_b} p(\theta_b)}.$$

This expression highlights how posterior beliefs balance prior information with evidence from the data.

Conjugate prior for the Poisson model

We now seek a prior distribution for θ that leads to a posterior distribution of the same functional form. From the likelihood,

$$p(\theta \mid y) \propto p(\theta)\, \theta^{\sum y_i} e^{-n\theta},$$

we see that a conjugate prior must involve terms of the form

$$\theta^{c_1} e^{-c_2 \theta}$$

for some constants c_1 and c_2. The simplest family of distributions with this structure is the family of Gamma distributions, also called **Gamma family**.

3.5.3 Gamma distribution

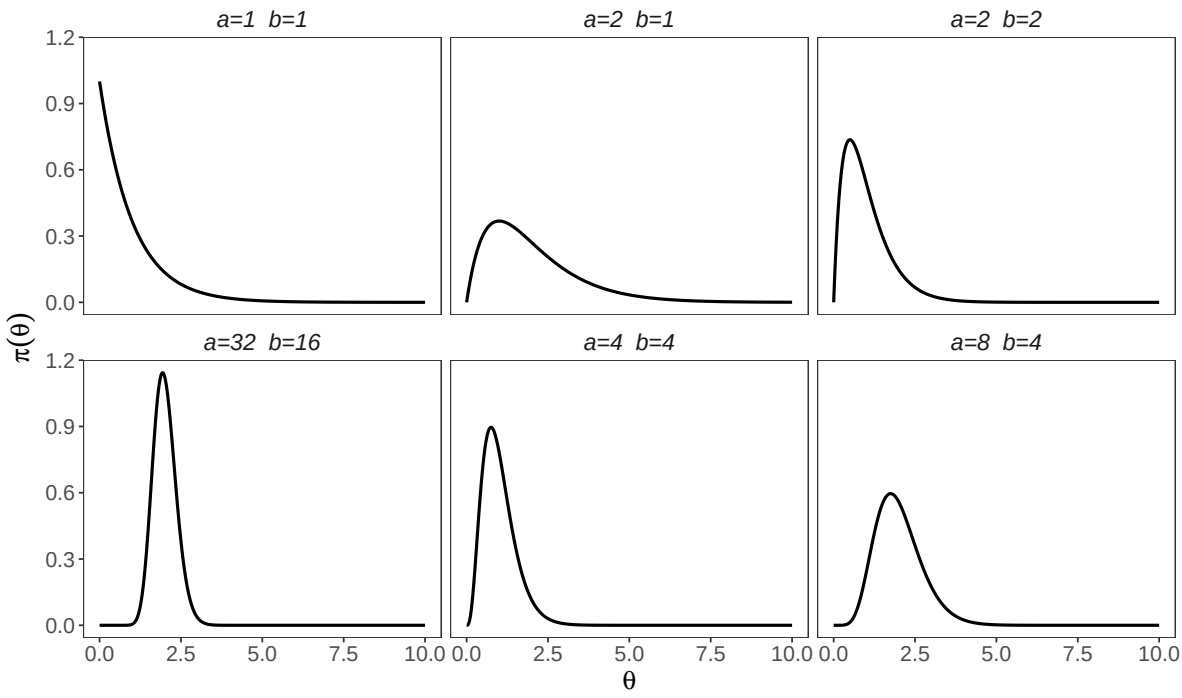
A positive random variable θ has a Gamma(a, b) distribution if

$$\pi(\theta) = \frac{b^a}{\Gamma(a)} \theta^{a-1} e^{-b\theta}, \qquad \theta > 0,$$

where $a > 0$ is the shape parameter and $b > 0$ is the rate parameter.

For a Gamma(a, b) random variable,

- $\mathbb{E}(\theta) = a/b$,
- $\mathrm{Var}(\theta) = a/b^2$.
- $\mathrm{mode}[\theta] = \begin{cases} (a-1)/b & \text{if } a > 1 \\ 0 & \text{if } a \leq 1 \end{cases}$.



3.5.4 Posterior distribution of θ

If the prior is $\theta \sim \text{Gamma}(a, b)$, then combining the prior with the Poisson likelihood yields

$$\begin{aligned} p\left(\theta \mid y_1, \ldots, y_n\right) &= \pi(\theta) \times p\left(y_1, \ldots, y_n \mid \theta\right) / p\left(y_1, \ldots, y_n\right) \\ &= \left\{\theta^{a-1} e^{-b\theta}\right\} \times \left\{\theta^{\sum y_i} e^{-n\theta}\right\} \times c\left(y_1, \ldots, y_n, a, b\right) \\ &= \left\{\theta^{a+\sum y_i - 1} e^{-(b+n)\theta}\right\} \times c\left(y_1, \ldots, y_n, a, b\right) \\ &\propto \theta^{a+\sum y_i - 1} e^{-(b+n)\theta}. \end{aligned}$$

Thus, by the uniqueness theorem (of the density), the posterior distribution is

$$\theta \mid y_1, \ldots, y_n \sim \text{Gamma}(a + \sum_{i=1}^{n} y_i,\ b + n).$$

This shows that the Gamma distribution is **conjugate** to the Poisson likelihood.

Interpretation

Posterior inference for the Poisson model is therefore straightforward:

- The data enter only through the sufficient statistic $\sum Y_i$;

- The posterior mean is

$$\begin{aligned} \mathrm{E}\left[\theta \mid y_1, \dots, y_n\right] &= \frac{a + \sum y_i}{b + n} \\ &= \frac{b}{b+n}\frac{a}{b} + \frac{n}{b+n}\frac{\sum y_i}{n} \end{aligned}$$

- This decomposition shows that it is a **convex combination** of the prior mean a/b and the sample mean $\bar{y}$, and gives a useful information
- b acts like the **number of prior observations**;
- a acts like the **total count** from those b observations;
- a/b is the prior mean.
- Increasing the sample size n reduces posterior uncertainty, because the ifnromation from the data *dominates* the prior belief. To see this, we have, for $n \gg b$, we have
- $\mathrm{E}[\theta \mid y] \approx \bar{y}$,
- $\mathrm{Var}(\theta \mid y) \approx \bar{y}/n$.

This conjugate structure makes the Poisson–Gamma model a convenient and interpretable starting point for Bayesian analysis of *count data.*

3.5.5 Posterior predictive distribution for Poisson Model

We have seen that, the Bayesian prediction for a future observation $\tilde{y}$ is based on the **posterior predictive distribution**. In Gamma-Poisson model, we have

$$p(\tilde{y} \mid y_1, \dots, y_n) = \int_0^\infty p(\tilde{y} \mid \theta, y)\, p(\theta \mid y_1, \dots, y_n)\, d\theta.$$

For the Poisson model,

$$p(\tilde{y} \mid \theta) = \mathrm{Poisson}(\theta), \qquad p(\theta \mid y) = \mathrm{Gamma}\Big(a + \sum y_i,\ b + n\Big)\,.$$

Substituting,

$$p(\tilde{y} \mid y) = \int_0^\infty \mathrm{dpois}(\tilde{y}, \theta)\, \mathrm{dgamma}\Big(\theta,\ a + \sum y_i,\ b + n\Big)\ d\theta.$$

Writing this integral explicitly,

$$p(\tilde{y} \mid y) = \int_0^\infty \frac{\theta^{\tilde{y}} e^{-\theta}}{\tilde{y}!} \cdot \frac{(b+n)^{a+\sum y_i}}{\Gamma(a + \sum y_i)} \theta^{a+\sum y_i - 1} e^{-(b+n)\theta}\, d\theta.$$

Combining terms, we have

$$p(\tilde{y} \mid y) = \frac{(b+n)^{a+\sum y_i}}{\tilde{y}!\,\Gamma(a+\sum y_i)} \int_0^\infty \theta^{a+\sum y_i+\tilde{y}-1} e^{-(b+n+1)\theta}\, d\theta.$$

The integral term seems difficult to evaluate in a glance, but there is actually a clear "trick" to help use. Recall the density of a Gamma distribution:

$$1 = \int_0^\infty \frac{\beta^\alpha}{\Gamma(\alpha)} \theta^{\alpha-1} e^{-\beta\theta}\, d\theta, \quad ,\alpha,\beta > 0,$$

which implies

$$\int_0^\infty \theta^{\alpha-1} e^{-\beta\theta}\, d\theta = \frac{\Gamma(\alpha)}{\beta^\alpha}, \qquad \alpha,\beta > 0.$$

Applying this with

$$\alpha = a + \sum y_i + \tilde{y}, \qquad \beta = b + n + 1,$$

we obtain

$$\int_0^\infty \theta^{a+\sum y_i+\tilde{y}-1} e^{-(b+n+1)\theta}\, d\theta = \frac{\Gamma(a+\sum y_i+\tilde{y})}{(b+n+1)^{a+\sum y_i+\tilde{y}}}.$$

Substituting back and simplifying,

$$p(\tilde{y} \mid y_1, \dots, y_n) = \frac{\Gamma(a+\sum y_i+\tilde{y})}{\Gamma(a+\sum y_i)\,\tilde{y}!} \left(\frac{b+n}{b+n+1}\right)^{a+\sum y_i} \left(\frac{1}{b+n+1}\right)^{\tilde{y}},$$

for $\tilde{y} \in \{0, 1, 2, ...\}$. We relieazed that it is a **negative binomial distribution** with parameters $(a + \sum y_i, b + n)$. That is, $\tilde{Y} \mid y_1, \dots, y_n \sim \text{NB}(a + \sum y_i, b + n)$. As a result, we have the mean and variance of the posterior predictive distribution

$$\mathbb{E}(\tilde{Y} \mid y) = \frac{a+\sum y_i}{b+n} = \mathbb{E}[\theta \mid y],$$

and

$$\begin{aligned}
\text{Var}\left[\tilde{Y} \mid y_1, \dots, y_n\right] &= \frac{a+\sum y_i}{b+n} \frac{b+n+1}{b+n} \\
&= \text{Var}\left[\theta \mid y_1, \dots y_n\right] \times (b+n+1) \\
&= \mathbb{E}\left[\theta \mid y_1, \dots, y_n\right] \times \frac{b+n+1}{b+n} \quad , \\
&= \frac{a+\sum y_i}{b+n} \times \frac{b+n+1}{b+n}
\end{aligned}$$

respectively

Interpretation and Take away

Recall that the predictive variance is to some extent a measure of our posterior uncertainty about a new observation $\tilde{Y}$. It reflects **two sources of uncertainty**:

1. **Sampling variability** (Sampling) For a Poisson model, the variance of Y given θ is equal to θ.
2. **Parameter uncertainty** (Population) When θ is unknown, uncertainty about θ inflates the variance of future observations.

For large n, the data dominate the prior:

$$\frac{b+n+1}{b+n} \approx 1,$$

so predictive uncertainty is driven primarily by sampling variability. In this case, the uncertainy abbout θ is small which for the Possion model is equal to *theta*

For small n, posterior uncertainty about θ is substantial, and

$$\frac{b+n+1}{b+n} > 1,$$

leading to larger predictive variance than under a fixed-θ Poisson model (i.e., larger than just the sampling variability).

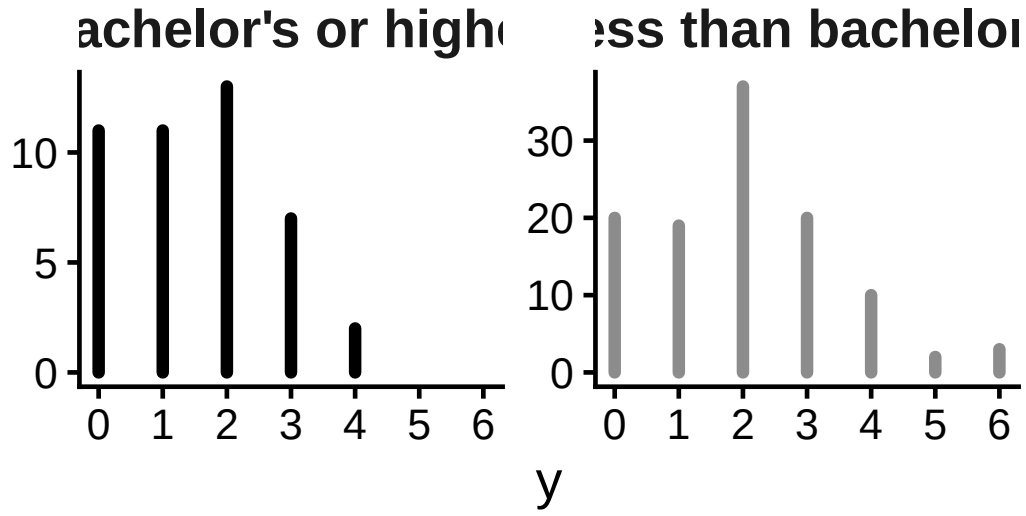
- Bayesian prediction naturally incorporates both **sampling variability** and **parameter uncertainty**.

This completes posterior inference and prediction for the Gamma-Poisson model.

3.6 Example: Birth rates

Over the course of the 1990s, the General Social Survey collected data on the educational attainment and number of children of women who were 40 years old at the time of participation in the survey. These women were in their 20s during the 1970s, a period of historically low fertility rates in the United States.

In this example, we compare women **with** a bachelor's degree to those **without** a bachelor's degree in terms of their numbers of children.



Numbers of children for the two groups.

Sampling models

Let

- $Y_{i,1}$ denote the number of children for woman i **without** a bachelor's degree, $i = 1, \dots, n_1$,
- $Y_{i,2}$ denote the number of children for woman i **with** a bachelor's degree, $i = 1, \dots, n_2$.

Then, since it is a coint data, we assume the following Poisson sampling models:

$$Y_{1,1}, \dots, Y_{n_1,1} \mid \theta_1 \sim \text{i.i.d. Poisson}(\theta_1),$$

$$Y_{1,2}, \dots, Y_{n_2,2} \mid \theta_2 \sim \text{i.i.d. Poisson}(\theta_2).$$

Here, θ_1 and θ_2 represent the population mean birth rates for the two groups.

Data summaries

The empirical summaries of the data are:

- No college degree:

$$n_1 = 111, \qquad \sum_{i=1}^{n_1} Y_{i,1} = 217, \qquad \bar{Y}_1 = 1.95$$

- Bachelor's degree or higher:

$$n_2 = 44, \qquad \sum_{i=1}^{n_2} Y_{i,2} = 66, \qquad \bar{Y}_2 = 1.50$$

Prior distributions

We place independent Gamma priors on the two population means:

$$\theta_1, \theta_2 \sim \text{i.i.d. Gamma}(a = 2, b = 1),$$

where the Gamma distribution is parameterized by **shape** a and **rate** b.

The prior mean is $a/b = 2$, representing weak prior information corresponding to roughly one prior observation with an average count of two. (Why?)

Posterior distributions

Under the Poisson–Gamma conjugate model, the posterior distributions are

$$\theta_1 \mid y \sim \text{Gamma}(a + \sum Y_{i,1},\, b + n_1) = \text{Gamma}(219, 112),$$

$$\theta_2 \mid y \sim \text{Gamma}(a + \sum Y_{i,2},\, b + n_2) = \text{Gamma}(68, 45).$$

These posteriors summarize our updated beliefs about the two population mean birth rates after observing the data. From the Gamma posterior distributions, we can compute posterior means, modes, and 95% quantile-based credible intervals.

- Posterior means:

$$\mathbb{E}(\theta_1 \mid y) = \frac{219}{112} \approx 1.96, \qquad \mathbb{E}(\theta_2 \mid y) = \frac{68}{45} \approx 1.51$$

- Posterior modes:

$$\text{mode}(\theta_1 \mid y) = \frac{218}{112} \approx 1.95, \qquad \text{mode}(\theta_2 \mid y) = \frac{67}{45} \approx 1.49$$

```r
# ============================================================
# Poisson-Gamma model: two-group posterior summaries
# Prior:  theta ~ Gamma(a, b)  with shape = a, rate = b
# Data:   Yi | theta ~ i.i.d. Poisson(theta)
# Posterior: theta | y ~ Gamma(a + sum(y), b + n)
# ============================================================

# ----------------------------
# 1) Inputs: prior + data
# ----------------------------
a <- 2
b <- 1

group <- data.frame(
  group = c("Less than bachelor's", "Bachelor's or higher"),
  n     = c(111, 44),
  sum_y = c(217, 66)
)

# ----------------------------
# 2) Posterior functions
# ----------------------------
post_shape <- function(a, sum_y) a + sum_y
post_rate  <- function(b, n)     b + n

post_mean <- function(shape, rate) shape / rate
post_mode <- function(shape, rate) ifelse(shape > 1, (shape - 1) / rate, 0)

post_ci <- function(shape, rate, level = 0.95) {
  alpha <- (1 - level) / 2
  qgamma(c(alpha, 1 - alpha), shape = shape, rate = rate)
}

# ----------------------------
# 3) Compute summaries
# ----------------------------
out <- within(group, {
  shape_post <- post_shape(a, sum_y)
  rate_post  <- post_rate(b, n)

  mean_post  <- post_mean(shape_post, rate_post)
  mode_post  <- post_mode(shape_post, rate_post)
```

```r
  ci_post    <- t(mapply(post_ci, shape_post, rate_post))
  ci_lower   <- ci_post[, 1]
  ci_upper   <- ci_post[, 2]
})

# ----------------------------
# 4) Print results clearly
# ----------------------------
cat("Prior: theta ~ Gamma(shape = ", a, ", rate = ", b, ")\n\n", sep = "")
```

```
Prior: theta ~ Gamma(shape = 2, rate = 1)
```

```r
for (i in seq_len(nrow(out))) {
  cat("------------------------------------------------------------\n")
  cat("Group: ", out$group[i], "\n", sep = "")
  cat("n = ", out$n[i], ",  sum(y) = ", out$sum_y[i], "\n", sep = "")
  cat("Posterior: theta | y ~ Gamma(shape = ", out$shape_post[i],
      ", rate = ", out$rate_post[i], ")\n", sep = "")
  cat(sprintf("Posterior mean = %.6f\n", out$mean_post[i]))
  cat(sprintf("Posterior mode = %.6f\n", out$mode_post[i]))
  cat(sprintf("Posterior 95%% CI = [%.6f, %.6f]\n", out$ci_lower[i], out$ci_upper[i]))
}
```

```
------------------------------------------------------------
Group: Less than bachelor's
n = 111,  sum(y) = 217
Posterior: theta | y ~ Gamma(shape = 219, rate = 112)
Posterior mean = 1.955357
Posterior mode = 1.946429
Posterior 95% CI = [1.704943, 2.222679]
------------------------------------------------------------
Group: Bachelor's or higher
n = 44,  sum(y) = 66
Posterior: theta | y ~ Gamma(shape = 68, rate = 45)
Posterior mean = 1.511111
Posterior mode = 1.488889
Posterior 95% CI = [1.173437, 1.890836]
```

```r
summary_tbl <- out[, c("group", "n", "sum_y", "mean_post", "mode_post", "ci_lower", "ci_upper
print(summary_tbl, row.names = FALSE)
```

```
                group   n sum_y mean_post mode_post ci_lower ci_upper
 Less than bachelor's 111   217  1.955357  1.946429 1.704943 2.222679
 Bachelor's or higher  44    66  1.511111  1.488889 1.173437 1.890836
```

The posterior distributions indicate **strong evidence** that the mean birth rate is higher for women without a bachelor's degree (i.e., $\theta_1 > \theta_2$). For example,

$$\Pr(\theta_1 > \theta_2 \mid y) \approx 0.97.$$

Posterior predictive distributions

Now consider two randomly sampled future individuals:

- $\tilde{Y}_1$: a woman without a bachelor's degree,
- $\tilde{Y}_2$: a woman with a bachelor's degree.

  Question: To what extent do we expect the one without the college degree to have more children than the other?

The posterior predictive distributions integrate over uncertainty in θ:

$$p(\tilde{y} \mid y) = \int p(\tilde{y} \mid \theta)\, p(\theta \mid y)\, d\theta.$$

Under the Poisson–Gamma model, the posterior predictive distributions are **Negative Binomial**:

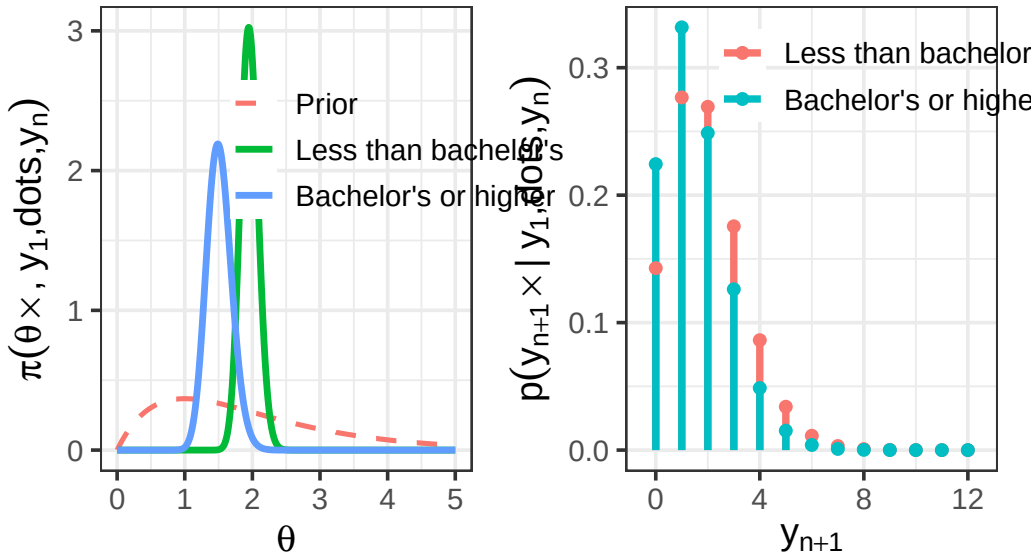
$$\tilde{Y}_1 \mid y \sim \text{NegBin}(a + \sum Y_{i,1},\, b + n_1),$$

$$\tilde{Y}_2 \mid y \sim \text{NegBin}(a + \sum Y_{i,2},\, b + n_2).$$

```
# A tibble: 2 x 9
  group                   n sum_y shape  rate post_mean post_mode  q025  q975
  <chr>               <dbl> <dbl> <dbl> <dbl>     <dbl>     <dbl> <dbl> <dbl>
1 Less than bachelor's   111   217   219   112      1.96      1.95  1.70  2.22
2 Bachelor's or higher    44    66    68    45      1.51      1.49  1.17  1.89
```

```
y <- 0:10
```

```
Group 1: Less than bachelor's
```



posterior predictive distributions for the number of children in each group.

Figure 3.1

```
size = 219   mu = 1.955357

 [1] 1.427473e-01 2.766518e-01 2.693071e-01 1.755660e-01 8.622930e-02
 [6] 3.403387e-02 1.124423e-02 3.198421e-03 7.996053e-04 1.784763e-04
[11] 3.601115e-05


Group 2: Bachelor's or higher

size = 68   mu = 1.511111

 [1] 2.243460e-01 3.316420e-01 2.487315e-01 1.261681e-01 4.868444e-02
 [6] 1.524035e-02 4.030961e-03 9.263700e-04 1.887982e-04 3.465861e-05
[11] 5.801551e-06

    y Less.than.bachelor.s Bachelor.s.or.higher
1   0         1.427473e-01         2.243460e-01
2   1         2.766518e-01         3.316420e-01
3   2         2.693071e-01         2.487315e-01
```

```
4   3          1.755660e-01          1.261681e-01
5   4          8.622930e-02          4.868444e-02
6   5          3.403387e-02          1.524035e-02
7   6          1.124423e-02          4.030961e-03
8   7          3.198421e-03          9.263700e-04
9   8          7.996053e-04          1.887982e-04
10  9          1.784763e-04          3.465861e-05
11 10          3.601115e-05          5.801551e-06
```

Interpretation and Conclusion

Although the posterior distributions of θ_1 and θ_2 are clearly separated, the posterior predictive distributions for $\tilde{Y}_1$ and $\tilde{Y}_2$ exhibit substantial overlap.

For example,

$$\Pr(\tilde{Y}_1 > \tilde{Y}_2 \mid y) \approx 0.48, \qquad \Pr(\tilde{Y}_1 = \tilde{Y}_2 \mid y) \approx 0.22.$$

This illustrates an important distinction:

> Strong evidence that population means differ does **not** imply that the difference is large or easily detectable at the individual level.

> **Note**
>
> Population-level effects and individual-level variability are fundamentally different sources of uncertainty.

3.7 Exponential Family

Many common sampling models belong to the **exponential family** of distributions, including the binomial and Poisson distribution we saw in this chapter. A one-parameter exponential family has probability density (or mass) function of the form

$$\begin{aligned} p(y \mid \phi) &= h(y) \exp\{\phi t(y) - A(\phi)\} \\ &= h(y) c(\phi) \exp\{\phi t(y)\}, \end{aligned}$$

where ϕ is the unknown parameter, $t(y)$ is a sufficient statistic, and $h(y)$, $A(\phi)$, and $c(\phi)$ are known functions. Diaconis and Ylvisaker (1979) studied conjugate priors for exponential family models and showed that they have the general form

$$p(\phi \mid n_0, t_0) = \kappa(n_0, t_0) c(\phi)^{n_0} \exp\{n_0 t_0 \phi\},$$

where $n_0 > 0$ and t_0 are hyperparameters.

With this result, and suppose the data consist of n independent observations $y_1, \dots, y_n$ are sampled from $Y_1, \dots, Y_n \overset{iid}{\sim} p(y \mid \theta)$. Combining the prior with the likelihood gives the posterior distribution

$$p(\phi \mid y_1, \dots, y_n) \;\propto\; p(\phi)\, p(y_1, \dots, y_n \mid \phi).$$

Substituting the exponential-family form,

$$\begin{aligned} p(\phi \mid y_1, \dots, y_n) &\propto c(\phi)^{\,n_0+n} \exp\left\{ \phi \left[n_0 t_0 + \sum_{i=1}^{n} t(y_i) \right] \right\} \\ &\propto p(\phi \mid n_0 + n,\; n_0 t_0 + n\,\bar{t}(y))\,, \end{aligned}$$

where

$$\bar{t}(y) = \frac{1}{n} \sum_{i=1}^{n} t(y_i).$$

Thus, the posterior distribution has the **same functional form** as the prior. This is why such priors are called *conjugate*.

3.7.1 Interpretation of n_0 and t_0

The similarity between the prior and posterior distributions suggests an interpretation of the hyperparameters:

- n_0 can be interpreted as a **prior sample size**,
- t_0 can be interpreted as a **prior guess** of $t(Y)$.

This interpretation can be made more precise. Diaconis and Ylvisaker (1979) show that

$$\mathbb{E}[t(Y)] = \mathbb{E}[\,\mathbb{E}[t(Y) \mid \phi]\,] = \mathbb{E}\left[-\frac{c'(\phi)}{c(\phi)} \right] = t_0.$$

(See also Exercise 3.6 in Hopf, 2009)

Thus, t_0 represents the **prior expected value** of the sufficient statistic $t(Y)$.

The parameter n_0 measures how informative the prior is. One way to see this is to note that, as a function of ϕ, $p(\phi \mid n_0, t_0)$ has the same shape as a likelihood $p(\tilde{y}_1, \dots, \tilde{y}_{n_0} \mid \phi)$ based on n_0 hypothetical "prior observations" $\tilde{y}_1, \dots, \tilde{y}_{n_0}$ satisfying

$$\frac{1}{n_0} \sum_{i=1}^{n_0} t(\tilde{y}_i) = t_0.$$

In this sense, the prior distribution $p(\phi \mid n_0, t_0)$ contains the same amount of information as would be obtained from n_0 independent observations from the population.

The exponential family representation of the binomial(θ) model can be obtained from the density of a single Bernoulli random variable:

$$p(y \mid \theta) = \theta^y (1-\theta)^{1-y}.$$

Rewrite this as

$$\begin{aligned} p(y \mid \theta) &= \left(\frac{\theta}{1-\theta}\right)^y (1-\theta) \\ &= \exp\left\{y \log\left(\frac{\theta}{1-\theta}\right)\right\}(1-\theta). \end{aligned}$$

Let

$$\phi = \log\left(\frac{\theta}{1-\theta}\right),$$

the **log-odds**. Then

$$\theta = \frac{e^\phi}{1+e^\phi}, \qquad 1-\theta = \frac{1}{1+e^\phi}.$$

Substituting gives

$$p(y \mid \phi) = e^{\phi y}(1+e^\phi)^{-1}.$$

Thus the Bernoulli model is an exponential family model with

- $t(y) = y$,
- $c(\phi) = (1+e^\phi)^{-1}$,
- $h(y) = 1$.

Conjugate prior

The conjugate prior for ϕ has the form

$$p(\phi \mid n_0, t_0) \propto (1+e^\phi)^{-n_0} \exp\{n_0 t_0 \phi\}.$$

Since $t(y) = y$, the parameter t_0 represents the prior expectation of Y, or equivalently,

$$t_0 = \mathbb{E}(\theta).$$

Using a change of variables back to θ, the prior becomes

$$p(\theta \mid n_0, t_0) \propto \theta^{n_0 t_0 - 1}(1-\theta)^{n_0(1-t_0)-1},$$

which is a Beta distribution:

$$\theta \sim \text{Beta}(n_0 t_0,\ n_0(1-t_0)).$$

A weakly informative prior can be obtained by setting

- t_0 equal to our prior expectation,
- $n_0 = 1$.

For example, if our prior expectation is $1/2$, this gives

$$\theta \sim \text{Beta}(1/2, 1/2),$$

which is Jeffreys' prior for the binomial model.

Under this prior, the posterior distribution is

$$\theta \mid y_1, \ldots, y_n \sim \text{Beta}\left(t_0 + \sum y_i,\ (1-t_0) + \sum (1-y_i)\right).$$

The Poisson(θ) model can also be written in exponential family form.

The Poisson pmf is

$$p(y \mid \theta) = \frac{\theta^y e^{-\theta}}{y!}.$$

Rewrite as

$$p(y \mid \theta) = \frac{1}{y!}\exp\{y \log\theta - \theta\}.$$

Thus it is an exponential family with

- $t(y) = y$,
- $\phi = \log\theta$,
- $c(\phi) = \exp(-e^{\phi})$,

- $h(y) = 1/y!$.

Conjugate prior

The conjugate prior for ϕ has the form

$$p(\phi \mid n_0, t_0) \propto \exp\{n_0 t_0 \phi - n_0 e^{\phi}\}.$$

Transforming back to $\theta = e^{\phi}$, this becomes

$$p(\theta \mid n_0, t_0) \propto \theta^{n_0 t_0 - 1} e^{-n_0 \theta},$$

which is a Gamma distribution:

$$\theta \sim \text{Gamma}(n_0 t_0,\ n_0).$$

A weakly informative prior can be obtained by setting

- t_0 equal to the prior expectation of θ,
- $n_0 = 1$,

giving

$$\theta \sim \text{Gamma}(t_0, 1).$$

Under this prior, the posterior distribution is

$$\theta \mid y_1, \ldots, y_n \sim \text{Gamma}\left(t_0 + \sum y_i,\ n_0 + n\right).$$

3.8 Discussion

The notion of conjugacy for classes of prior distributions was developed in Raiffa and Schlaifer (1961). Important results on conjugacy for exponential families appear in Diaconis and Ylvisaker (1979) and Diaconis and Ylvisaker (1985). They show that any prior distribution may be approximated by a mixture of conjugate priors.

Most authors refer to intervals of high posterior probability as **credible intervals**, as opposed to confidence intervals. However, credible intervals do not necessarily have frequentist coverage properties. In many common models they are numerically similar to classical intervals, but this is not guaranteed.

Some authors argue that accurate frequentist coverage can guide the construction of prior distributions. See Kass and Wasserman (1996) for a review of formal methods for selecting prior distributions.

This Chapter follows closely with Chapter 3 in Hoff (2009).

4 Monte Carlo Methods

Leading objectives:

- understand how Monte Carlo (MC) and Markov chain Monte Carlo (MCMC) methods differ
- learn how to use MC methods to approximate posterior expectations and probabilities
- understand how to sample from the posterior predictive distribution using MC methods

What we have seen in the last chapter up to now, is to use the `conjugate prior` to obtain closed form expressions for the posterior distribution. However, in many cases, conjugate priors are not available or not desirable. In such cases, we need to resort to numerical methods to approximate the posterior distribution.

Question.

How can we perform Bayesian inference when conjugate priors are not available and the posterior has no closed-form expression?

There are two broad classes of approaches:

- **Simulation-based methods**:
  accept–reject sampling, Markov chain Monte Carlo (MCMC), particle filters, and related algorithms.
- **Deterministic approximation methods**:
  Laplace approximations (including INLA), variational Bayes, expectation propagation, and related techniques.

We will be focusing on the Monte Carlo (MC) methods and its variation.

4.1 Overview

Monte Carlo

MC methods approximate expectations or probabilities using **random sampling**.

If samples can be drawn **directly** from the target distribution, Monte Carlo methods provide simple and effective estimators.

Typical use:

- Numerical integration
- Bootstrap methods
- Simulation-based probability estimation

Markov Chain Monte Carlo

MCMC methods are used when **direct sampling is infeasible**. They construct a **Markov chain** whose stationary distribution is the target distribution, and use dependent samples from the chain after burn-in.

Typical use:

- Bayesian posterior sampling
- High-dimensional or unnormalized distributions

Gibbs Sampler

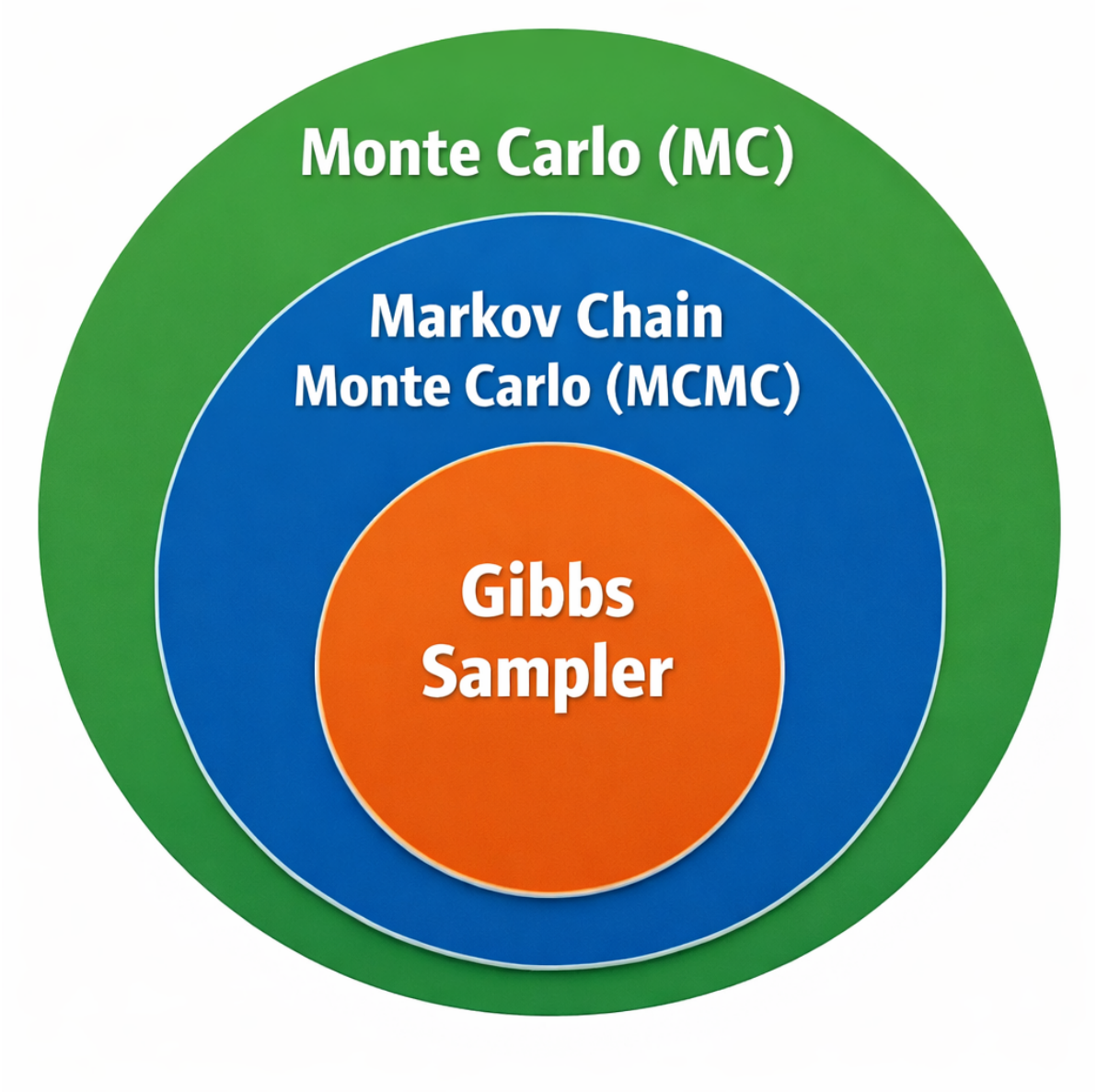
The Gibbs sampler is a **special case of MCMC** that samples sequentially from **full conditional distributions**. Because proposals are drawn exactly from conditionals, all updates are automatically accepted.

Typical use:

- Bayesian hierarchical models
- Models with conjugate full conditionals

4.1.1 Relationship Between Monte Carlo, MCMC, and Gibbs Sampling

These three concepts are **not competing methods**, but rather form a **nested hierarchy** of ideas used for approximating expectations and probability distributions using randomness.



4.1.2 Summary Table

Method	Independent Samples	Uses Markov Chain	Accept–Reject Step	Typical Application
Monte Carlo	Yes	No	No	Direct simulation, integration

Method	Independent Samples	Uses Markov Chain	Accept–Reject Step	Typical Application
MCMC	No	Yes	Usually	Bayesian posterior sampling
Gibbs Sampler	No	Yes	No (always accept)	Bayesian models with tractable conditionals

i Key summary

- **MC** is the general idea of using randomness for approximation.
- **MCMC** is MC with dependent samples generated by a Markov chain.
- **Gibbs sampling** is a specific MCMC algorithm based on full conditional distributions.

4.2 Monte Carlo Method

In the previous chapter, we obtained the following posterior distributions for the birth rates of women without and with bachelor's degrees:

$$\theta_1 \mid \sum_{i=1}^{111} Y_{i,1} = 217 \sim \text{Gamma}(219, 112),$$

$$\theta_2 \mid \sum_{i=1}^{44} Y_{i,2} = 66 \sim \text{Gamma}(68, 45).$$

It was claimed that

$$\Pr(\theta_1 > \theta_2 \mid \text{data}) = 0.97.$$

How do we compute such a probability? From the previous chapter, since θ_1 and θ_2 are conditionally independent given the data y, we have

$$\Pr(\theta_1 > \theta_2 \mid y) = \int_0^{\infty} \int_0^{\theta_1} p(\theta_1 \mid y) p(\theta_2 \mid y)\, d\theta_2\, d\theta_1.$$

Substituting the gamma densities gives

$$\int_0^{\infty} \int_0^{\theta_1} \text{dgamma}(\theta_1; 219, 112)\, \text{dgamma}(\theta_2; 68, 45)\, d\theta_2\, d\theta_1.$$

This integral can be evaluated numerically. However, in realistic Bayesian models, such integrals quickly become high-dimensional and analytically intractable. This motivates **Monte Carlo (MC) methods**.

In Bayesian inference we repeatedly encounter integrals such as

$$\mathbb{E}[g(\theta) \mid y] = \int g(\theta)\, p(\theta \mid y)\, d\theta, \qquad \Pr(\theta \in A \mid y) = \int_A p(\theta \mid y)\, d\theta,$$

that are not available in closed form. MC replaces these integrals by averages of random draws.

> **Key Idea**
>
> Replace an intractable integral by an empirical mean.

4.2.1 MC Approximation

Suppose we wish to compute

$$\mathbb{E}[g(\theta) \mid y] = \int g(\theta)\, p(\theta \mid y)\, d\theta.$$

If we can generate independent samples

$$\theta^{(1)}, \ldots, \theta^{(S)} \sim p(\theta \mid y),$$

then we approximate the expectation by

$$\frac{1}{S} \sum_{s=1}^{S} g(\theta^{(s)}).$$

This is called a **Monte Carlo approximation**. By the Law of Large Numbers,

$$\frac{1}{S} \sum_{s=1}^{S} g(\theta^{(s)}) \;\longrightarrow\; \mathbb{E}[g(\theta) \mid y] = \int g(\theta) p(\theta \mid y) d\theta \quad \text{as } S \to \infty.$$

With the property above, we can calculate many quantities of interest about the posterior distribution. For example, suppose $\bar{\theta}$ is the average of $\{\theta^{(1)}, \dots, \theta^{(S)}\}$, then as $S \to \infty$:

- $\bar{\theta} \to \mathbb{E}[\theta \mid y]$,
- $\frac{1}{S-1} \sum_{s=1}^{S} (\theta^{(s)} - \bar{\theta})^2 \to \text{Var}(\theta \mid y)$.
- $\frac{1}{S} \sum_{s=1}^{S} \mathbf{1}\{\theta^{(s)} \in A\} \to \Pr(\theta \in A \mid y)$.
- the empirical distribution of $\{\theta^{(1)}, \dots, \theta^{(S)}\}$ converges to $p(\theta \mid y)$.
- the sample median converges to the posterior median $\theta_{1/2}$.
- the sample α-quantile converges to θ_α.

Key message

Almost any aspect of a posterior distribution can be approximated arbitrarily well using a sufficiently large Monte Carlo sample.

- working in high dimensions
- requires only the ability to simulate
- avoids symbolic integration.
- Scales to complex hierarchical models.

Thus Monte Carlo sampling allows us to approximate:

- posterior means,
- posterior variances,
- posterior probabilities,
- credible intervals,
- many more

To approximate

$$\Pr(\theta_1 > \theta_2 \mid y),$$

we can:

0. Choose a (large) number of samples S (e.g., $S = 10,000$).
1. Draw $\theta_1^{(s)} \sim \text{Gamma}(219, 112)$.
2. Draw $\theta_2^{(s)} \sim \text{Gamma}(68, 45)$.
3. Compute the indicator
$$I^{(s)} = \mathbf{1}\{\theta_1^{(s)} > \theta_2^{(s)}\}.$$

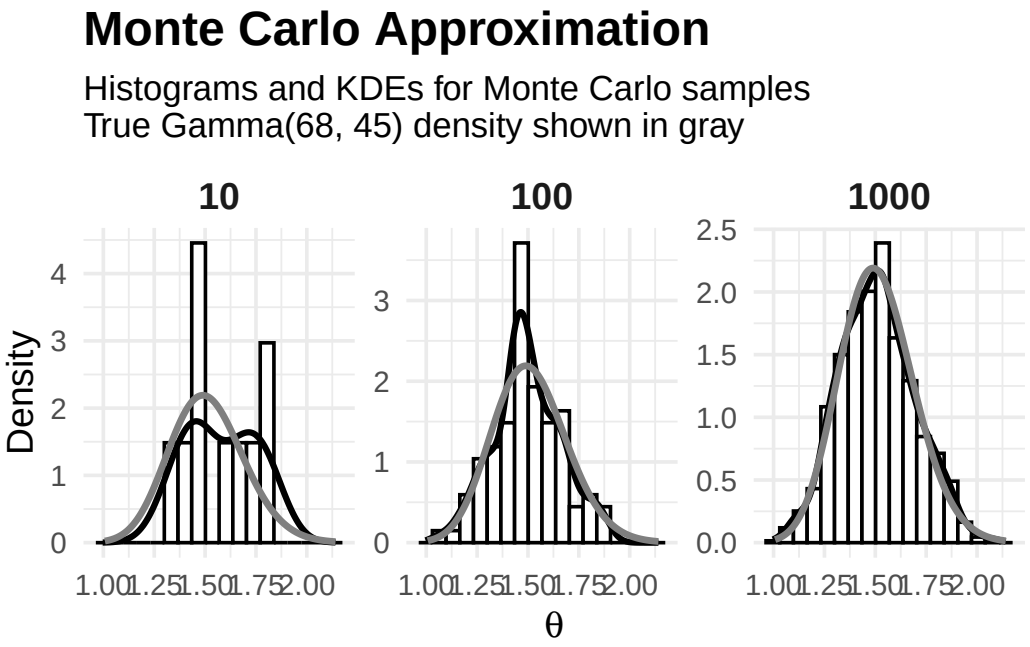
Then

$$\Pr(\theta_1 > \theta_2 \mid y) \approx \frac{1}{S} \sum_{s=1}^{S} I^{(s)}.$$

This avoids evaluating any double integrals. This is the foundation of modern Bayesian computation.

```
# Figure 4.1 - Monte Carlo Approximation (Gamma(68,45))
# Histograms + KDEs for S = 10, 100, 1000; true density in gray

set.seed(8310)
library(ggplot2)

# Posterior: Gamma(shape=68, rate=45)
shape_post <- 68
rate_post  <- 45

# MC samples
S_list <- c(10, 100, 1000)
mc_df <- do.call(rbind, lapply(S_list, function(S) {
  data.frame(theta = rgamma(S, shape = shape_post, rate = rate_post),
             S = factor(S, levels = S_list))
}))

# Grid for true density (choose a sensible range around the mass)
xgrid <- seq(
  qgamma(0.001, shape = shape_post, rate = rate_post),
  qgamma(0.999, shape = shape_post, rate = rate_post),
  length.out = 600
)

true_df <- data.frame(
  theta = xgrid,
  dens  = dgamma(xgrid, shape = shape_post, rate = rate_post)
)

# Plot
ggplot(mc_df, aes(x = theta)) +
  # histogram (density scale so it overlays with densities)
  geom_histogram(aes(y = after_stat(density)),
                 bins = 18, color = "black", fill = "white") +
  # KDE from MC samples
  geom_density(linewidth = 1.1) +
  # True density (gray)
  geom_line(data = true_df, aes(x = theta, y = dens),
            linewidth = 1.2, color = "gray50") +
  facet_wrap(~ S, nrow = 1, scales = "free_y") +
```

```
  labs(
    title = "Monte Carlo Approximation",
  subtitle = paste(
    "Histograms and KDEs for Monte Carlo samples",
    "True Gamma(68, 45) density shown in gray",
    sep = "\n"
  ),
    x = expression(theta),
    y = "Density"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    plot.title = element_text(size = 18, face = "bold"),
    plot.subtitle = element_text(size = 13),
    strip.text = element_text(size = 14, face = "bold")
  )
```



4.2.1.1 Numerical Evaluation

We now compare MC approximations to quantities that can be computed analytically in this conjugate example.

Suppose

$$Y_1, \ldots, Y_n \mid \theta \sim \text{Poisson}(\theta), \quad \theta \sim \text{Gamma}(a, b).$$

After observing $y_1, \ldots, y_n$ with $\sum y_i = sy$ and sample size n, the posterior distribution is

$$\theta \mid y \sim \text{Gamma}(a + sy,\ b + n).$$

For the birth-rate example:

- $a = 2$
- $b = 1$
- $sy = 66$
- $n = 44$

Posterior:

$$\theta \mid y \sim \text{Gamma}(68, 45).$$

Posterior mean:

$$\mathbb{E}[\theta \mid y] = \frac{a + sy}{b + n} = \frac{68}{45} = 1.51.$$

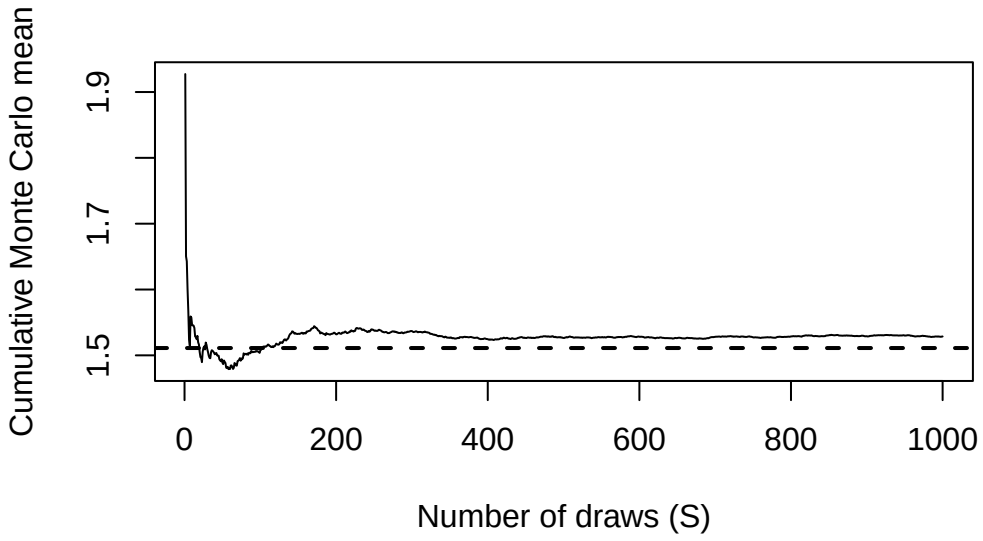
```r
set.seed(8310)

## Posterior parameters
a  <- 2
b  <- 1
sy <- 66
n  <- 44

shape_post <- a + sy
rate_post  <- b + n

## Exact quantities
mean_exact <- shape_post / rate_post
p_exact    <- pgamma(1.75, shape = shape_post, rate = rate_post)
ci_exact   <- qgamma(c(0.025, 0.975),
                     shape = shape_post,
                     rate  = rate_post)
```

```r
## Monte Carlo samples
theta_mc10   <- rgamma(10,   shape_post, rate_post)
theta_mc100  <- rgamma(100,  shape_post, rate_post)
theta_mc1000 <- rgamma(1000, shape_post, rate_post)

## Function to summarize MC output
mc_summary <- function(theta_sample) {
  c(
    Mean        = mean(theta_sample),
    Prob_less   = mean(theta_sample < 1.75),
    CI_lower    = quantile(theta_sample, 0.025),
    CI_upper    = quantile(theta_sample, 0.975)
  )
}

## Build comparison table
results <- rbind(
  Exact   = c(Mean      = mean_exact,
              Prob_less = p_exact,
              CI_lower  = ci_exact[1],
              CI_upper  = ci_exact[2]),

  MC_10   = mc_summary(theta_mc10),
  MC_100  = mc_summary(theta_mc100),
  MC_1000 = mc_summary(theta_mc1000)
)

round(results, 4)
```

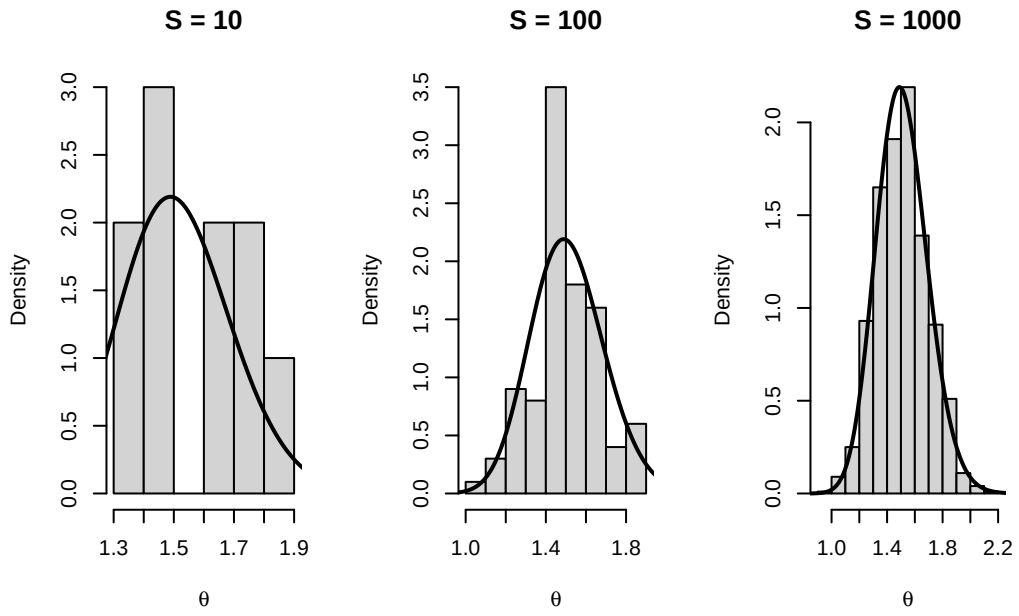
```
          Mean Prob_less CI_lower CI_upper
Exact   1.5111    0.8998   1.1734   1.8908
MC_10   1.5782    0.8000   1.3408   1.8149
MC_100  1.4968    0.9300   1.1801   1.8349
MC_1000 1.5102    0.8850   1.1578   1.8817
```

```r
Smax <- 1000
theta_seq <- rgamma(Smax, shape = shape_post, rate = rate_post)
cum_mean  <- cumsum(theta_seq) / seq_along(theta_seq)

plot(cum_mean, type="l",
     xlab="Number of draws (S)",
```

```
     ylab="Cumulative Monte Carlo mean")
abline(h = mean_exact, lty = 2, lwd = 2)
```




```
xgrid <- seq(0.5, 2.5, length.out = 400)
true_pdf <- dgamma(xgrid, shape = shape_post, rate = rate_post)

par(mfrow=c(1,3))

for (S in c(10,100,1000)) {
  x <- get(paste0("theta_mc", S))
  hist(x, prob=TRUE,
       main=paste0("S = ", S),
       xlab=expression(theta),
       border="black")
  lines(xgrid, true_pdf, lwd=2)
}
```



MC offers much more

There are much more about the MC method**

- Variance reduction methods
- Antithetic variates
- Control variables
- Importance Sampling
- Stratified Sampling
- Stratified Importance Sampling
- etc...

You may refer to my notes in Chapter 4 in the Computational Methods in Statistics Course.

4.2.2 MC for predictive distribution and Sampling from it

As discussed earlier, the **predictive distribution** of a future random variable $\tilde{Y}$ is the probability distribution that reflects uncertainty about $\tilde{Y}$ after accounting for both:

- known quantities (conditioned on observed data), and
- unknown quantities (integrated out).

4.2.2.1 Sampling Model vs Predictive Model

Suppose $\tilde{Y}$ denotes the number of children for a randomly selected woman aged 40 with a college degree.

If the true mean birthrate θ were known, uncertainty about $\tilde{Y}$ would be described by the **sampling model**

$$\Pr(\tilde{Y} = \tilde{y} \mid \theta) = p(\tilde{y} \mid \theta) = \frac{\theta^{\tilde{y}} e^{-\theta}}{\tilde{y}!},$$

that is,

$$\tilde{Y} \mid \theta \sim \text{Poisson}(\theta).$$

In practice, however, θ is unknown. Therefore, predictions must account for uncertainty in θ.

4.2.2.2 Prior Predictive Distribution

If no data have been observed, predictions are obtained by integrating out θ using the prior distribution:

$$\Pr(\tilde{Y} = \tilde{y}) = \int p(\tilde{y} \mid \theta)\, p(\theta)\, d\theta.$$

This is called the **prior predictive distribution**.

For a Poisson model with a Gamma prior,

$$\theta \sim \text{Gamma}(a, b),$$

the prior predictive distribution of $\tilde{Y}$ is

$$\tilde{Y} \sim \text{Negative Binomial}(a, b).$$

4.2.2.3 Posterior Predictive Distribution

After observing data $Y_1 = y_1, \ldots, Y_n = y_n$, the relevant predictive distribution for a new observation is

$$\Pr(\tilde{Y} = \tilde{y} \mid Y_1 = y_1, \ldots, Y_n = y_n) = \int p(\tilde{y} \mid \theta)\, p(\theta \mid y_1, \ldots, y_n)\, d\theta.$$

This distribution is called the **posterior predictive distribution**.

For the Poisson–Gamma model, the posterior distribution is

$$\theta \mid y_1, \ldots, y_n \sim \text{Gamma}\left(a + \sum_{i=1}^{n} y_i,\ b + n\right),$$

and the posterior predictive distribution is again Negative Binomial.

4.2.2.4 MC Sampling from the Posterior Predictive Distribution

In many models, the posterior predictive distribution cannot be evaluated analytically. However, it can often be **sampled using MC methods**.

The idea is simple:

1. Draw $\theta^{(s)} \sim p(\theta \mid y_1, \dots, y_n)$
2. Draw $\tilde{Y}^{(s)} \sim p(\tilde{y} \mid \theta^{(s)})$
3. Repeat for $s = 1, \dots, S$

This produces samples

$$\tilde{Y}^{(1)}, \dots, \tilde{Y}^{(S)} \sim p(\tilde{y} \mid y_1, \dots, y_n),$$

which approximate the posterior predictive distribution.

Suppose we observe two independent groups with Poisson data:

- Group 1: $\sum Y_{i,1} = 217$, $n_1 = 111$
- Group 2: $\sum Y_{i,2} = 66$, $n_2 = 44$

With a common prior

$$\theta_k \sim \text{Gamma}(a, b), \quad k = 1, 2,$$

the posterior distributions are

$$\theta_1 \mid \mathbf{y}_1 \sim \text{Gamma}(a + 217,\ b + 111),$$
$$\theta_2 \mid \mathbf{y}_2 \sim \text{Gamma}(a + 66,\ b + 44).$$

Because θ_1 and θ_2 are **posterior independent**, posterior predictive sampling proceeds independently for each group:

$$\theta_1^{(s)} \sim p(\theta_1 \mid \mathbf{y}_1), \quad \tilde{Y}_1^{(s)} \sim \text{Poisson}(\theta_1^{(s)}),$$
$$\theta_2^{(s)} \sim p(\theta_2 \mid \mathbf{y}_2), \quad \tilde{Y}_2^{(s)} \sim \text{Poisson}(\theta_2^{(s)}).$$

Using Monte Carlo samples $\{\tilde{Y}_1^{(s)}, \tilde{Y}_2^{(s)}\}$, we can approximate quantities such as

$$\Pr(\tilde{Y}_1 > \tilde{Y}_2 \mid \text{data}) \approx \frac{1}{S} \sum_{s=1}^{S} \mathbb{I}(\tilde{Y}_1^{(s)} > \tilde{Y}_2^{(s)}).$$

More generally, MC samples from the posterior predictive distribution allow us to approximate:

- predictive probabilities,
- predictive expectations,
- quantiles and credible intervals,
- functions of future observations.

This flexibility is one of the main strengths of MC methods in Bayesian analysis.

4.3 Posterior inference for arbitrary functions

Often we care about the posterior distribution of some *computable* function $g(\theta)$ of a parameter θ. For example, in the binomial model we may be interested in the **log-odds**

$$\log \text{odds}(\theta) = \log\Big(\frac{\theta}{1-\theta}\Big) = \gamma.$$

If we can generate posterior draws $\{\theta^{(1)}, \theta^{(2)}, ...\}$ from $p(\theta \mid y_1, \dots, y_n)$, then the law of large numbers implies that Monte Carlo averages converge to posterior expectations. For instance,

$$\frac{1}{S}\sum_{s=1}^{S} \log\Big(\frac{\theta^{(s)}}{1-\theta^{(s)}}\Big) \;\longrightarrow\; \mathbb{E}\Big[\log\Big(\frac{\theta}{1-\theta}\Big)\Big| y_1, \dots, y_n\Big].$$

More generally, we can approximate the entire posterior distribution of

$$\gamma = g(\theta) = \log\Big(\frac{\theta}{1-\theta}\Big)$$

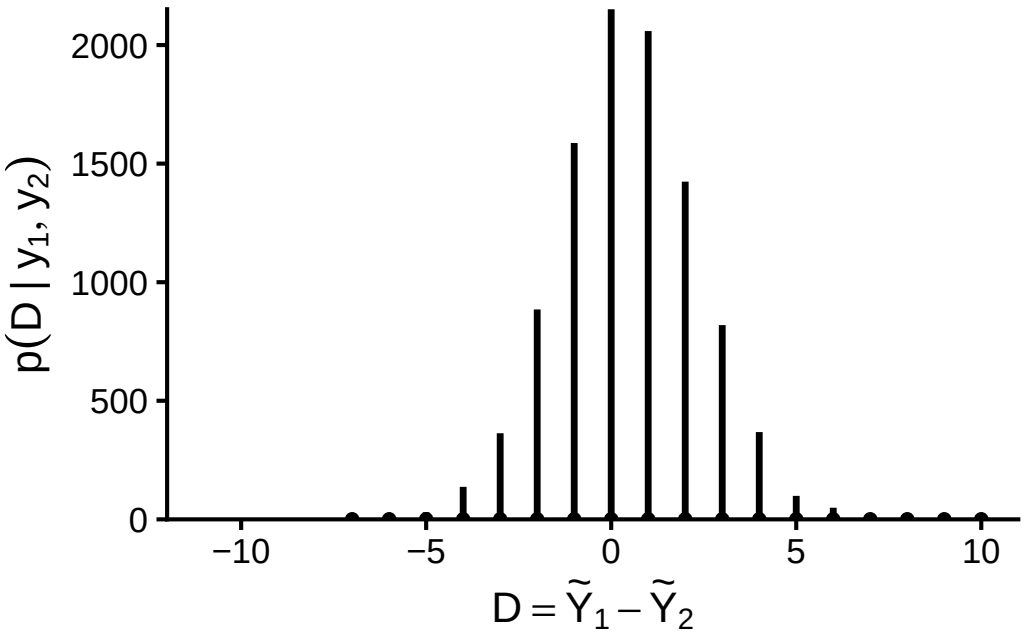
by transforming posterior samples:

$$\begin{aligned}
\theta^{(1)} &\sim p(\theta \mid y_1, \dots, y_n), \quad \gamma^{(1)} = g(\theta^{(1)}),\\
\theta^{(2)} &\sim p(\theta \mid y_1, \dots, y_n), \quad \gamma^{(2)} = g(\theta^{(2)}),\\
&\vdots\\
\theta^{(S)} &\sim p(\theta \mid y_1, \dots, y_n), \quad \gamma^{(S)} = g(\theta^{(S)}).
\end{aligned}$$

The collection $\{\gamma^{(1)}, \dots, \gamma^{(S)}\}$ approximates the posterior distribution of γ, so we can estimate posterior means, variances, quantiles, and credible intervals for γ directly from these transformed draws.

4.3.1 Posterior Predictive Model Checking

```r
library(ggplot2)

set.seed(8670)

## --- Birth-rate example (Hoff): posterior predictive for D = Y~1 - Y~2 ---
a  <- 2;  b  <- 1
n1 <- 111; sy1 <- 217   # less than bachelor's
n2 <- 44;  sy2 <- 66    # bachelor's or higher

shape1 <- a + sy1; rate1 <- b + n1
shape2 <- a + sy2; rate2 <- b + n2

S <- 10000  # using 10,000 makes the y-axis peak around ~2000 (counts), like Fig 4.5

theta1 <- rgamma(S, shape = shape1, rate = rate1)
theta2 <- rgamma(S, shape = shape2, rate = rate2)

y1 <- rpois(S, lambda = theta1)
y2 <- rpois(S, lambda = theta2)

D <- y1 - y2

df <- as.data.frame(table(D))
df$D <- as.integer(as.character(df$D))
df$Freq <- as.numeric(df$Freq)

xlim <- c(-12, 11)

ggplot(df, aes(x = D, y = Freq)) +
  geom_segment(aes(xend = D, y = 0, yend = Freq),
               linewidth = 1.2, color = "black") +
  geom_point(aes(y = 0), size = 1.6, color = "black") +
  coord_cartesian(xlim = xlim, expand = FALSE) +
  labs(
    x = expression(D == tilde(Y)[1] - tilde(Y)[2]),
    y = expression(p(D~"|"~y[1],y[2]))
  ) +
  theme_classic(base_size = 16)
```



Posterior predictive model checking assesses whether a fitted Bayesian model can plausibly reproduce key features of the observed data. Back to the example we studied before, we focus on women aged 40 without a college degree. The empirical distribution of the number of children for these women, together with the corresponding posterior predictive distribution.

In this sample of size $n = 111$, the number of women with exactly two children is $y_{\text{obs}} = 38$, which is twice the number of women with exactly one child.

In contrast, the posterior predictive distribution (shown in gray) suggests that sampling a woman with two children is slightly less likely than sampling a woman with one child, with probabilities approximately

$$0.27 \quad \text{vs.} \quad 0.28.$$

These two distributions appear to be in conflict: if the observed data contain twice as many women with two children as with one child, why does the model predict otherwise?

Possible Explanations

One explanation is sampling variability. The empirical distribution of a finite sample does not necessarily match the true population distribution, and with moderate sample sizes, random fluctuations can be substantial. A smooth population distribution can easily produce a bumpy empirical histogram.

An alternative explanation is model misspecification. In particular, the Poisson model cannot capture certain features of the data. There is no Poisson distribution with a sharp peak at $y = 2$, whereas the empirical distribution shows exactly such behaviour.

These explanations can be investigated systematically using Monte Carlo simulation. Define the **discrepancy statistic**

$$t(y) = \frac{\#y_i = 2}{\#y_i = 1},$$

the ratio of the number of women with two children to the number with one child.

For the observed data, $t(y_{\text{obs}}) = 2$. To assess whether this value is surprising under the model, we examine the posterior predictive distribution of $t(\tilde{Y})$.

For each Monte Carlo iteration $s = 1, \ldots, S$:

1. Sample from the posterior
$$\theta^{(s)} \sim p(\theta \mid y_{\text{obs}})$$

2. Generate a posterior predictive dataset
$$\tilde{Y}^{(s)} = (\tilde{y}_1^{(s)}, \ldots, \tilde{y}_n^{(s)}), \quad \text{where } \tilde{y}_i^{(s)} \overset{\text{i.i.d.}}{\sim} \text{Poisson}(\theta^{(s)})$$

3. Compute the discrepancy
$$t^{(s)} = t(\tilde{Y}^{(s)})$$

This yields samples $\{t^{(1)}, \ldots, t^{(S)}\}$ from the posterior predictive distribution of $t(\tilde{Y})$.

```
## Prior parameters
a <- 2
b <- 1

## Data summary (no bachelor's degree group)
n  <- 111
sy <- 217   # sum(y_i)

## Storage
t_mc <- numeric(10000)
set.seed(8310)
for (s in 1:10000) {
  ## Draw from posterior
  theta <- rgamma(1, shape = a + sy, rate = b + n)

  ## Posterior predictive sample
  y_mc <- rpois(n, theta)

  ## Discrepancy statistic
  n1 <- sum(y_mc == 1)
  n2 <- sum(y_mc == 2)
```

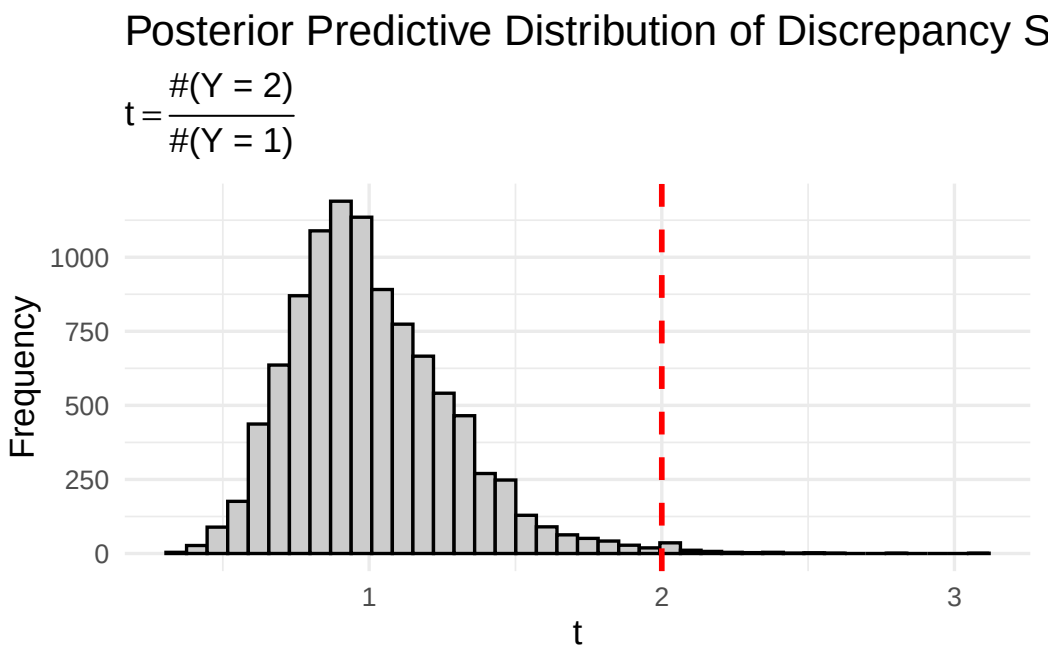
```r
  ## Avoid division by zero
  t_mc[s] <- ifelse(n1 > 0, n2 / n1, NA)
}

## Remove undefined values
t_mc <- t_mc[!is.na(t_mc)]
library(tibble)
library(tibble)
library(knitr)

summary_table <- tibble(
  Quantity = c(
    "Number of valid posterior predictive draws",
    "Number of draws with T   2.0",
    "Posterior predictive probability P(T   2.0)"
  ),
  Value = c(
    format(length(t_mc), big.mark = ","),
    format(sum(t_mc >= 2.0), big.mark = ","),
    formatC(mean(t_mc >= 2.0), format = "f", digits = 4)
  )
)

kable(
  summary_table,
  align = c("l", "r"),
  caption = "Posterior Predictive Summary for Discrepancy Statistic T"
)
```

Table 4.1: Posterior Predictive Summary for Discrepancy Statistic T

Quantity	Value
Number of valid posterior predictive draws	10,000
Number of draws with T 2.0	71
Posterior predictive probability P(T 2.0)	0.0071

```r
library(ggplot2)

df <- data.frame(t = t_mc)
```

```r
ggplot(df, aes(x = t)) +
  geom_histogram(
    aes(y = after_stat(count)),
    bins = 40,
    fill = "grey80",
    color = "black"
  ) +
  geom_vline(
    xintercept = 2,
    linewidth = 1,
    linetype = "dashed",
    color = "red"
  ) +
  labs(
    title = "Posterior Predictive Distribution of Discrepancy Statistic",
    subtitle = expression(
      t == frac("#(Y = 2)", "#(Y = 1)")
    ),
    x = expression(t),
    y = "Frequency"
  ) +
  theme_minimal(base_size = 13)
```



This Figure shows the posterior predictive distribution of $t(\tilde{Y})$, with the observed value $t(y_{\text{obs}}) = 2$ indicated by a vertical line. Out of 10,000 Monte Carlo samples, only about 0.71% produce values of

$$t(\tilde{Y}) \geq t(y_{\text{obs}}).$$

This indicates that the observed discrepancy is extremely unlikely under the fitted Poisson model.

Conclusion

The posterior predictive check suggests that the Poisson model is inadequate for these data. Although it matches the posterior mean reasonably well, it fails to reproduce important distributional features.

This does not imply that the model is useless for all inferential goals. However, if our goal is to accurately describe the distribution of family sizes, a more flexible model is needed.

Posterior predictive checks provide a principled, simulation-based tool for diagnosing such failures and guiding model refinement.

This Chapter borrows materials from Chapter 4 in Hoff (2009) and Chapter 4 in Computational Methods in Statistics Course

5 Gibbs Sampler

Leading objectives:

- understand why posterior approximation is needed beyond conjugate models
- learn how to approximate posteriors using discrete grids and Gibbs sampling
- understand how Gibbs sampling uses full conditional distributions to generate dependent posterior samples

5.1 Introduction

For many multi-parameter Bayesian models, the joint posterior distribution does not belong to a standard family (e.g., exponential family) and is therefore difficult to sample from it directly. However, it is often the case that **sampling from the full conditional distribution of each parameter is straightforward**.

In such situations, posterior approximation can be carried out using the **Gibbs sampler**, an iterative MC algorithm that constructs a **dependent sequence of parameter values** whose distribution converges to the target joint posterior distribution. Here, we introduce the Gibbs sampler in the context of the normal model with a **semi-conjugate prior**, and study how well it approximates the posterior distribution.

5.2 A Semi-conjugate Prior Distribution

For normal distribution, it may be modelled our uncertainty about the population mean θ as depending on the sampling variance σ^2 via

$$\theta \mid \sigma^2 \sim N\left(\mu_0,\ \frac{\sigma^2}{\kappa_0}\right).$$

This formulation ties the prior variance of θ to the sampling variability of the data, and μ_0 can be interpreted as representing κ_0 prior observations from the population.

In some settings this dependence is reasonable, but in others we may wish to specify prior uncertainty about θ *independently* of σ^2 denoted as $\theta \perp \sigma^2$, so that

$$p(\theta, \sigma^2) = p(\theta)\, p(\sigma^2).$$

One such specification is the following **semi-conjugate prior distribution**:

$$\theta \sim \text{Normal}(\mu_0, \tau_0^2), \qquad \frac{1}{\sigma^2} \sim \text{Gamma}\left(\frac{\nu_0}{2}, \frac{\nu_0 \sigma_0^2}{2}\right).$$

Posterior Distribution of $\theta \mid \sigma^2$

Note that, if

$$Y_1, \ldots, Y_n \mid \theta, \sigma^2 \overset{i.i.d.}{\sim} N(\theta, \sigma^2),$$

then the posterior distribution of θ is

$$\theta \mid \sigma^2, y_1, \ldots, y_n \sim \text{Normal}(\mu_n, \tau_n^2),$$

where

$$\mu_n = \frac{\mu_0/\tau_0^2 + n\bar{y}/\sigma^2}{1/\tau_0^2 + n/\sigma^2}, \qquad \tau_n^2 = \left(\frac{1}{\tau_0^2} + \frac{n}{\sigma^2}\right)^{-1}.$$

Those calculation may be found in Section 5.2 in Hoff (2009).

This conditional posterior distribution will form one step of the Gibbs sampler.

> **Key Takeaway**
>
> - The **joint posterior** of (θ, σ^2) is not available in closed form.
> - The **full conditional distributions** of $\theta \mid \sigma^2, y$ and $\sigma^2 \mid \theta, y$ *are* available in standard forms.
> - This structure makes the Gibbs sampler a natural and efficient tool for posterior approximation.

In the next section, we derive the full conditional distribution of σ^2 and combine the two conditional updates into a complete Gibbs sampling algorithm.

In the conjugate case where τ_0^2 is proportional to σ^2, we showed that the marginal posterior distribution

$$p(\sigma^2 \mid y_1, \ldots, y_n)$$

is an inverse-gamma distribution. In this setting, MC samples of (θ, σ^2) from the joint posterior distribution can be obtained by the following two-step procedure:

1. Sample
$$\sigma^{2(s)} \sim p(\sigma^2 \mid y_1, \dots, y_n),$$
which is an inverse-gamma distribution.

2. Sample
$$\theta^{(s)} \sim p(\theta \mid \sigma^{2(s)}, y_1, \dots, y_n),$$
which is a normal distribution.

   This approach works because both full conditional distributions are standard and easy to sample from.

However, when τ_0^2 is **not proportional** to σ^2, the marginal posterior distribution of the precision
$$\frac{1}{\sigma^2}$$
is **not** a gamma distribution, nor any other standard distribution from which we can easily sample. As a result, direct MC sampling from the marginal posterior is no longer straightforward, motivating the need for alternative approximation methods.

5.3 Discrete Approximations

Posterior Density Ratios

Let $\tilde{\sigma}^2 = 1/\sigma^2$ denote the precision. Recall that the posterior distribution of $(\theta, \tilde{\sigma}^2)$ is equal to the **joint distribution**
$$p(\theta, \tilde{\sigma}^2, y_1, \dots, y_n),$$
divided by $p(y_1, \dots, y_n)$, which does not depend on the parameters. Therefore, the **relative posterior probabilities** of two parameter values $(\theta_1, \tilde{\sigma}_1^2)$ and $(\theta_2, \tilde{\sigma}_2^2)$ are directly computable:
$$\begin{aligned}\frac{p(\theta_1, \tilde{\sigma}_1^2 \mid y_1, \dots, y_n)}{p(\theta_2, \tilde{\sigma}_2^2 \mid y_1, \dots, y_n)} &= \frac{p\left(\theta_1, \tilde{\sigma}_1^2, y_1, \dots, y_n\right) / p\left(y_1, \dots, y_n\right)}{p\left(\theta_2, \tilde{\sigma}_2^2, y_1, \dots, y_n\right) / p\left(y_1, \dots, y_n\right)} \\ &= \frac{p(\theta_1, \tilde{\sigma}_1^2, y_1, \dots, y_n)}{p(\theta_2, \tilde{\sigma}_2^2, y_1, \dots, y_n)}.\end{aligned}$$

Joint Distribution

The joint density can be written as
$$\begin{aligned}p(\theta, \tilde{\sigma}^2, y_1, \dots, y_n) &= p(\theta, \tilde{\sigma}^2)\, p(y_1, \dots, y_n \mid \theta, \tilde{\sigma}^2) \\ &= \text{Normal}(\theta \mid \mu_0, \tau_0^2) \times \text{Gamma}\left(\tilde{\sigma}^2 \mid \frac{\nu_0}{2}, \frac{\nu_0 \sigma_0^2}{2}\right) \\ &\quad \times \prod_{i=1}^{n} \text{Normal}\left(y_i \mid \theta, \frac{1}{\tilde{\sigma}^2}\right).\end{aligned}$$

All components of this joint density are standard distributions and therefore easy to evaluate numerically.

Discrete Posterior Approximation

A **discrete approximation** to the posterior distribution is obtained by evaluating relative posterior probabilities on a finite grid.

Let

- $\{\theta_1, \dots, \theta_G\}$ be a grid of values for θ;
- $\{\tilde{\sigma}_1^2, \dots, \tilde{\sigma}_H^2\}$ be a grid of values for $\tilde{\sigma}^2$.

At each grid point $(\theta_g, \tilde{\sigma}_h^2)$, compute

$$p(\theta_g, \tilde{\sigma}_h^2, y_1, \dots, y_n).$$

The **discrete joint posterior** is then defined by

$$\begin{aligned} p_D(\theta_g, \tilde{\sigma}_h^2 \mid y_1, \dots, y_n) &= \frac{p(\theta_g, \tilde{\sigma}_h^2 \mid y_1, \dots, y_n)}{\sum_{g'=1}^{G} \sum_{h'=1}^{H} p(\theta_{g'}, \tilde{\sigma}_{h'}^2 \mid y_1, \dots, y_n)} \\ &= \frac{p(\theta_g, \tilde{\sigma}_h^2, y_1, \dots, y_n) \,/ p(y_1, \dots, y_n)}{\sum_{g'=1}^{G} \sum_{h'=1}^{H} p(\theta_{g'}, \tilde{\sigma}_{h'}^2, y_1, \dots, y_n) \,/ p(y_1, \dots, y_n)} \\ &= \frac{p(\theta_g, \tilde{\sigma}_h^2, y_1, \dots, y_n)}{\sum_{g'=1}^{G} \sum_{h'=1}^{H} p(\theta_{g'}, \tilde{\sigma}_{h'}^2, y_1, \dots, y_n)}. \end{aligned}$$

This defines a valid joint probability distribution over

$$\theta \in \{\theta_1, \dots, \theta_G\}, \qquad \tilde{\sigma}^2 \in \{\tilde{\sigma}_1^2, \dots, \tilde{\sigma}_H^2\},$$

since the probabilities sum to one. If the joint prior distribution were discrete on this grid, then p_D would be **exactly** the posterior distribution.

i Key Takeaway

A **discrete approximation** to the posterior distribution is obtained by constructing a grid over the parameter space and evaluating relative posterior probabilities on that grid. Specifically:

- choose grids $\{\theta_1, \dots, \theta_G\}$ and $\{\tilde{\sigma}_1^2, \dots, \tilde{\sigma}_H^2\}$ consisting of evenly spaced parameter values;
- evaluate $p(\theta_g, \tilde{\sigma}_h^2, y_1, \dots, y_n)$ for each grid point $(\theta_g, \tilde{\sigma}_h^2)$;
- assign posterior probabilities proportional to these values:

$$p(\theta_g, \tilde{\sigma}_h^2 \mid y_1, \ldots, y_n) \;\propto\; p(\theta_g, \tilde{\sigma}_h^2, y_1, \ldots, y_n).$$

This discrete approximation can then be normalized and used to compute posterior summaries such as means, variances, and credible regions.

Marginal Posterior Distributions

Marginal posterior distributions can be obtained by summing over the grid. For example, the marginal posterior of θ is

$$p_D(\theta_k \mid y_1, \ldots, y_n) = \sum_{h=1}^{H} p_D(\theta_k, \tilde{\sigma}_h^2 \mid y_1, \ldots, y_n).$$

A similar expression holds for $\tilde{\sigma}^2$.

We illustrate the discrete (grid-based) posterior approximation using the *midge data* from the previous chapter. The sample summaries are

$$n = 9, \qquad \bar{y} = 1.804, \qquad s^2 = 0.017.$$

In the conjugate normal–inverse-gamma setup, the prior variance of θ is tied to the sampling variance (through σ^2/κ_0). When s^2 is very small, this can unintentionally force the prior uncertainty about θ to be unrealistically small. The **semiconjugate** prior avoids this coupling.

We use the semiconjugate prior

$$\theta \sim \text{Normal}(\mu_0, \tau_0^2), \qquad \tilde{\sigma}^2 = 1/\sigma^2 \sim \text{Gamma}\left(\frac{\nu_0}{2}, \frac{\nu_0 \sigma_0^2}{2}\right),$$

with

$$\mu_0 = 1.9, \qquad \tau_0 = 0.95, \qquad \nu_0 = 1, \qquad \sigma_0^2 = 0.01.$$

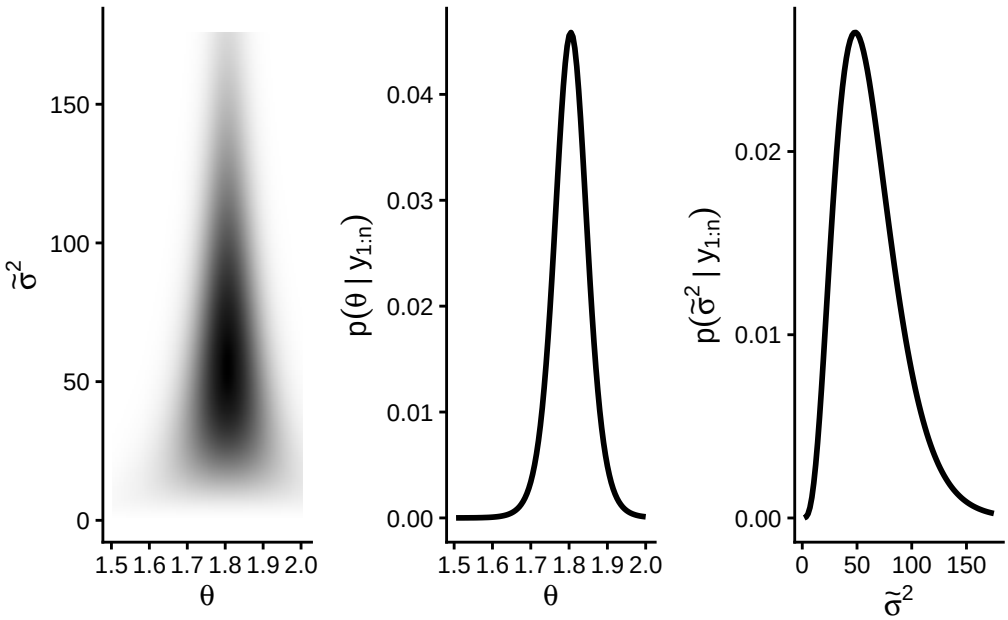
The joint posterior $p(\theta, \tilde{\sigma}^2 \mid y_1, \ldots, y_n)$ is evaluated on a 100×100 grid (evenly spaced values of θ and $\tilde{\sigma}^2$) and then normalized to form $p_D(\theta, \tilde{\sigma}^2 \mid y)$. The **joint** discrete approximation corresponds to the first panel of the Figure below, while the **marginals** are obtained by summation, e.g.

$$p_D(\theta_k \mid y_1, \ldots, y_n) = \sum_{h=1}^{H} p_D(\theta_k, \tilde{\sigma}_h^2 \mid y_1, \ldots, y_n),$$

which yields the second and third panels in the Figure below.

```r
library(ggplot2)
library(dplyr)
library(patchwork)
library(scales)

## ----------------------------
## Data and semiconjugate prior
## ----------------------------
y  <- c(1.64,1.70,1.72,1.74,1.82,1.82,1.82,1.90,2.08)
n  <- length(y)

mu0     <- 1.9
tau0_sq <- 0.95^2
nu0     <- 1
s20     <- 0.01

## ----------------------------
## Grids: theta and precision (tilde sigma^2 = 1/sigma^2)
## ----------------------------
G <- 100
H <- 100
mean.grid <- seq(1.505, 2.00, length.out = G)
prec.grid <- seq(1.75, 175,  length.out = H)

post.grid <- matrix(NA_real_, nrow = G, ncol = H)

## ----------------------------
## Discrete joint posterior on the grid
## p(theta, prec | y)  p(theta) p(prec)  N(y_i | theta, 1/prec)
## ----------------------------
for (g in 1:G) {
  for (h in 1:H) {
    post.grid[g, h] <-
      dnorm(mean.grid[g], mu0, sqrt(tau0_sq)) *
      dgamma(prec.grid[h], shape = nu0/2, rate = s20*nu0/2) *
      prod(dnorm(y, mean.grid[g], sd = 1 / sqrt(prec.grid[h])))
  }
}
post.grid <- post.grid / sum(post.grid)

## ----------------------------
## Data frame + marginals
```

```r
## ----------------------------
post_df <- expand.grid(theta = mean.grid, prec = prec.grid)
post_df$prob <- as.vector(post.grid)

theta_marg <- post_df %>%
  group_by(theta) %>%
  summarise(prob = sum(prob), .groups = "drop")

prec_marg <- post_df %>%
  group_by(prec) %>%
  summarise(prob = sum(prob), .groups = "drop")

## ----------------------------
## Plots (Figure 6.1 style)
## ----------------------------
p_joint <- ggplot(post_df, aes(theta, prec, fill = prob)) +
  geom_raster(interpolate = TRUE) +
  scale_fill_gradient(low = "white", high = "black",
                      trans = "sqrt", labels = label_number()) +
  labs(x = expression(theta),
       y = expression(tilde(sigma)^2)) +
  theme_classic() +
  theme(legend.position = "none")

p_theta <- ggplot(theta_marg, aes(theta, prob)) +
  geom_line(linewidth = 1) +
  labs(x = expression(theta),
       y = expression(p(theta~"|"~y[1:n]))) +
  theme_classic()

p_prec <- ggplot(prec_marg, aes(prec, prob)) +
  geom_line(linewidth = 1) +
  labs(x = expression(tilde(sigma)^2),
       y = expression(p(tilde(sigma)^2~"|"~y[1:n]))) +
  theme_classic()

p_joint + p_theta + p_prec
```



Remarks

- Discrete approximations are conceptually simple and transparent.
- They are feasible only in **low-dimensional parameter spaces**.
- As the dimension increases, grid-based methods become infeasible.
- For higher-dimensional models, simulation-based methods such as the **Gibbs sampler** become essential.
- This motivates **Markov chain Monte Carlo methods**, such as the **Gibbs sampler**, which we introduce soon.

5.4 Sampling from the Conditional Distribution

Suppose, for the moment, that the value of θ were known. The conditional distribution of $\tilde{\sigma}^2$ given θ and $\{y_1, \dots, y_n\}$ satisfies

$$\begin{aligned} p(\tilde{\sigma}^2 \mid \theta, y_1, \dots, y_n) &\propto p\left(y_1, \dots, y_n, \theta, \tilde{\sigma}^2\right) \\ &= p(y_1, \dots, y_n \mid \theta, \tilde{\sigma}^2)\, p(\tilde{\sigma}^2) p(\theta \mid \tilde{\sigma}^2) \end{aligned}$$

If θ and σ^2 are independent *a priori*, then $p(\theta, \tilde{\sigma}^2) = p(\theta)p(\tilde{\sigma}^2)$ and $p\left(\theta \mid \tilde{\sigma}^2\right) = p(\theta)$. Note that,

now, since $p(\theta)$ does have have $\tilde{\sigma}^2$ involved, so it can be treated like a constant in calculating the posterior for $\tilde{\sigma}^2$. Then

$$\begin{aligned} p\left(\tilde{\sigma}^2 \mid \theta, y_1, \ldots, y_n\right) \propto & p\left(y_1, \ldots, y_n \mid \theta, \tilde{\sigma}^2\right) p\left(\tilde{\sigma}^2\right) \\ & \propto \left(\left(\tilde{\sigma}^2\right)^{n/2} \exp\left\{-\tilde{\sigma}^2 \sum_{i=1}^{n}\left(y_i-\theta\right)^2/2\right\}\right) \times \\ & \quad \left(\left(\tilde{\sigma}^2\right)^{\nu_0/2-1} \exp\left\{-\tilde{\sigma}^2 \nu_0 \sigma_0^2/2\right\}\right) \\ & = \left(\tilde{\sigma}^2\right)^{(\nu_0+n)/2-1} \times \exp\left\{-\tilde{\sigma}^2 \times \left[\nu_0\sigma_0^2 + \sum\left(y_i-\theta\right)^2\right]/2\right\}. \end{aligned}$$

Alternatively, if one want to use σ^2 instead, the normal sampling model and the semi-conjugate prior,

$$\begin{aligned} p(\sigma^2 \mid \theta, y_1, \ldots, y_n) &\propto (\sigma^2)^{-n/2} \exp\left\{-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i-\theta)^2\right\} \times (\sigma^2)^{-\nu_0/2-1} \exp\left\{-\frac{\nu_0\sigma_0^2}{2\sigma^2}\right\} \\ &= (\sigma^2)^{-(\nu_0+n)/2-1} \exp\left\{-\frac{1}{2\sigma^2}\left[\nu_0\sigma_0^2 + \sum_{i=1}^{n}(y_i-\theta)^2\right]\right\}. \end{aligned}$$

This is the kernel of an inverse-gamma distribution. Therefore,

$$\sigma^2 \mid \theta, y_1, \ldots, y_n \sim \text{Inverse-Gamma}\left(\frac{\nu_n}{2}, \frac{\nu_n\sigma_n^2(\theta)}{2}\right),$$

where

$$\nu_n = \nu_0 + n, \qquad \sigma_n^2(\theta) = \frac{1}{\nu_n}\left[\nu_0\sigma_0^2 + \sum_{i=1}^{n}(y_i-\theta)^2\right].$$

Here,

$$s_n^2(\theta) = \frac{1}{n}\sum_{i=1}^{n}(y_i-\theta)^2$$

is the unbiased estimator of σ^2 when θ is known.

This result shows that we can easily sample from $p(\sigma^2 \mid \theta, y_1, \ldots, y_n)$.
Similarly, as shown earlier, we can sample from $p(\theta \mid \sigma^2, y_1, \ldots, y_n)$. But so far, we have not shown how to sample from the joint posterior distribution of (θ, σ^2), which is our ultimate goal, i.e., sampling from $p(\theta, \sigma^2 \mid y_1, \ldots, y_n)$.

5.4.1 Motivation for Gibbs sampling

Suppose we were given a single draw $\sigma^{2(1)}$ from the marginal posterior $p(\sigma^2 \mid y_1, \dots, y_n)$.
We could then sample

$$\theta^{(1)} \sim p(\theta \mid \sigma^{2(1)}, y_1, \dots, y_n),$$

and the pair $(\theta^{(1)}, \sigma^{2(1)})$ would be a draw from the joint posterior distribution of (θ, σ^2).

Moreover, $\theta^{(1)}$ can be regarded as a draw from the marginal posterior $p(\theta \mid y_1, \dots, y_n)$.
From this value, we can generate

$$\sigma^{2(2)} \sim p(\sigma^2 \mid \theta^{(1)}, y_1, \dots, y_n).$$

Since $\theta^{(1)}$ is drawn from the marginal distribution of θ and $\sigma^{2(2)}$ is drawn from the conditional distribution of σ^2 given $\theta^{(1)}$, the pair $(\theta^{(1)}, \sigma^{2(2)})$ is again a draw from the joint posterior distribution.

Repeating this alternating procedure suggests that the two full conditional distributions,

$$p(\theta \mid \sigma^2, y_1, \dots, y_n) \quad \text{and} \quad p(\sigma^2 \mid \theta, y_1, \dots, y_n),$$

can be used to generate samples from the joint posterior distribution.
This iterative sampling scheme is the basis of the **Gibbs sampler**, introduced in the next section, as long as we have the $\sigma^{2(1)}$ to start the process.

5.5 Gibbs Sampler

The distributions

$$p(\theta \mid \sigma^2, y_1, \dots, y_n) \quad \text{and} \quad p(\sigma^2 \mid \theta, y_1, \dots, y_n)$$

are called the **full conditional distributions** of θ and σ^2, respectively.
Each is the conditional distribution of one parameter given all the other parameters and the data.

We now formalize the iterative sampling idea introduced in the previous section.
Let

$$\phi^{(s)} = \left\{\theta^{(s)}, \sigma^{2(s)}\right\}$$

denote the current state of the parameters at iteration s.

Given $\phi^{(s)}$, a new state $\phi^{(s+1)}$ is generated by alternately sampling from the full conditional distributions:

1. **Update** θ:

$$\theta^{(s+1)} \sim p\big(\theta \mid \sigma^{2(s)}, y_1, \dots, y_n\big)\,.$$

2. **Update** σ^2:

$$\sigma^{2(s+1)} \sim p\big(\sigma^2 \mid \theta^{(s+1)}, y_1, \ldots, y_n\big)\,.$$

Repeating these two steps produces a sequence

$$\phi^{(1)}, \phi^{(2)}, \ldots$$

that forms a **Markov chain**, since each new state depends only on the immediately preceding state.

Under mild regularity conditions, the distribution of $\phi^{(s)}$ converges to the joint posterior distribution

$$p(\theta, \sigma^2 \mid y_1, \ldots, y_n)$$

as $s \to \infty$.
This iterative procedure is known as the **Gibbs sampler**, a special case of Markov chain Monte Carlo (MCMC) methods.

1. Sample

$$\theta^{(s+1)} \sim p\big(\theta \mid \tilde{\sigma}^{2(s)}, y_1, \ldots, y_n\big)\,.$$

2. Sample

$$\tilde{\sigma}^{2(s+1)} \sim p\big(\tilde{\sigma}^2 \mid \theta^{(s+1)}, y_1, \ldots, y_n\big)\,.$$

3. Set

$$\phi^{(s+1)} = \big\{\theta^{(s+1)}, \tilde{\sigma}^{2(s+1)}\big\}.$$

It generates a *dependent* sequence of parameter values

$$\big\{\phi^{(1)}, \phi^{(2)}, \ldots, \phi^{(S)}\big\}.$$

Below we give R code for implementing this Gibbs sampling scheme for the normal model with a semi-conjugate prior distribution.

This algorithm is called the *Gibbs sampler*, and generates a *dependent* sequence of parameter values

$$\{\phi^{(1)}, \phi^{(2)}, \ldots, \phi^{(S)}\}.$$

The R code below implements this sampling scheme for the normal model with a semiconjugate prior distribution.

```r
mu0 <- 1.9
t20 <- 0.95^2
s20 <- 0.01
nu0 <- 1

y <- c(1.64, 1.70, 1.72, 1.74, 1.82, 1.82, 1.82, 1.90, 2.08)

G <- 100
H <- 100
### data
mean.y <- mean(y)
var.y  <- var(y)
n      <- length(y)

### starting values
S   <- 1000
PHI <- matrix(nrow = S, ncol = 2)
PHI[1, ] <- phi <- c(mean.y, 1 / var.y)
###

### Gibbs sampling
set.seed(8310)
for (s in 2:S) {

  # generate a new theta value from its full conditional
  mun <- (mu0 / t20 + n * mean.y * phi[2]) / (1 / t20 + n * phi[2])
  t2n <- 1 / (1 / t20 + n * phi[2])
  phi[1] <- rnorm(1, mun, sqrt(t2n))

  # generate a new 1/sigma^2 value from its full conditional
  nun <- nu0 + n
  s2n <- (nu0 * s20 + (n - 1) * var.y + n * (mean.y - phi[1])^2) / nun
  phi[2] <- rgamma(1, nun / 2, nun * s2n / 2)

  PHI[s, ] <- phi
}
colnames(PHI) <- c("mu", "sigma2_inv")
head(PHI,3)
```

```
          mu sigma2_inv
[1,] 1.804444   59.24951
[2,] 1.802021   33.25979
```

```
[3,] 1.750617   54.98838
```

In this code, we have used the identity

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

$$= \sum_{i=1}^{n}\left[(y_i - \bar{y})^2 + 2(y_i - \bar{y})(\bar{y} - \theta) + (\bar{y} - \theta)^2\right]$$

$$= \sum_{i=1}^{n}(y_i - \bar{y})^2 + 0 + \sum_{i=1}^{n}(\bar{y} - \theta)^2$$

$$= (n-1)s^2 + n(\bar{y} - \theta)^2.$$

The reason for writing the code this way is that s^2 and $\bar{y}$ do not change with each new θ value. Therefore computing

$$(n-1)s^2 + n(\bar{y} - \theta)^2$$

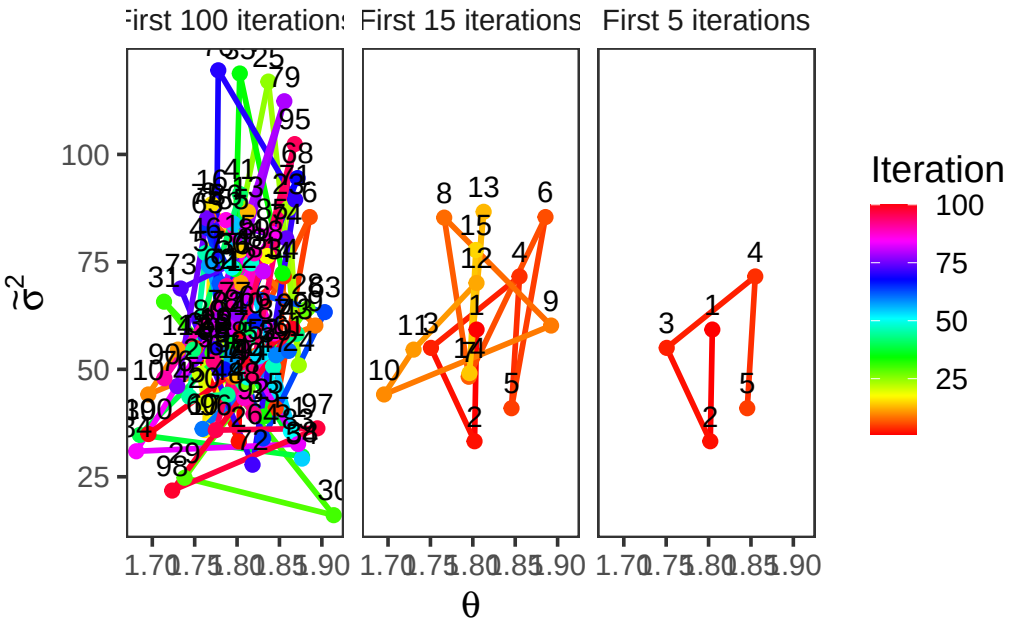
is faster than recomputing

$$\sum_{i=1}^{n}(y_i - \theta)^2$$

at each iteration.

```r
library(ggplot2)
library(dplyr)
library(knitr)

# Convert Gibbs output
df <- as.data.frame(PHI)
colnames(df) <- c("mu", "sigma2_inv")
df$iter <- 1:nrow(df)

# Create subsets
make_subset <- function(n) {
  df %>%
    filter(iter <= n) %>%
    mutate(panel = paste0("First ", n, " iterations"))
}
```

```
plot_df <- bind_rows(
  make_subset(5),
  make_subset(15),
  make_subset(100)
)

ggplot(plot_df, aes(mu, sigma2_inv)) +
  geom_path(aes(color = iter), linewidth = 1) +
  geom_point(aes(color = iter), size = 2) +
  geom_text(aes(label = iter), vjust = -0.7, size = 4) +
  facet_wrap(~panel, nrow = 1) +
  scale_color_gradientn(colors = rainbow(100)) +
  labs(
    x = expression(theta),
    y = expression(tilde(sigma)^2),
    color = "Iteration"
  ) +
  theme_bw(base_size = 14) +
  theme(
    panel.grid = element_blank(),
    strip.background = element_blank()
  )
```



Finally, we compute some empirical quantiles from the Gibbs samples.

```
# ### Credible interval for the population mean (theta)
# quantile(PHI[,1], c(.025, .5, .975))
# ### Credible interval for the population precision (1/sigma^2)
# quantile(PHI[,2], c(.025, .5, .975))
# ### Credible interval for the population standard deviation (sigma)
# quantile(1/sqrt(PHI[,2]), c(.025, .5, .975))

ci_table <- rbind(
  "  (mean)" = quantile(PHI[,1], c(.025, .5, .975)),
  "1/ ² (precision)" = quantile(PHI[,2], c(.025, .5, .975)),
  "  (standard deviation)" = quantile(1/sqrt(PHI[,2]), c(.025, .5, .975))
)

ci_table
```

```
                            2.5%        50%       97.5%
  (mean)                1.7056353  1.8044393   1.9042953
1/ ² (precision)       19.7469320 57.3331268 132.1650512
  (standard deviation)  0.0869845  0.1320679   0.2250375
```

```
colnames(ci_table) <- c("2.5%", "Median", "97.5%")
library(knitr)

kable(ci_table, digits = 4,
      caption = "Posterior credible intervals from Gibbs samples")
```

Table 5.1: Posterior credible intervals from Gibbs samples

	2.5%	Median	97.5%
(mean)	1.7056	1.8044	1.9043
1/ ² (precision)	19.7469	57.3331	132.1651
(standard deviation)	0.0870	0.1321	0.2250

The empirical distribution of the Gibbs samples closely matches the discrete approximation to the posterior distribution. Comparing Figures 6.1 and 6.3 shows that the simulated draws reproduce the shape of the posterior density. This suggests that the Gibbs sampler is an effective method for approximating

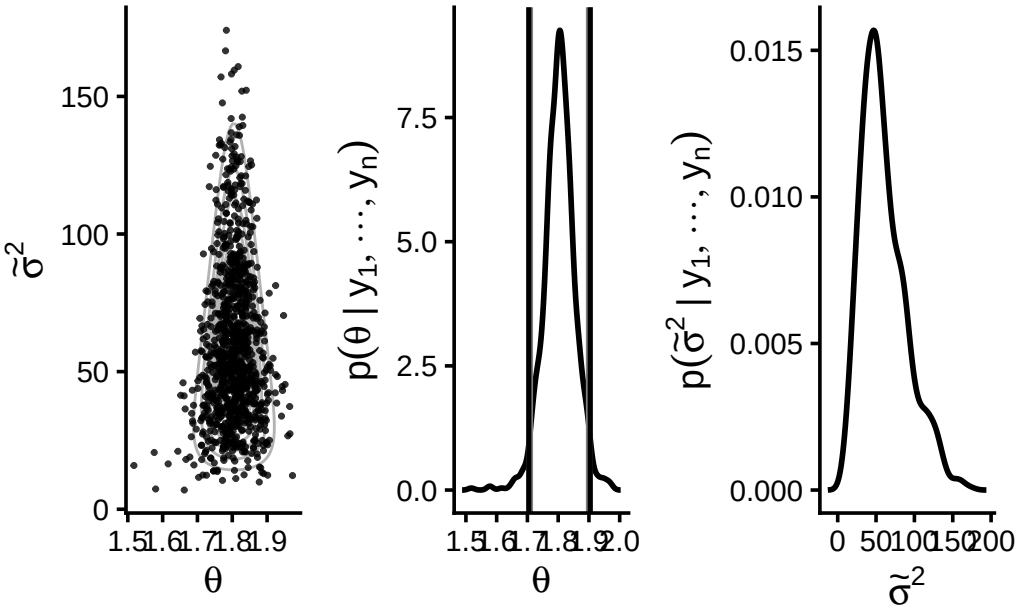
$$p(\theta, \sigma^2 \mid y_1, \ldots, y_n).$$

```r
library(ggplot2)
library(dplyr)
library(patchwork)

## ------------------------------------------------
## Data and prior
## ------------------------------------------------
y <- c(1.64, 1.70, 1.72, 1.74, 1.82, 1.82, 1.82, 1.90, 2.08)
n <- length(y)

mu0 <- 1.9
t20 <- 0.95^2
s20 <- 0.01
nu0 <- 1

## ------------------------------------------------
## Discrete approximation on a grid
## ------------------------------------------------
G <- 100
H <- 100

mean.grid <- seq(1.505, 2.00, length.out = G)
prec.grid <- seq(1.75, 175, length.out = H)

post.grid <- matrix(NA_real_, nrow = G, ncol = H)

for (g in 1:G) {
  for (h in 1:H) {
    post.grid[g, h] <-
      dnorm(mean.grid[g], mu0, sqrt(t20)) *
      dgamma(prec.grid[h], shape = nu0/2, rate = nu0*s20/2) *
      prod(dnorm(y, mean.grid[g], sd = 1 / sqrt(prec.grid[h])))
  }
}
post.grid <- post.grid / sum(post.grid)

post_df <- expand.grid(theta = mean.grid, sigma2_inv = prec.grid)
post_df$prob <- as.vector(post.grid)

## ------------------------------------------------
## Gibbs sampler
## ------------------------------------------------
```

```r
mean.y <- mean(y)
var.y  <- var(y)

S <- 1000
PHI <- matrix(NA_real_, nrow = S, ncol = 2)
colnames(PHI) <- c("mu", "sigma2_inv")

phi <- c(mean.y, 1 / var.y)
PHI[1, ] <- phi

set.seed(1)
for (s in 2:S) {
  ## update theta
  mun <- (mu0 / t20 + n * mean.y * phi[2]) / (1 / t20 + n * phi[2])
  t2n <- 1 / (1 / t20 + n * phi[2])
  phi[1] <- rnorm(1, mun, sqrt(t2n))

  ## update precision = 1/sigma^2
  nun <- nu0 + n
  s2n <- (nu0 * s20 + (n - 1) * var.y + n * (mean.y - phi[1])^2) / nun
  phi[2] <- rgamma(1, nun / 2, rate = nun * s2n / 2)

  PHI[s, ] <- phi
}

phi_df <- as.data.frame(PHI)

## ------------------------------------------------
## Quantiles and t interval
## ------------------------------------------------
theta_q <- quantile(phi_df$mu, c(0.025, 0.975))

tt <- t.test(y)
t_ci <- tt$conf.int

## ------------------------------------------------
## Panel 1: samples over contours
## ------------------------------------------------
p1 <- ggplot() +
  geom_contour(
    data = post_df,
    aes(x = theta, y = sigma2_inv, z = prob),
```

```
    bins = 12,
    color = "grey70",
    linewidth = 0.5
  ) +
  geom_point(
    data = phi_df,
    aes(x = mu, y = sigma2_inv),
    size = 0.4,
    alpha = 0.8
  ) +
  labs(
    x = expression(theta),
    y = expression(tilde(sigma)^2)
  ) +
  theme_classic(base_size = 14)

## ------------------------------------------------
## Panel 2: density of theta
## ------------------------------------------------
dens_theta <- density(phi_df$mu)
dens_theta_df <- data.frame(x = dens_theta$x, y = dens_theta$y)

p2 <- ggplot(dens_theta_df, aes(x, y)) +
  geom_line(linewidth = 1) +
  geom_vline(xintercept = theta_q, color = "grey50", linewidth = 1.2) +
  geom_vline(xintercept = t_ci, color = "black", linewidth = 0.9) +
  labs(
    x = expression(theta),
    y = expression(p(theta ~ "|" ~ y[1], cdots, y[n]))
  ) +
  theme_classic(base_size = 14)

## ------------------------------------------------
## Panel 3: density of precision
## ------------------------------------------------
dens_prec <- density(phi_df$sigma2_inv)
dens_prec_df <- data.frame(x = dens_prec$x, y = dens_prec$y)

p3 <- ggplot(dens_prec_df, aes(x, y)) +
  geom_line(linewidth = 1) +
  labs(
    x = expression(tilde(sigma)^2),
```

```
    y = expression(p(tilde(sigma)^2 ~ "|" ~ y[1], cdots, y[n]))
  ) +
  theme_classic(base_size = 14)

## ------------------------------------------------
## Combine
## ------------------------------------------------
p1 + p2 + p3
```



In this Figure:

- left: Gibbs samples overlaid on contours of the discrete approximation
- middle: kernel density estimate of Gibbs samples of θ, with
- gray vertical lines = Gibbs 2.5% and 97.5% quantiles
- black vertical lines = classical t-interval for the mean
- right: kernel density estimate of Gibbs samples of $\tilde{\sigma}^2 = 1/\sigma^2$

5.6 Gibbs sampler using an R package

So far, we have implemented the Gibbs sampler by hand in R. In practice, a common alternative is to use a package interface to **JAGS**, such as `rjags` or a higher-level wrapper like `jagsUI`. These tools let us specify the model, data, and parameters to monitor, and then run the Gibbs

sampler automatically. Note that these packages require **JAGS** to be installed on your system. User Manual can be found Here.

5.6.1 Example: using `jagsUI`

```
pacman::p_load(rjags, jagsUI)

## data
y <- c(1.64, 1.70, 1.72, 1.74, 1.82, 1.82, 1.82, 1.90, 2.08)
n <- length(y)

data_jags <- list(
  y = y,
  n = n,
  mu0 = 1.9,
  tau0 = 0.95^2,
  nu0 = 1,
  s20 = 0.01
)

## JAGS model
model_string <- "
model {
  for (i in 1:n) {
    y[i] ~ dnorm(theta, tau)
  }

  theta ~ dnorm(mu0, 1 / tau0)
  tau ~ dgamma(nu0 / 2, nu0 * s20 / 2)

  sigma <- 1 / sqrt(tau)
}
"

# writeLines(model_string, con = "gibbs_model.jags")

## fit model
fit <- jagsUI::jags(
  data = data_jags,
  parameters.to.save = c('theta', 'tau', 'sigma'),
  model.file = textConnection(model_string),
```

```
  n.chains = 3,
  n.iter = 5000,
  n.burnin = 1000,
  n.thin = 2
)
```

```
Processing function input.......

Done.

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 9
   Unobserved stochastic nodes: 2
   Total graph size: 24

Initializing model

Adaptive phase.....
Adaptive phase complete


 Burn-in phase, 1000 iterations x 3 chains


Sampling from joint posterior, 4000 iterations x 3 chains


Calculating statistics.......

Done.
```

```
fit
```

```
JAGS output for model '4', generated by jagsUI.
Estimates based on 3 chains of 5000 iterations,
adaptation = 100 iterations (sufficient),
burn-in = 1000 iterations and thin rate = 2,
```

```
yielding 6000 total samples from the joint posterior.
MCMC ran for 0 minutes at time 2026-03-29 11:33:04.469096.

            mean     sd    2.5%     50%   97.5% overlap0     f Rhat n.eff
theta      1.804  0.048   1.708   1.804   1.901    FALSE 1.000    1  6000
tau       61.549 29.128  18.520  57.195 128.714    FALSE 1.000    1  6000
sigma      0.140  0.038   0.088   0.132   0.232    FALSE 1.000    1  6000
deviance -10.170  2.038 -12.204 -10.778  -4.600    FALSE 0.998    1  6000

Successful convergence based on Rhat values (all < 1.1).
Rhat is the potential scale reduction factor (at convergence, Rhat=1).
For each parameter, n.eff is a crude measure of effective sample size.

overlap0 checks if 0 falls in the parameter's 95% credible interval.
f is the proportion of the posterior with the same sign as the mean;
i.e., our confidence that the parameter is positive or negative.

DIC info: (pD = var(deviance)/2)
pD = 2.1 and DIC = -8.092
DIC is an estimate of expected predictive error (lower is better).
```

Other useful implementations

Besides coding the Gibbs sampler directly in R, several packages can be useful:

- `rjags` / `jagsUI`: convenient interfaces to JAGS for Gibbs sampling
- `nimble`: flexible for building and customizing MCMC algorithms
- `MCMCpack`: easy to use for standard Bayesian models
- `Stan` (`rstan` or `cmdstanr`): widely used for modern Bayesian computation, typically using Hamiltonian Monte Carlo rather than Gibbs sampling

For learning purposes, writing the Gibbs sampler by hand is often the most transparent approach. For larger or more complicated models, package-based implementations are usually more practical.

5.7 General Properties of Gibbs Sampler

Suppose the parameter vector is

$$\phi = (\phi_1, \phi_2, \dots, \phi_p)$$

and our target distribution is the posterior

$$p(\phi) = p(\phi_1, \dots, \phi_p).$$

For example, in the normal model we may have

$$\phi = (\theta, \sigma^2)$$

and the distribution of interest is

$$p(\theta, \sigma^2 \mid y_1, \dots, y_n).$$

Starting from an initial value

$$\phi^{(0)} = (\phi_1^{(0)}, \dots, \phi_p^{(0)}),$$

the Gibbs sampler generates a sequence of samples by updating each component from its **full conditional distribution**.

At iteration s:

1. Sample

$$\phi_1^{(s)} \sim p(\phi_1 \mid \phi_2^{(s-1)}, \phi_3^{(s-1)}, \dots, \phi_p^{(s-1)})$$

2. Sample

$$\phi_2^{(s)} \sim p(\phi_2 \mid \phi_1^{(s)}, \phi_3^{(s-1)}, \dots, \phi_p^{(s-1)})$$

$$\vdots$$

p. Sample

$$\phi_p^{(s)} \sim p(\phi_p \mid \phi_1^{(s)}, \phi_2^{(s)}, \dots, \phi_{p-1}^{(s)})$$

The algorithm produces a sequence of vectors

$$\phi^{(1)},\ \phi^{(2)},\ \dots,\ \phi^{(S)}$$

where

$$\phi^{(s)} = (\phi_1^{(s)}, \ldots, \phi_p^{(s)}).$$

Each new sample depends only on the previous sample:

$$p(\phi^{(s)} \mid \phi^{(0)}, \ldots, \phi^{(s-1)}) = p(\phi^{(s)} \mid \phi^{(s-1)}).$$

This property is called the **Markov property**, and the sequence

$$\{\phi^{(s)}\}$$

is therefore a **Markov chain**.

A sequence of random variables $\{\phi^{(s)}\}$ has the **Markov property** if the distribution of each new sample depends only on the previous sample, and not on any earlier samples. Formally, for all s,

$$p(\phi^{(s)} \mid \phi^{(0)}, \ldots, \phi^{(s-1)}) = p(\phi^{(s)} \mid \phi^{(s-1)}).$$

A **Markov chain** is a sequence of random variables $\{\phi^{(s)}\}$ that satisfies the Markov property. In other words, it is a sequence where the distribution of each new sample depends only on the immediately preceding sample.

Under mild regularity conditions, the stationary distribution of this Markov chain is the target posterior distribution

$$p(\phi \mid y_1, \ldots, y_n).$$

Consequently, after a sufficient number of iterations, the Gibbs samples can be used to approximate posterior quantities such as

$$E[g(\phi) \mid y].$$

Under mild regularity conditions, the Gibbs sampler converges to the target distribution. Specifically,

$$\Pr(\phi^{(s)} \in A) \longrightarrow \int_A p(\phi)\, d\phi, \qquad \text{as } s \to \infty.$$

In words, the sampling distribution of $\phi^{(s)}$ approaches the target distribution as the number of iterations increases, regardless of the starting value $\phi^{(0)}$ (although some starting values reach the target distribution faster than others).

More importantly, for most functions $g(\cdot)$,

$$\frac{1}{S}\sum_{s=1}^{S} g(\phi^{(s)}) \;\longrightarrow\; E[g(\phi)] = \int g(\phi)p(\phi)\,d\phi, \qquad \text{as } S \to \infty.$$

This means that we can approximate the expectation $E[g(\phi)]$ using the sample average of the Gibbs samples

$$\{g(\phi^{(1)}), \ldots, g(\phi^{(S)})\}.$$

This idea is analogous to MC integration. Because the samples come from a **Markov chain**, these methods are called

Markov Chain Monte Carlo (MCMC).

In the semiconjugate normal model, the Gibbs sampler produces samples

$$\{(\theta^{(1)}, \sigma^{2(1)}), \ldots, (\theta^{(1000)}, \sigma^{2(1000)})\}$$

that approximate the posterior distribution

$$p(\theta, \sigma^2 \mid y_1, \ldots, y_n).$$

For example,

$$E[\theta \mid y_1, \ldots, y_n] \approx \frac{1}{1000}\sum_{s=1}^{1000} \theta^{(s)} = 1.804.$$

Similarly, a posterior credible interval can be approximated by

$$\Pr(\theta \in [1.71, 1.90] \mid y_1, \ldots, y_n) \approx 0.95.$$

We will discuss practical aspects of MCMC in the next chapters.

Distinguishing parameter estimation from posterior approximation

Bayesian data analysis using MC methods often involves a confusing array of sampling procedures and probability distributions. It is therefore useful to distinguish the **statistical modelling part** of Bayesian analysis from the **numerical approximation part**.

Recall from Chapter 2 that the necessary ingredients of Bayesian data analysis are:

1. **Model specification:**
   a collection of probability distributions $\{p(y \mid \phi), \phi \in \Phi\}$ representing the sampling distribution of the data for some parameter $\phi \in \Phi$.
2. **Prior specification:**
   a probability distribution $p(\phi)$ representing prior information about which parameter values are likely to describe the sampling distribution.

Once the model and prior are specified and the data are observed, the posterior distribution

$$p(\phi \mid y)$$

is completely determined. It is given by the Bayes' rule:

$$p(\phi \mid y) = \frac{p(\phi)p(y \mid \phi)}{p(y)} = \frac{p(\phi)p(y \mid \phi)}{\int p(\phi)p(y \mid \phi)\, d\phi}.$$

At this stage, there is no further modelling or estimation involved. All that remains is to **describe or summarize the posterior distribution**.

3. **Posterior summary:**
   describing the posterior distribution $p(\phi \mid y)$ using quantities of interest such as posterior means, medians, modes, predictive probabilities, and credible regions.

For many models, the $p(\phi \mid y)$ is complicated and difficult to compute analytically. In such cases, we use **MC methods** to study the posterior distribution by generating samples from it.

Thus, we have the following

> **i** Key point of MC and MCMC
>
> MC and MCMC algorithms
>
> - are **not models** (i.e., the likelihood), nor the prior,
> - do **not create additional information** beyond what is in y and $p(\phi)$,
> - are simply **numerical tools for studying the posterior distribution**.

For example, suppose we obtain Monte Carlo samples

$$\phi^{(1)}, \ldots, \phi^{(S)}$$

that are approximately drawn from $p(\phi \mid y)$. These samples can be used to approximate posterior quantities.

For instance,

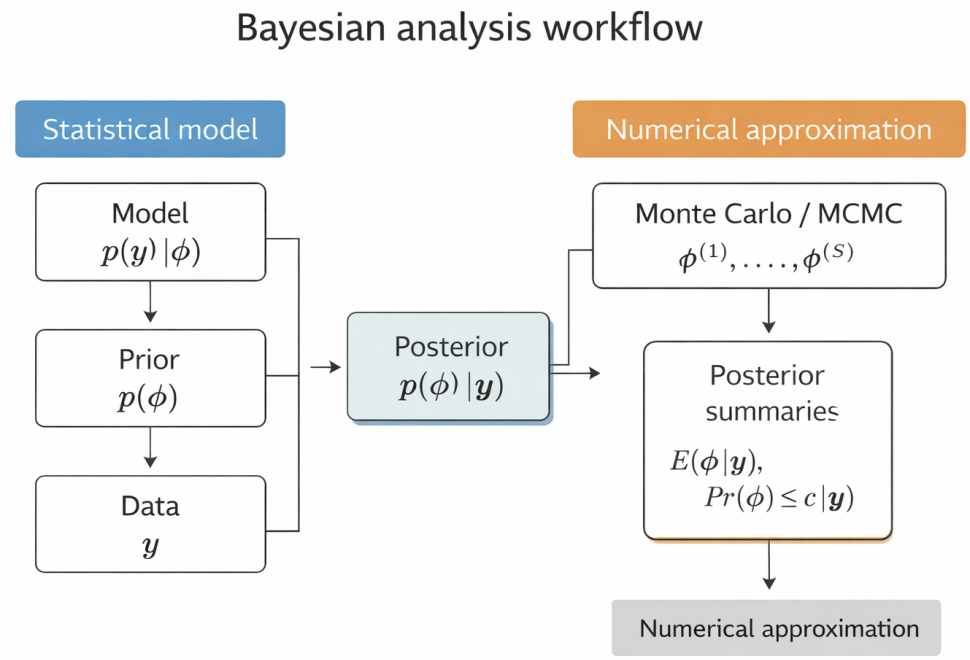
$$\frac{1}{S}\sum_{s=1}^{S} \phi^{(s)} \;\approx\; \int \phi\, p(\phi \mid y)\, d\phi,$$

and

$$\frac{1}{S}\sum_{s=1}^{S} \mathbf{1}(\phi^{(s)} \leq c) \;\approx\; \Pr(\phi \leq c \mid y) = \int_{-\infty}^{c} p(\phi \mid y)\, d\phi.$$

To keep this distinction clear, it is helpful to reserve the word **estimation** for the statistical inference based on the posterior distribution $p(\phi \mid y)$, and the word **approximation** for the numerical procedures used such as MC, to evaluate integrals involving that posterior distribution.

Bayesian analysis workflow



Bayesian data analysis involves two conceptually different components:

1. **Statistical modeling**
2. **Numerical approximation**

It is important not to confuse these two parts.
The first part of Bayesian analysis is **statistical modeling**: we specify the likelihood $p(y \mid \phi)$ and prior $p(\phi)$, which together determine the posterior distribution $p(\phi \mid y)$.
The second part is **numerical approximation**. When the posterior cannot be computed analytically, Monte Carlo or MCMC methods generate samples from $p(\phi \mid y)$, which are then used to compute posterior summaries.

This Chapter borrows materials from Chapter 6 in Hoff (2009).

6 Introduction to MCMC diagnostics

In the previous chapter, we learned how Monte Carlo (MC) and Markov chain Monte Carlo (MCMC) methods can be used to approximate posterior quantities of interest. We now turn to an important practical question:

> How do we know whether an MCMC approximation is reliable?

This question motivates the study of **MCMC diagnostics**.

6.1 Why do we need diagnostics?

The purpose of MC or MCMC approximation is to generate a sequence of parameter values

$$\{\phi^{(1)}, \dots, \phi^{(S)}\}$$

such that, for a function g of interest,

$$\frac{1}{S}\sum_{s=1}^{S} g(\phi^{(s)}) \;\approx\; \int g(\phi)\, p(\phi)\, d\phi = E[g(\phi)].$$

In other words, the **empirical average** of

$$\{g(\phi^{(1)}), \dots, g(\phi^{(S)})\}$$

approximates the **expected value** of $g(\phi)$ under the target distribution $p(\phi)$. In Bayesian inference, the target distribution is usually the **posterior distribution** $p(\phi \mid y)$.

> **Key idea**
>
> For this approximation to work well for many different functions g, the **empirical distribution of the simulated values**
>
> $$\{\phi^{(1)}, \dots, \phi^{(S)}\}$$
>
> should resemble the **target distribution** $p(\phi)$.

For ordinary MC, this is relatively easy to understand, because the simulated values are independent. For MCMC, however, the values are dependent, so we must check more carefully whether the approximation is trustworthy.

6.2 MC versus MCMC

6.2.1 Monte Carlo simulation

In ordinary Monte Carlo, we generate **independent samples**

$$\phi^{(1)}, \dots, \phi^{(S)} \sim p(\phi).$$

This is often viewed as the **gold standard**, because each draw already has the correct target distribution.

For any set A,

$$\Pr(\phi^{(s)} \in A) = \int_A p(\phi)\, d\phi$$

for each $s = 1, \dots, S$, and the samples are independent across s.

As a result, MC samples generally explore the parameter space efficiently.

6.2.2 Markov chain Monte Carlo

In MCMC, the samples

$$\phi^{(1)}, \dots, \phi^{(S)}$$

are **not independent**. Instead, they form a **Markov chain**, so each value depends on the previous ones.

The hope is that the chain is constructed in such a way that

$$\Pr(\phi^{(s)} \in A) \to \int_A p(\phi)\, d\phi \qquad \text{as } s \to \infty.$$

Thus, after enough iterations, the distribution of the simulated values becomes close to the target distribution.

> **Reminder**
>
> - **MC:** independent samples from $p(\phi)$
> - **MCMC:** dependent samples whose distribution converges to $p(\phi)$
> - **Goal:** approximate expectations of the form
>
> $$E[g(\phi)] = \int g(\phi)\, p(\phi)\, d\phi$$

6.3 Two main concerns in MCMC

When we use an MCMC algorithm, there are two major practical questions:

1. **Has the chain reached its target distribution?**
   This is the issue of **convergence** or **stationarity**.
2. **How strongly are the samples correlated?**
   This is the issue of **mixing** and **autocorrelation**.

A chain can be approximately stationary but still highly autocorrelated. In that case, it may still provide a poor approximation unless we run it for a long time.

6.4 A motivating example: a three-component mixture distribution

To see why these issues matter, consider a simple target distribution involving two variables:

- a **discrete variable**

$$\delta \in \{1, 2, 3\},$$

- and a **continuous variable**

$$\theta \in \mathbb{R}.$$

The target joint distribution is defined as follows.

The prior probabilities for δ are

$$\Pr(\delta = 1), \Pr(\delta = 2), \Pr(\delta = 3) = (0.45,\ 0.10,\ 0.45).$$

Conditional on δ, the variable θ follows a normal distribution:

$$\theta \mid \delta \sim \text{Normal}(\mu_\delta, \sigma_\delta).$$

The parameters are

$$(\mu_1, \mu_2, \mu_3) = (-3, 0, 3)$$

and

$$(\sigma_1^2, \sigma_2^2, \sigma_3^2) = \left(\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3}\right).$$

6.4.1 Interpretation

This is a **mixture of three normal distributions**.

- δ indicates **group membership**
- $(\mu_\delta, \sigma_\delta^2)$ are the **mean and variance** for group δ

Therefore, the marginal distribution of θ is

$$p(\theta) = \sum_{\delta=1}^{3} p(\theta \mid \delta)\, p(\delta).$$

This distribution has **three modes**, located near

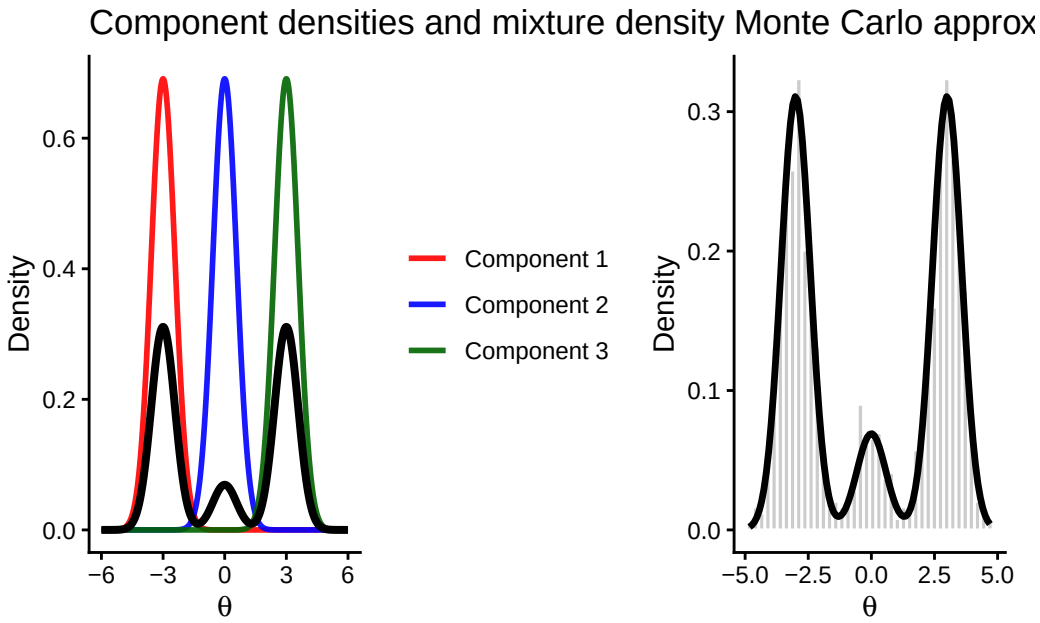
$$-3,\ 0,\ 3.$$

These three modes make the example especially useful for illustrating the difference between MC and MCMC.

```r
library(ggplot2)
library(patchwork)

set.seed(8310)

# ------------------------------------------------
# Parameters of the three-component mixture
# ------------------------------------------------
mu <- c(-3, 0, 3)
sigma <- sqrt(1 / 3)
p <- c(0.45, 0.10, 0.45)

# ------------------------------------------------
# Grid for exact density plots
# ------------------------------------------------
theta_grid <- seq(-6, 6, length.out = 1000)

# Unweighted component densities
f1 <- dnorm(theta_grid, mean = mu[1], sd = sigma)
f2 <- dnorm(theta_grid, mean = mu[2], sd = sigma)
f3 <- dnorm(theta_grid, mean = mu[3], sd = sigma)

# Mixture density
mix <- p[1] * f1 + p[2] * f2 + p[3] * f3

# Data frame for the exact densities
dens_df <- data.frame(
  theta = rep(theta_grid, 4),
  density = c(f1, f2, f3, mix),
  curve = factor(
    rep(c("Component 1", "Component 2", "Component 3", "Mixture"),
        each = length(theta_grid)),
    levels = c("Component 1", "Component 2", "Component 3", "Mixture")
  )
)

# ------------------------------------------------
# Monte Carlo samples from the mixture
# ------------------------------------------------
n_mc <- 1000
delta_mc <- sample(1:3, size = n_mc, replace = TRUE, prob = p)
theta_mc <- rnorm(n_mc, mean = mu[delta_mc], sd = sigma)
```

```r
mc_df <- data.frame(theta = theta_mc)

# ------------------------------------------------
# Plot 1: component densities + mixture density
# ------------------------------------------------
p1 <- ggplot() +
  geom_line(
    data = subset(dens_df, curve != "Mixture"),
    aes(x = theta, y = density, color = curve),
    linewidth = 1,
    alpha = 0.9
  ) +
  geom_line(
    data = subset(dens_df, curve == "Mixture"),
    aes(x = theta, y = density),
    linewidth = 1.4,
    color = "black"
  ) +
  scale_color_manual(values = c(
    "Component 1" = "red",
    "Component 2" = "blue",
    "Component 3" = "darkgreen"
  )) +
  labs(
    title = "Component densities and mixture density",
    x = expression(theta),
    y = "Density",
    color = NULL
  ) +
  theme_classic()

# ------------------------------------------------
# Plot 2: Monte Carlo approximation
# ------------------------------------------------
p2 <- ggplot(mc_df, aes(x = theta)) +
  geom_histogram(
    aes(y = after_stat(density)),
    bins = 40,
    fill = "grey80",
    color = "white"
  ) +
  geom_function(
```

```
    fun = function(x) {
      p[1] * dnorm(x, mean = mu[1], sd = sigma) +
      p[2] * dnorm(x, mean = mu[2], sd = sigma) +
      p[3] * dnorm(x, mean = mu[3], sd = sigma)
    },
    linewidth = 1.2,
    color = "black"
  ) +
  labs(
    title = "Monte Carlo approximation",
    x = expression(theta),
    y = "Density"
  ) +
  theme_classic()

# ----------------------------------------------------
# Combine the two plots
# ----------------------------------------------------
p1 + p2
```



6.5 MC sampling from the mixture distribution

It is straightforward to generate **independent MC samples** from the joint distribution of

$$\phi = (\delta, \theta).$$

The procedure is:

1. Sample

$$\delta \sim p(\delta)$$

2. Given δ, sample

$$\theta \sim p(\theta \mid \delta).$$

The pair

$$(\delta, \theta)$$

is therefore a draw from the joint distribution

$$p(\delta, \theta) = p(\delta)p(\theta \mid \delta).$$

Repeating this process produces independent samples. Because the samples are independent, the empirical distribution of the simulated θ values tends to approximate the target marginal density quite well, even with a moderate sample size.

6.6 A Gibbs sampler for the mixture distribution

Now suppose we instead construct a Gibbs sampler for

$$\phi = (\delta, \theta).$$

The full conditional distribution of θ is already known:

$$\theta \mid \delta \sim \text{Normal}(\mu_\delta, \sigma_\delta).$$

Using Bayes' rule, the full conditional distribution of δ given θ is

$$\Pr(\delta = d \mid \theta) = \frac{\Pr(\delta = d)\,\text{dnorm}(\theta, \mu_d, \sigma_d)}{\sum_{j=1}^{3} \Pr(\delta = j)\,\text{dnorm}(\theta, \mu_j, \sigma_j)}, \qquad d \in \{1, 2, 3\}.$$

The Gibbs sampler alternates between these two conditional distributions:

1. sample $\theta \mid \delta$
2. sample $\delta \mid \theta$

6.7 Why the Gibbs sampler can mix poorly

Although this Gibbs sampler is theoretically correct, it may perform poorly in practice.

The problem is that the chain can become **stuck near one mode** for many iterations. For example:

- if θ is near 0, then δ is likely to be sampled as 2;
- if $\delta = 2$, then the next value of θ is likely to remain near 0.

So once the chain enters the middle component, it may remain there for a long time. Similarly, the chain may also get stuck near -3 or 3.

This is an example of **high autocorrelation**: consecutive draws are strongly related to each other. As a result, even a long MCMC sequence may not explore the three modes in the correct proportions.

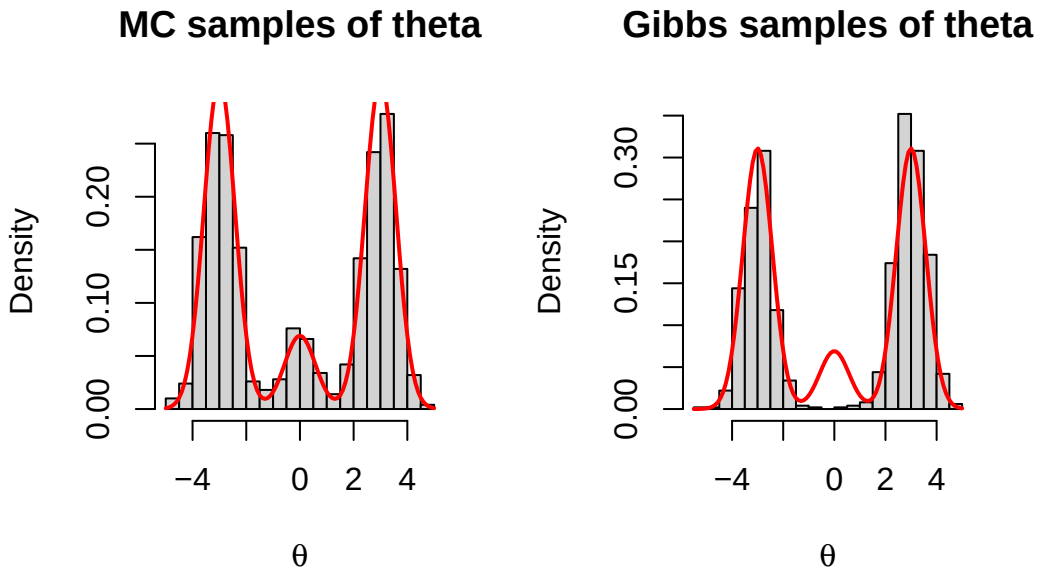
> ⚠ Important intuition
>
> A long MCMC chain is not automatically a good MCMC chain.
> If the chain mixes slowly, then many consecutive draws carry nearly the same information.

6.8 Example code: MC and Gibbs sampling for the mixture model

```
set.seed(8310)

# Parameters
mu <- c(-3, 0, 3)
sigma <- sqrt(1/3)
```

```r
p <- c(0.45, 0.10, 0.45)

# Function to sample from the mixture distribution
sample_mixture <- function(n) {
  delta <- sample(1:3, size = n, replace = TRUE, prob = p)
  theta <- rnorm(n, mean = mu[delta], sd = sigma)
  data.frame(delta = delta, theta = theta)
}

# Gibbs sampler
gibbs_sampler <- function(n) {
  delta <- numeric(n)
  theta <- numeric(n)

  # Initialize
  delta[1] <- sample(1:3, size = 1, replace = TRUE, prob = p)
  theta[1] <- rnorm(1, mean = mu[delta[1]], sd = sigma)

  for (i in 2:n) {
    prob <- p * dnorm(theta[i-1], mean = mu, sd = sigma)
    prob <- prob / sum(prob)
    delta[i] <- sample(1:3, size = 1, replace = TRUE, prob = prob)

    theta[i] <- rnorm(1, mean = mu[delta[i]], sd = sigma)
  }

  data.frame(delta = delta, theta = theta)
}

# Generate samples
mc_samples <- sample_mixture(1000)
mcmc_samples <- gibbs_sampler(1000)

# Plotting
par(mfrow = c(1, 2))

hist(mc_samples$theta, breaks = 30, probability = TRUE,
     main = "MC samples of theta", xlab = expression(theta))
curve(p[1] * dnorm(x, mean = mu[1], sd = sigma) +
      p[2] * dnorm(x, mean = mu[2], sd = sigma) +
      p[3] * dnorm(x, mean = mu[3], sd = sigma),
      add = TRUE, col = "red", lwd = 2)
```

```r
hist(mcmc_samples$theta, breaks = 30, probability = TRUE,
     main = "Gibbs samples of theta", xlab = expression(theta))
curve(p[1] * dnorm(x, mean = mu[1], sd = sigma) +
      p[2] * dnorm(x, mean = mu[2], sd = sigma) +
      p[3] * dnorm(x, mean = mu[3], sd = sigma),
      add = TRUE, col = "red", lwd = 2)
```



Why MCMC diagnostics matter

The main issue is that MCMC samples are *dependent*.

Suppose we use the sample mean

$$\bar{\phi}_S = \frac{1}{S}\sum_{s=1}^{S} \phi^{(s)}$$

to approximate

$$E[\phi].$$

If the samples were independent, then

$$\mathrm{Var}_{\mathrm{MC}}(\bar{\phi}_S) = \frac{\mathrm{Var}(\phi)}{S}.$$

For MCMC samples, however, the variance is larger because of correlation within the chain. Hoff (2009) shows that

$$\mathrm{Var}_{\mathrm{MCMC}}(\bar{\phi}_S) = \mathrm{Var}_{\mathrm{MC}}(\bar{\phi}_S) + \frac{1}{S^2}\sum_{s\neq t} E\left[\left(\phi^{(s)} - \phi_0\right)\left(\phi^{(t)} - \phi_0\right)\right],$$

where

$$\phi_0 = E[\phi].$$

Proof

Let $\phi_0 = E[\phi]$. Then

$$\begin{aligned}
\mathrm{Var}_{\mathrm{MCMC}}[\bar{\phi}] &= \mathrm{E}\left[\left(\bar{\phi} - \phi_0\right)^2\right] \\
&= \mathrm{E}\left[\left\{\frac{1}{S}\sum_{s=1}^{S}\left(\phi^{(s)} - \phi_0\right)\right\}^2\right] \\
&= \frac{1}{S^2}\mathrm{E}\left[\sum_{s=1}^{S}\left(\phi^{(s)} - \phi_0\right)^2 + \sum_{s\neq t}\left(\phi^{(s)} - \phi_0\right)\left(\phi^{(t)} - \phi_0\right)\right] \\
&= \frac{1}{S^2}\sum_{s=1}^{S}\mathrm{E}\left[\left(\phi^{(s)} - \phi_0\right)^2\right] + \frac{1}{S^2}\sum_{s\neq t}\mathrm{E}\left[\left(\phi^{(s)} - \phi_0\right)\left(\phi^{(t)} - \phi_0\right)\right] \\
&= \mathrm{Var}_{\mathrm{MC}}[\bar{\phi}] + \frac{1}{S^2}\sum_{s\neq t}\mathrm{E}\left[\left(\phi^{(s)} - \phi_0\right)\left(\phi^{(t)} - \phi_0\right)\right].
\end{aligned}$$

This expression makes the key point very clear:

> the variance of the MCMC approximation is the ordinary MC variance plus an extra term caused by dependence among the samples.

So, even if the chain has the correct stationary distribution, strong autocorrelation can make the approximation much less precise.

Autocorrelation

A standard way to measure dependence is through the autocorrelation function.

For a sequence

$$\phi_1, \dots, \phi_S,$$

the lag-t autocorrelation measures the relationship between values that are t steps apart. A sample version is

$$\text{acf}_t(\phi) = \frac{\frac{1}{S-t}\sum_{s=1}^{S-t}(\phi_s - \bar{\phi})(\phi_{s+t} - \bar{\phi})}{\frac{1}{S-1}\sum_{s=1}^{S}(\phi_s - \bar{\phi})^2}.$$

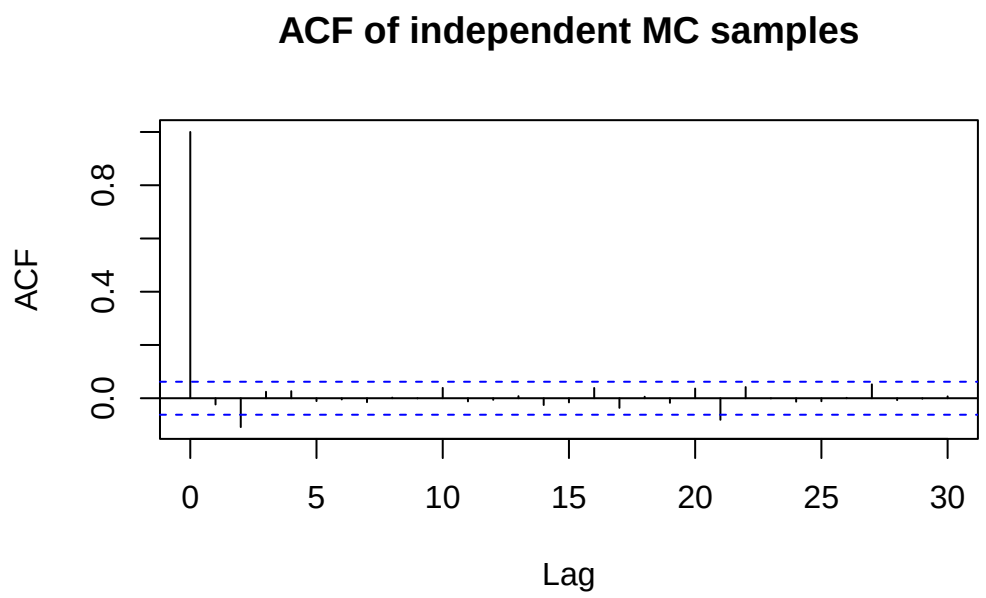
In R, this is computed by `acf()`.

How to interpret autocorrelation

- If autocorrelation drops quickly to zero, the chain mixes well.
- If autocorrelation remains large for many lags, the chain mixes slowly.
- High autocorrelation means we need many more samples to achieve a good approximation.

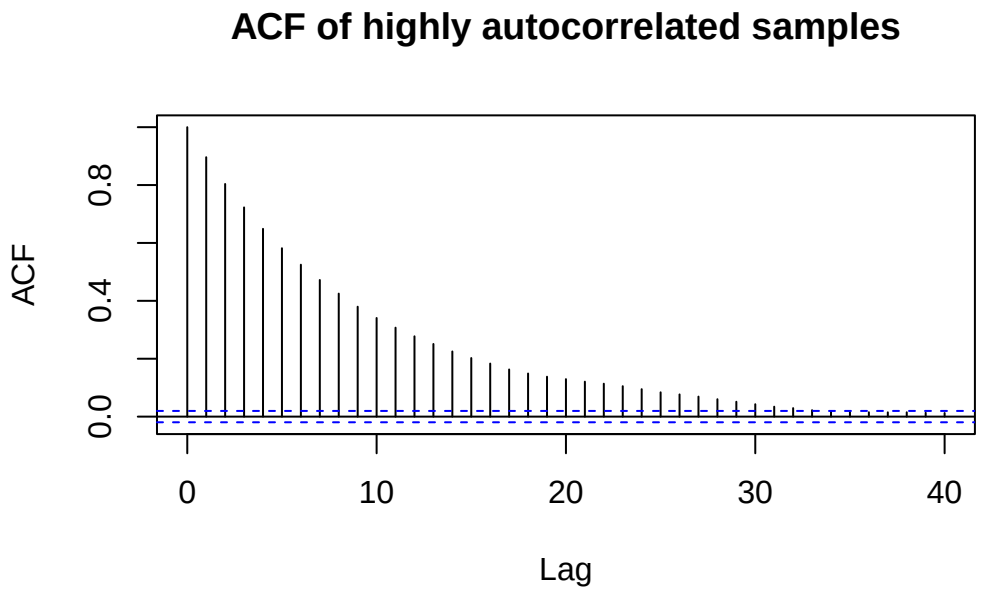
If we have an independent MC sample, the autocorrelation is zero at all lags. In this case, (auto)correlation is not a problem. As a result, the `acf()` function will show a spike at lag 0 and essentially zero autocorrelation at all other lags.

```
set.seed(8310)
x <- rnorm(1000)
acf(x, main = "ACF of independent MC samples")
```



If there is an extremely high autocorrelation, as in the Gibbs sampler for the mixture distribution, then the `acf()` function will show a slow decay of autocorrelation across many lags.

```
set.seed(8310)
n <- 10000
x <- numeric(n)
x[1] <- rnorm(1)
for (i in 2:n) {
  x[i] <- 0.9 * x[i-1] + sqrt(1
- 0.9^2) * rnorm(1)
}
acf(x, main = "ACF of highly autocorrelated samples")
```



In the mixture example, the autocorrelation is extremely high: for the chain of 10,000 θ-values, the lag-10 autocorrelation is about 0.93 and the lag-50 autocorrelation is still about 0.812. This indicates extremely slow movement through the parameter space.

Effective sample size

A useful summary of autocorrelation is the effective sample size.

The idea is simple:

> If the MCMC samples are highly dependent, how many independent Monte Carlo samples would give the same precision?

If S_{eff} denotes the effective sample size, it is defined through

$$\text{Var}_{\text{MCMC}}(\bar{\phi}) = \frac{\text{Var}(\phi)}{S_{\text{eff}}}.$$

Thus, S_{eff} is the number of independent samples that would be equivalent, in precision, to the dependent MCMC chain.

In the coda package, the effective sample size is computed by

```
effectiveSize(x)
```

In the mixture example, the effective sample size of 10,000 Gibbs samples of θ is only about 18.42. So even though the chain contains 10,000 draws, its precision is only about as good as 18 independent MC draws.

> ⚠ "Interpretation"
>
> A chain of length 10,000 does not necessarily contain 10,000 independent pieces of information.
> What matters is not only the number of iterations, but also how strongly the chain is correlated.

Diagnostics for the semiconjugate normal model

We now return to the semiconjugate normal model from the previous chapter

Recall that the Gibbs sampler generated samples of

$$\theta \qquad \text{and} \qquad \tilde{\sigma}^2 = \frac{1}{\sigma^2}.$$

For this example, the Gibbs sampler behaves much better than in the multimodal mixture example. Hoff (2009) reports that the traceplots suggest rapid convergence and relatively low autocorrelation. In particular, the lag-1 autocorrelation is about 0.031 for θ and about 0.147 for σ^2, with effective sample sizes about 1000 and 742, respectively.

This illustrates an important lesson:

> Not all MCMC problems are equally difficult. A Gibbs sampler can work extremely well in some models and very poorly in others.

The three most basic diagnostics

For beginning MCMC analysis, three diagnostics are especially important:

1. Traceplots

A traceplot displays the sampled values in the order they were generated.

What we hope to see:

- the chain fluctuates around a stable region;
- there is no obvious drift or trend;
- the chain moves around rather than sticking for too long in one place.

What would be concerning:

- a strong upward or downward trend;
- a sudden shift from one region to another after a long time;
- long flat stretches indicating the chain is barely moving.

2. Autocorrelation plots

An autocorrelation plot shows how strongly the samples depend on previous iterations.

What we hope to see:

- autocorrelation drops quickly toward zero.

What would be concerning:

- autocorrelation remains large for many lags.

3. Effective sample size

Effective sample size summarizes the practical amount of information in the chain.

What we hope to see:

- an effective sample size that is not too small relative to the total number of samples.

What would be concerning:

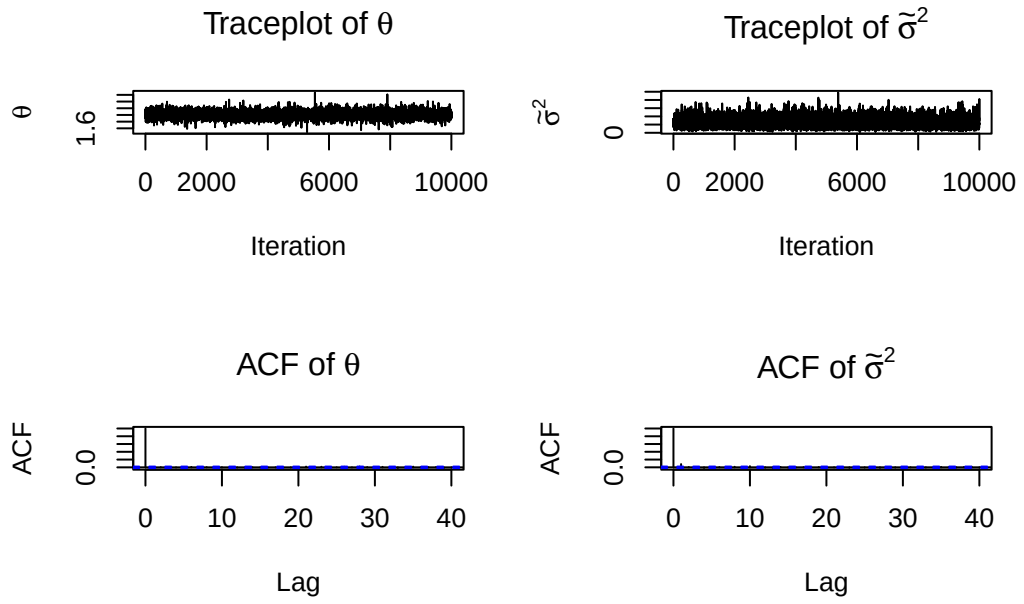
- a very small effective sample size, indicating that the chain is highly inefficient.

The following code reruns the Gibbs sampler so that the matrix PHI is available for diagnostic plots and effective sample size calculations.

```r
library(coda)
set.seed(8310)

# Data and prior hyperparameters
mu0 <- 1.9
t20 <- 0.95^2
s20 <- 0.01
nu0 <- 1
y <- c(1.64, 1.70, 1.72, 1.74, 1.82, 1.82, 1.82, 1.90, 2.08)

mean.y <- mean(y)
var.y <- var(y)
n <- length(y)

# Gibbs sampler

S <- 10000
PHI <- matrix(NA, nrow = S, ncol = 2)
phi <- c(mean.y, 1 / var.y)
PHI[1, ] <- phi

for (s in 2:S) {
  mun <- (mu0/t20 + n*mean.y*phi[2]) / (1/t20 + n*phi[2])
  t2n <- 1 / (1/t20 + n*phi[2])
  phi[1] <- rnorm(1, mean = mun, sd = sqrt(t2n))

  nun <- nu0 + n
  s2n <- (nu0*s20 + (n-1)*var.y + n*(mean.y - phi[1])^2) / nun
  phi[2] <- rgamma(1, shape = nun/2, rate = nun*s2n/2)

  PHI[s, ] <- phi
}

colnames(PHI) <- c("theta", "sigma2_inv")
```

```r
par(mfrow = c(2, 2))

plot(PHI[, "theta"], type = "l",
     main = expression("Traceplot of " * theta),
     xlab = "Iteration", ylab = expression(theta))

plot(PHI[, "sigma2_inv"], type = "l",
```

```
     main = expression("Traceplot of " * tilde(sigma)^2),
     xlab = "Iteration", ylab = expression(tilde(sigma)^2))

acf(PHI[, "theta"], main = expression("ACF of " * theta))
acf(PHI[, "sigma2_inv"], main = expression("ACF of " * tilde(sigma)^2))
```




```
coda::effectiveSize(PHI[, "theta"])
```

```
 var1
10000
```

```
coda::effectiveSize(PHI[, "sigma2_inv"])
```

```
    var1
8465.349
```

Practical interpretation

When looking at MCMC output, we are usually asking three basic questions:

1. Has the chain reached the target distribution? Traceplots help assess this.

2. How dependent are the samples? Autocorrelation plots help assess this.
3. How much information do the samples really contain? Effective sample size gives a practical summary.

These diagnostics are important because MCMC output can look large in quantity while still being limited in quality.

Takeaway

MCMC diagnostics are not just technical tools.
They answer three fundamental questions:

1. **Did the chain converge?** → traceplots, Gelman–Rubin
2. **How dependent are the samples?** → autocorrelation
3. **How much information do we really have?** → effective sample size

Summary

The main lesson of this chapter is that MCMC samples are useful only to the extent that they

- adequately represent the target distribution, and
- provide enough effective information for posterior approximation.

In practice, we therefore examine

- traceplots for approximate stationarity,
- autocorrelation functions for dependence, and
- effective sample sizes for precision.

Compared with ordinary MC, MCMC often requires much more care in assessing the quality of the approximation.

6.9 Example: Using `rjags` and `coda` for MCMC diagnostics

In this example, we fit a simple normal model in JAGS and then use the `coda` package to examine basic MCMC diagnostics.

Suppose

$$Y_1, \ldots, Y_n \mid \mu, \tau \sim \text{Normal}(\mu, \tau),$$

where $\tau = 1/\sigma^2$ is the precision parameter.

We place the priors

$$\mu \sim \text{Normal}(0, 0.001), \qquad \tau \sim \text{Gamma}(0.001, 0.001).$$

The goal is to generate posterior samples of μ, τ, and σ, and then assess the quality of the MCMC output.

6.9.1 Step 1: Load packages and simulate data

```
library(rjags)
```

```
Linked to JAGS 4.3.2
```

```
Loaded modules: basemod,bugs
```

```
library(coda)

set.seed(8310)

y <- rnorm(50, mean = 2, sd = 1)

data_list <- list(
  y = y,
  n = length(y)
)
```

6.9.2 Step 2: Write the JAGS model

```
model_string <- "
model {
  for (i in 1:n) {
    y[i] ~ dnorm(mu, tau)
  }
```

```
  mu ~ dnorm(0, 0.001)
  tau ~ dgamma(0.001, 0.001)

  sigma <- 1 / sqrt(tau)
}
"
cat(model_string)
```

```
model {
  for (i in 1:n) {
    y[i] ~ dnorm(mu, tau)
  }

  mu ~ dnorm(0, 0.001)
  tau ~ dgamma(0.001, 0.001)

  sigma <- 1 / sqrt(tau)
}
```

6.9.3 Step 3: Choose initial values

We use three chains with dispersed starting values.

```
inits1 <- list(mu = 0,  tau = 1)
inits2 <- list(mu = 5,  tau = 0.5)
inits3 <- list(mu = -3, tau = 2)
```

6.9.4 Step 4: Build the JAGS model

```
jags_mod <- jags.model(
  textConnection(model_string),
  data = data_list,
  inits = list(inits1, inits2, inits3),
  n.chains = 3,
  n.adapt = 1000
)
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 50
   Unobserved stochastic nodes: 2
   Total graph size: 58

Initializing model
```

6.9.5 Step 5: Burn in

```
update(jags_mod, n.iter = 5000)
```

6.9.6 Step 6: Draw posterior samples

```
mcmc_out <- coda.samples(
  model = jags_mod,
  variable.names = c("mu", "tau", "sigma"),
  n.iter = 10000
)
```

The object mcmc_out is an mcmc.list, which can be analyzed using functions from the `coda` package.

6.9.7 Step 7: Basic summary and diagnostics

```
summary(mcmc_out)
```

```
Iterations = 5001:15000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 10000

1. Empirical mean and standard deviation for each variable,
```

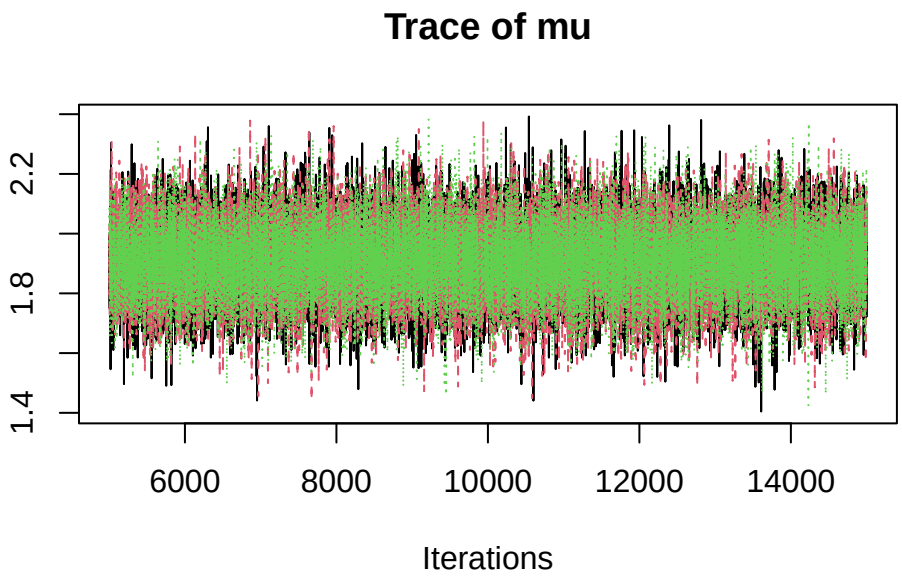
```
   plus standard error of the mean:

         Mean      SD  Naive SE Time-series SE
mu    1.9119 0.12544 0.0007242      0.0007331
sigma 0.8875 0.09193 0.0005308      0.0005405
tau   1.3096 0.26450 0.0015271      0.0015396

2. Quantiles for each variable:

         2.5%    25%   50%    75% 97.5%
mu    1.6655 1.8277 1.912 1.9954 2.160
sigma 0.7305 0.8227 0.879 0.9434 1.089
tau   0.8426 1.1237 1.294 1.4775 1.874
```

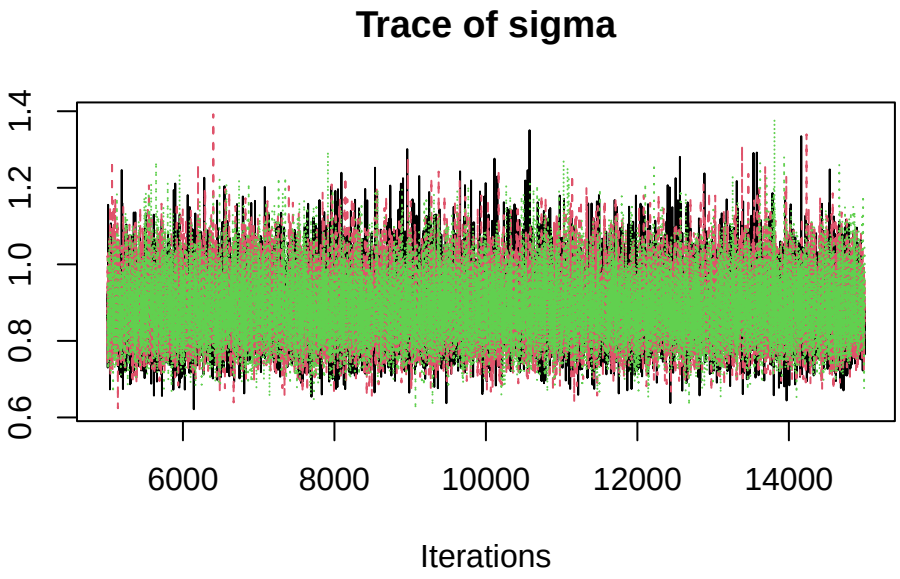
This gives posterior means, standard deviations, and quantiles for each monitored parameter.

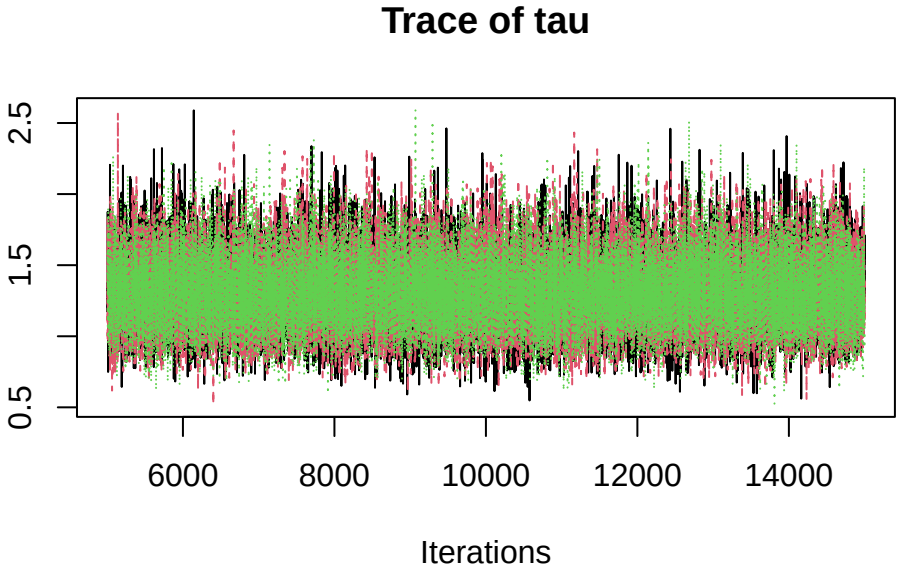
6.9.8 Step 8: Traceplots

A traceplot displays the sampled values in the order they were generated.

```
traceplot(mcmc_out)
```







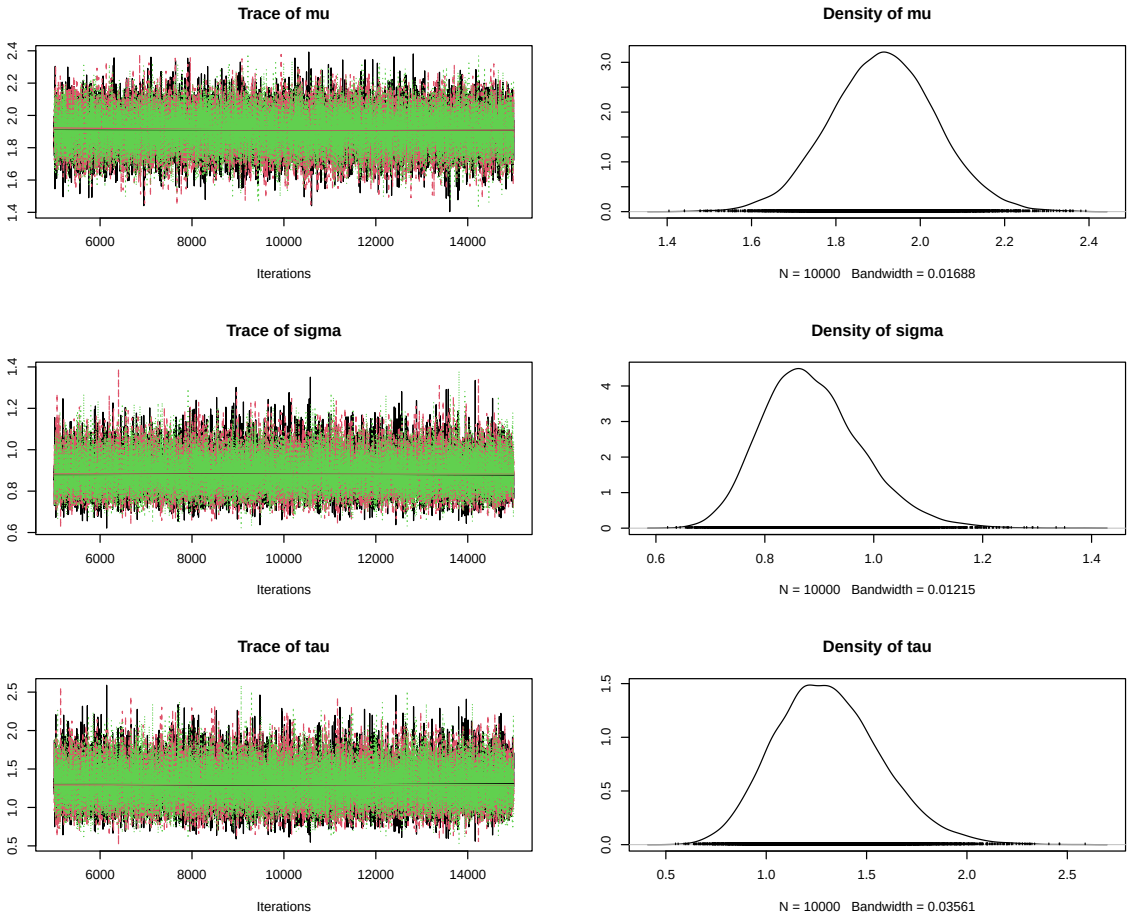
Interpretation:

- A good traceplot should fluctuate around a stable region.
- Different chains should overlap reasonably well.

- Strong trends or long drifts may indicate lack of convergence.

6.9.9 Step 9: Posterior density plots

```
plot(mcmc_out)
```



This produces both traceplots and marginal density plots.

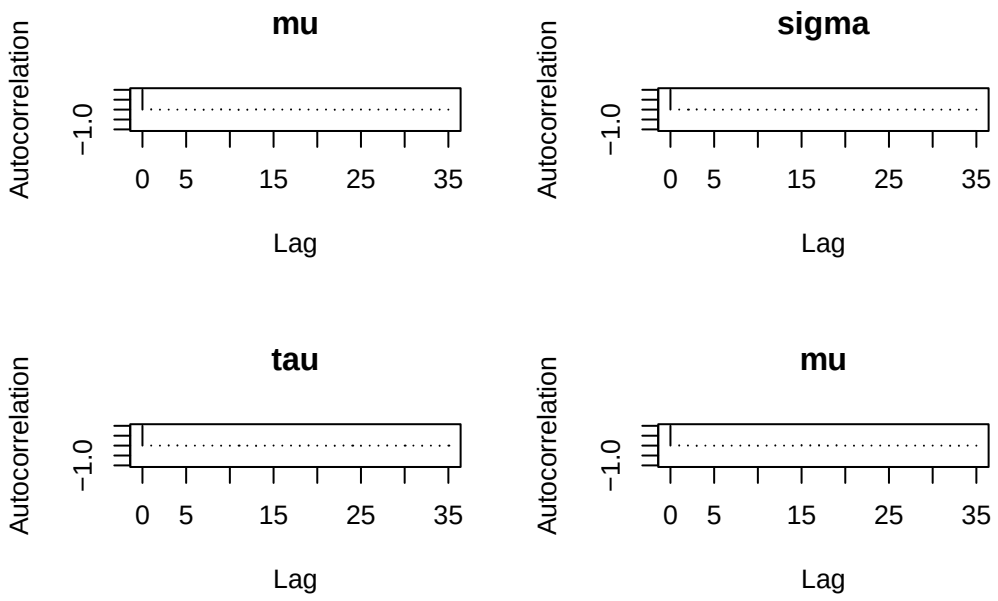
Interpretation:

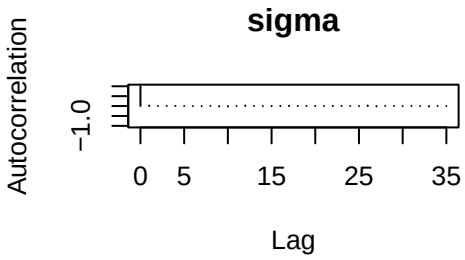
- Density plots summarize the posterior distribution.
- If the chains have converged, the density estimates from different chains should be similar.

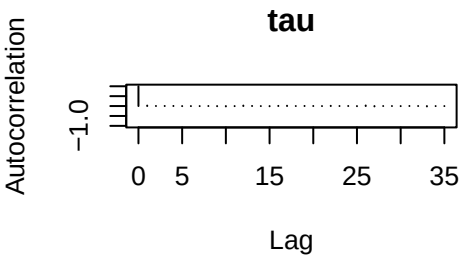
6.9.10 Step 10: Autocorrelation plots

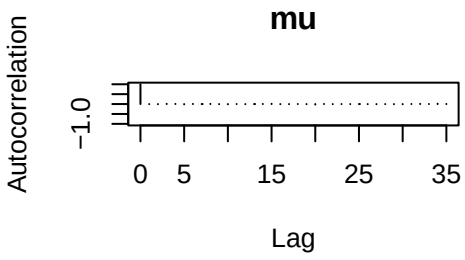
Autocorrelation shows how strongly the samples depend on previous draws.

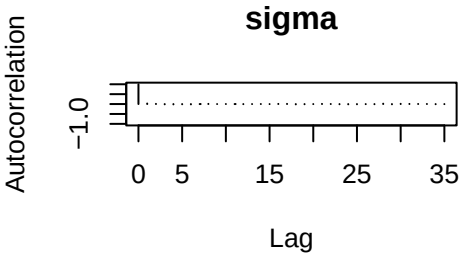
```
autocorr.plot(mcmc_out)
```

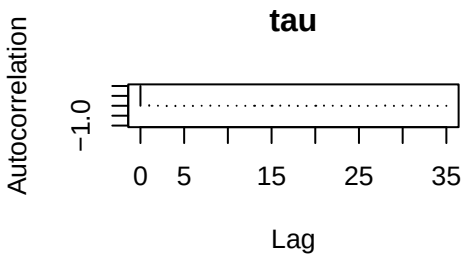












Interpretation:

- Rapid decay in autocorrelation is a good sign.
- Slow decay indicates that the chain is highly dependent and mixes slowly.

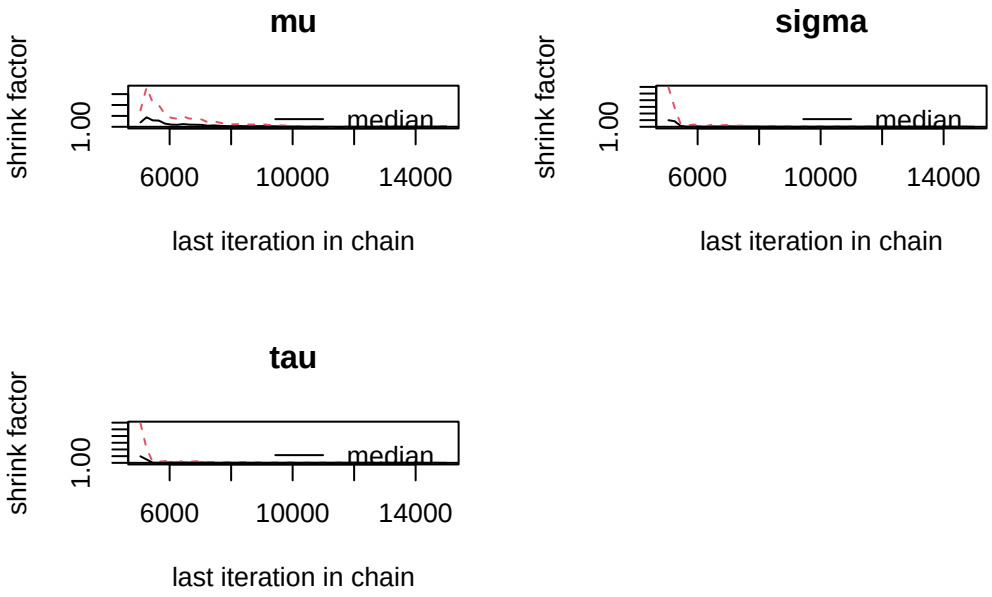
6.9.11 Step 11: Gelman–Rubin diagnostic

A standard convergence diagnostic is the Gelman–Rubin statistic.

```
gelman.diag(mcmc_out)
```

```
Potential scale reduction factors:

      Point est. Upper C.I.
mu             1          1
sigma          1          1
tau            1          1

Multivariate psrf

1
```

```
gelman.plot(mcmc_out)
```



Interpretation:

- Values close to 1 suggest that the chains have mixed well.
- Values substantially larger than 1 suggest that more iterations may be needed.

6.9.12 Step 12: Effective sample size

The effective sample size measures how many independent samples the MCMC output is approximately worth.

```
effectiveSize(mcmc_out)
```

```
      mu    sigma      tau
29295.34 28947.41 29534.65
```

Interpretation:

- Even if we generate many MCMC samples, the effective sample size may be much smaller if autocorrelation is high.
- Larger effective sample sizes indicate more precise MC approximation.

6.9.13 Step 13: Numerical autocorrelations

We can also compute autocorrelations directly.

```
autocorr.diag(mcmc_out)
```

```
                  mu        sigma          tau
Lag 0   1.0000000000  1.000000000  1.000000000
Lag 1   0.0096788947  0.019702064  0.020132439
Lag 5  -0.0057216231 -0.009169443 -0.008649988
Lag 10 -0.0114142466 -0.009246975 -0.009083252
Lag 50  0.0009351514 -0.004982560 -0.005433217
```

This reports estimated autocorrelations at selected lags.

6.9.14 Discussion

This example illustrates a typical workflow for MCMC diagnostics using rjags and coda:

1. run multiple chains from dispersed starting values;
2. discard burn-in;
3. examine traceplots and density plots;
4. check autocorrelation;
5. compute Gelman–Rubin diagnostics;

6. compute effective sample sizes.

In practice, no single diagnostic is sufficient on its own. It is best to use several diagnostics together when assessing MCMC quality.

Putting everything together

```
library(rjags)
library(coda)

set.seed(8310)

# Simulate data
y <- rnorm(50, mean = 2, sd = 1)

data_list <- list(
  y = y,
  n = length(y)
)

# JAGS model
model_string <- "
model {
  for (i in 1:n) {
    y[i] ~ dnorm(mu, tau)
  }

  mu ~ dnorm(0, 0.001)
  tau ~ dgamma(0.001, 0.001)

  sigma <- 1 / sqrt(tau)
}
"

# Initial values
inits1 <- list(mu = 0,  tau = 1)
inits2 <- list(mu = 5,  tau = 0.5)
inits3 <- list(mu = -3, tau = 2)

# Build model
jags_mod <- jags.model(
  file = textConnection(model_string),
  data = data_list,
```

```
  inits = list(inits1, inits2, inits3),
  n.chains = 3,
  n.adapt = 1000
)
```

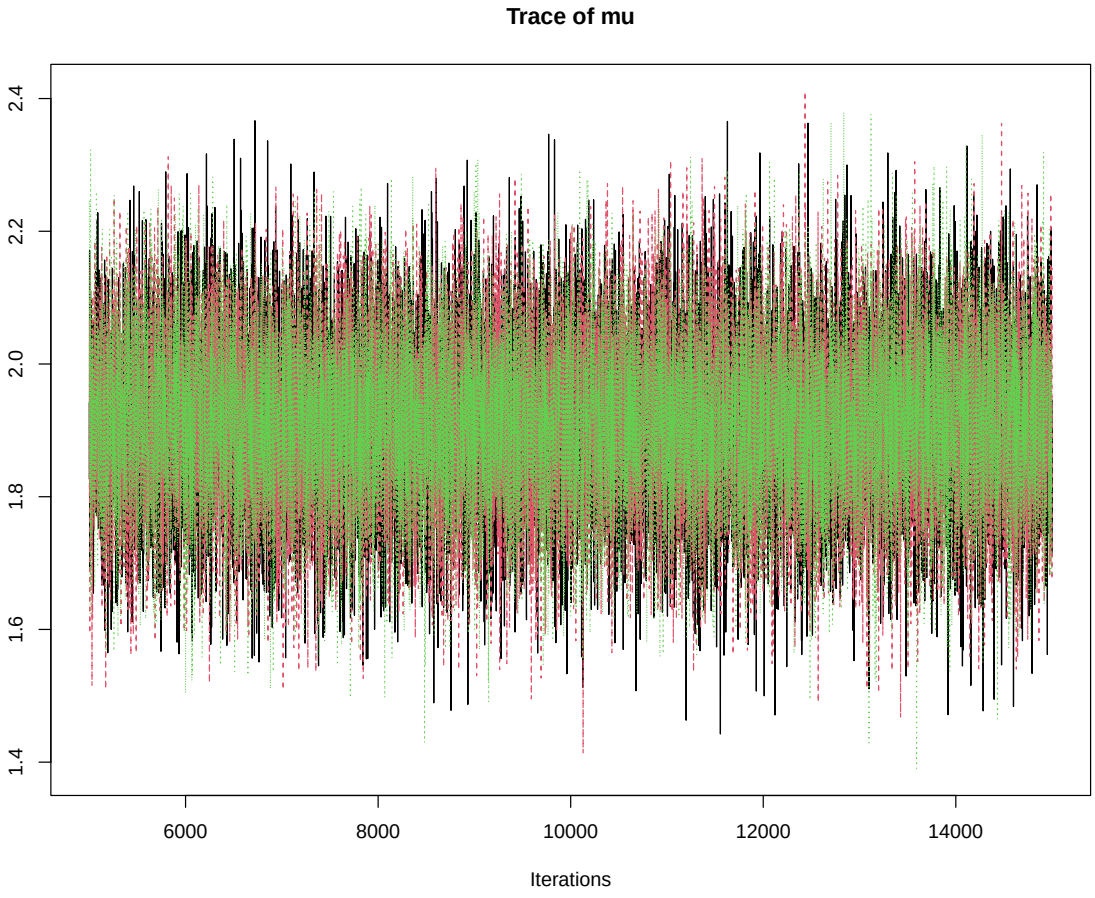
```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 50
   Unobserved stochastic nodes: 2
   Total graph size: 58

Initializing model
```

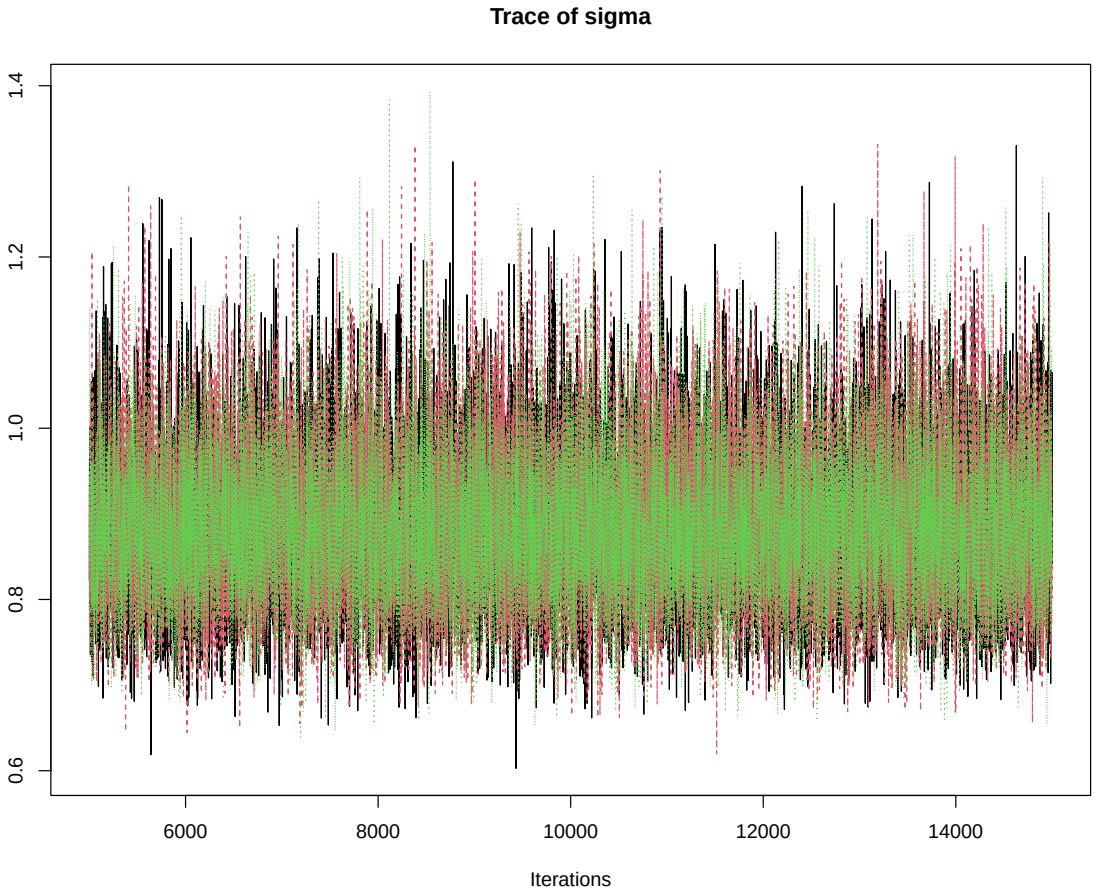
```
# Burn-in
update(jags_mod, n.iter = 5000)

# Posterior samples
mcmc_out <- coda.samples(
  model = jags_mod,
  variable.names = c("mu", "tau", "sigma"),
  n.iter = 10000
)

# Summaries
summary(mcmc_out)
```

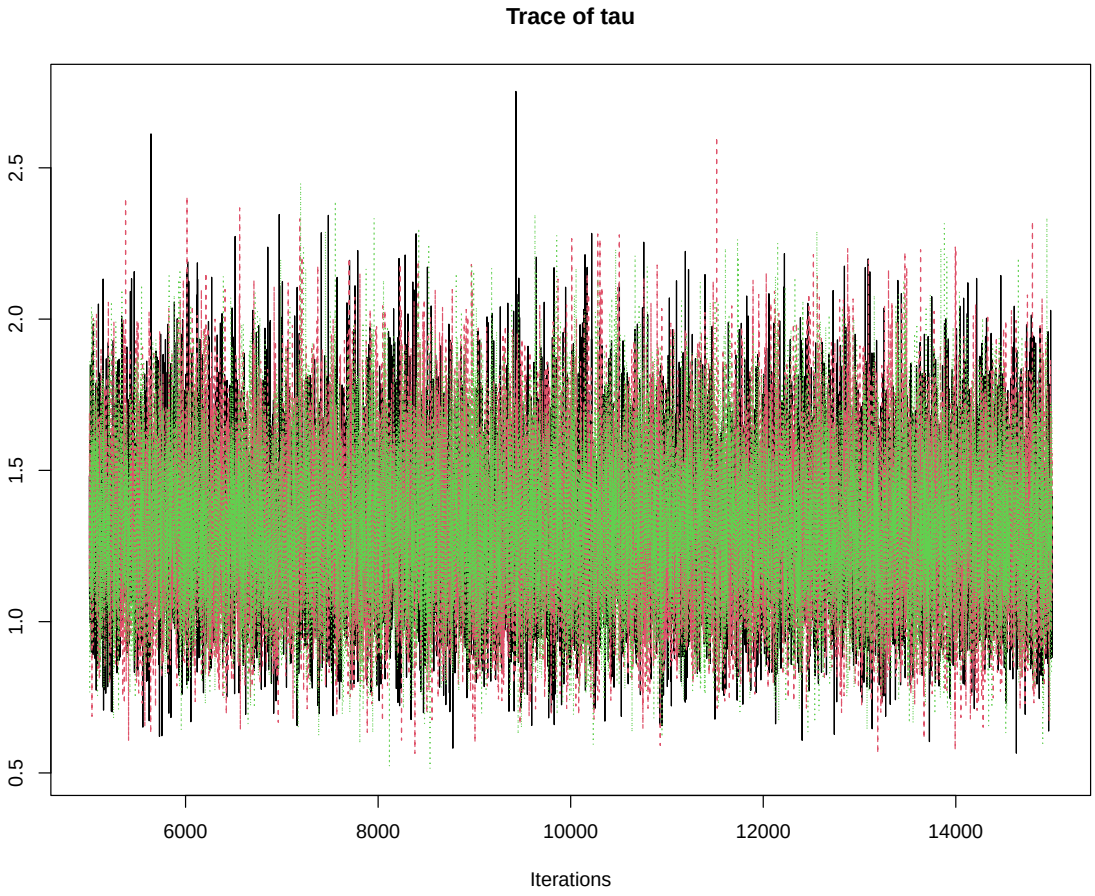
```
Iterations = 5001:15000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 10000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

        Mean      SD  Naive SE Time-series SE
mu    1.9106 0.12579 0.0007262      0.0007308
sigma 0.8869 0.09159 0.0005288      0.0005381
tau   1.3112 0.26450 0.0015271      0.0015857
```

```
2. Quantiles for each variable:

        2.5%    25%   50%    75% 97.5%
mu    1.6627 1.8271 1.910 1.9947 2.158
sigma 0.7292 0.8227 0.879 0.9433 1.088
tau   0.8453 1.1238 1.294 1.4774 1.881
```

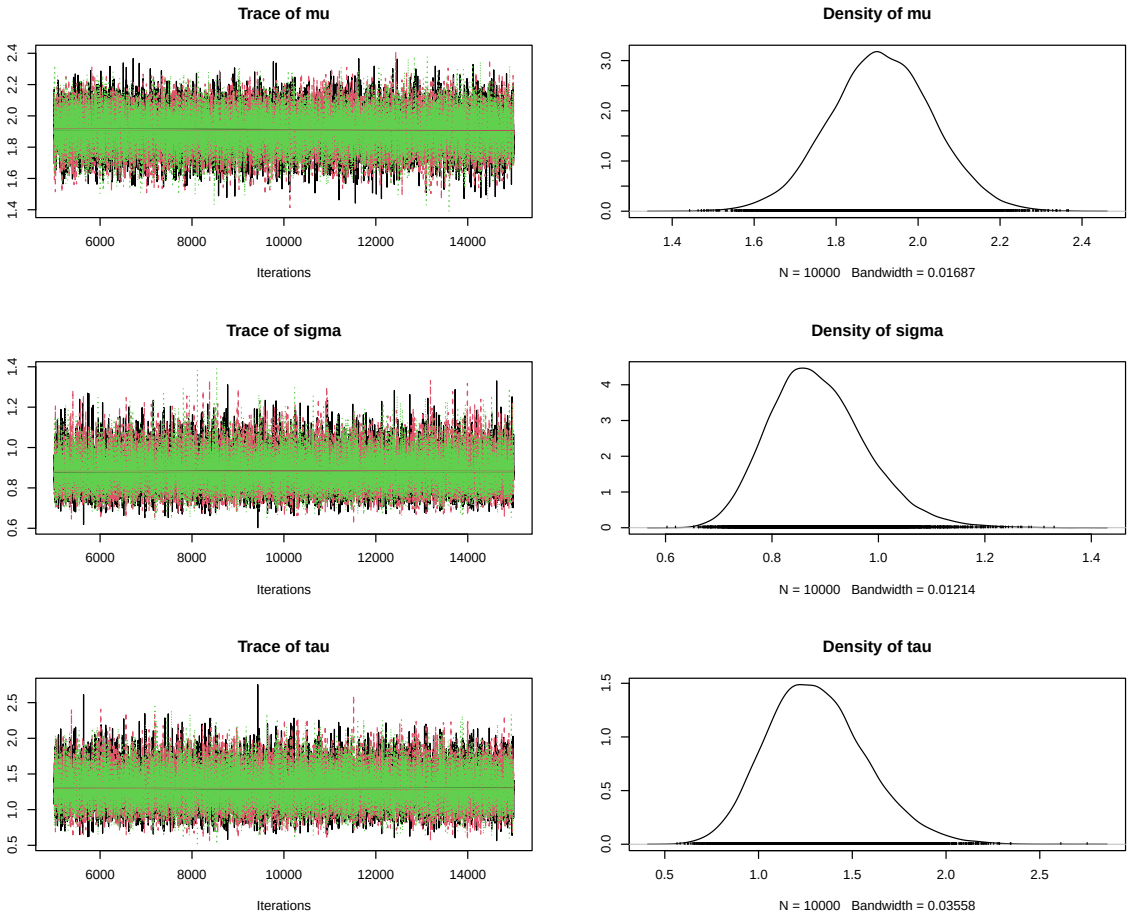
```
# Diagnostics
coda::traceplot(mcmc_out)
```



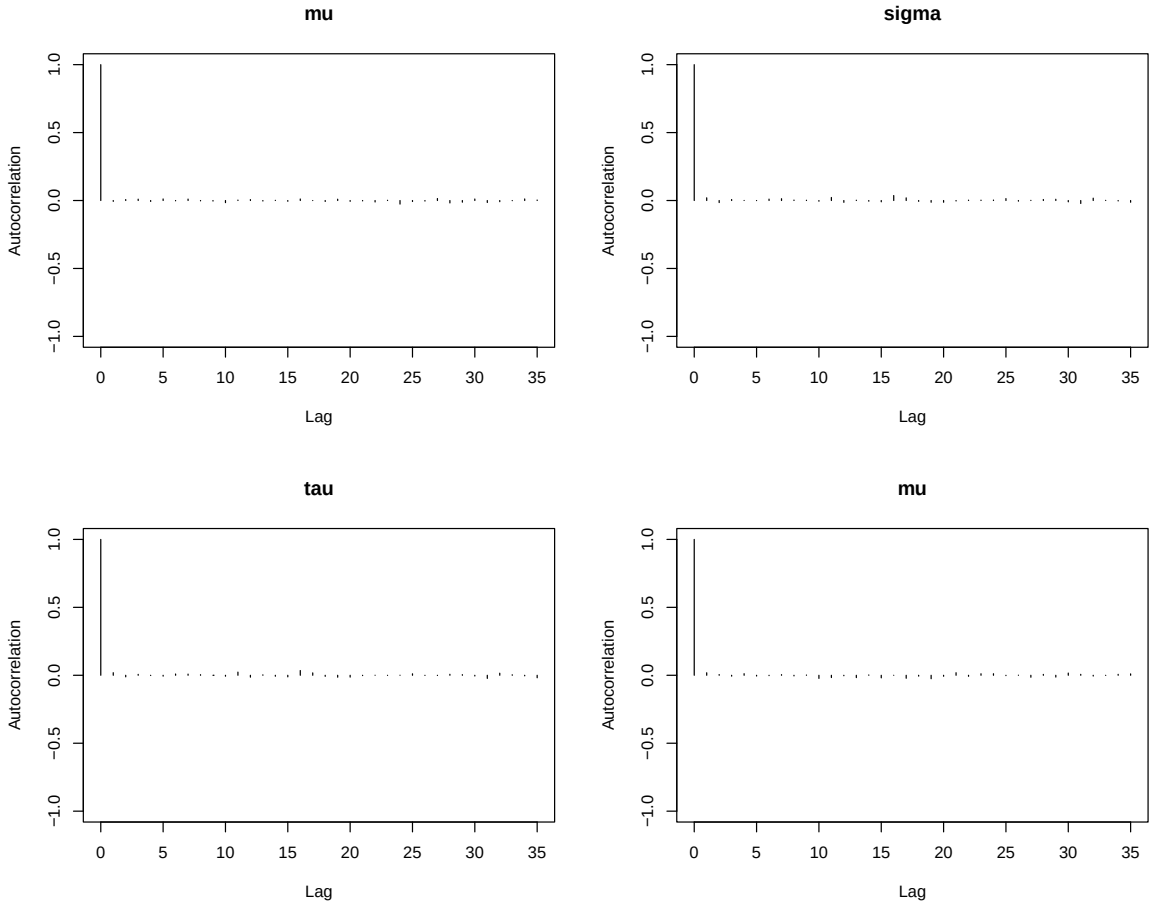

Trace of sigma
1.4
1.2
1.0
0.8
0.6
6000
8000
10000
12000
14000
Iterations

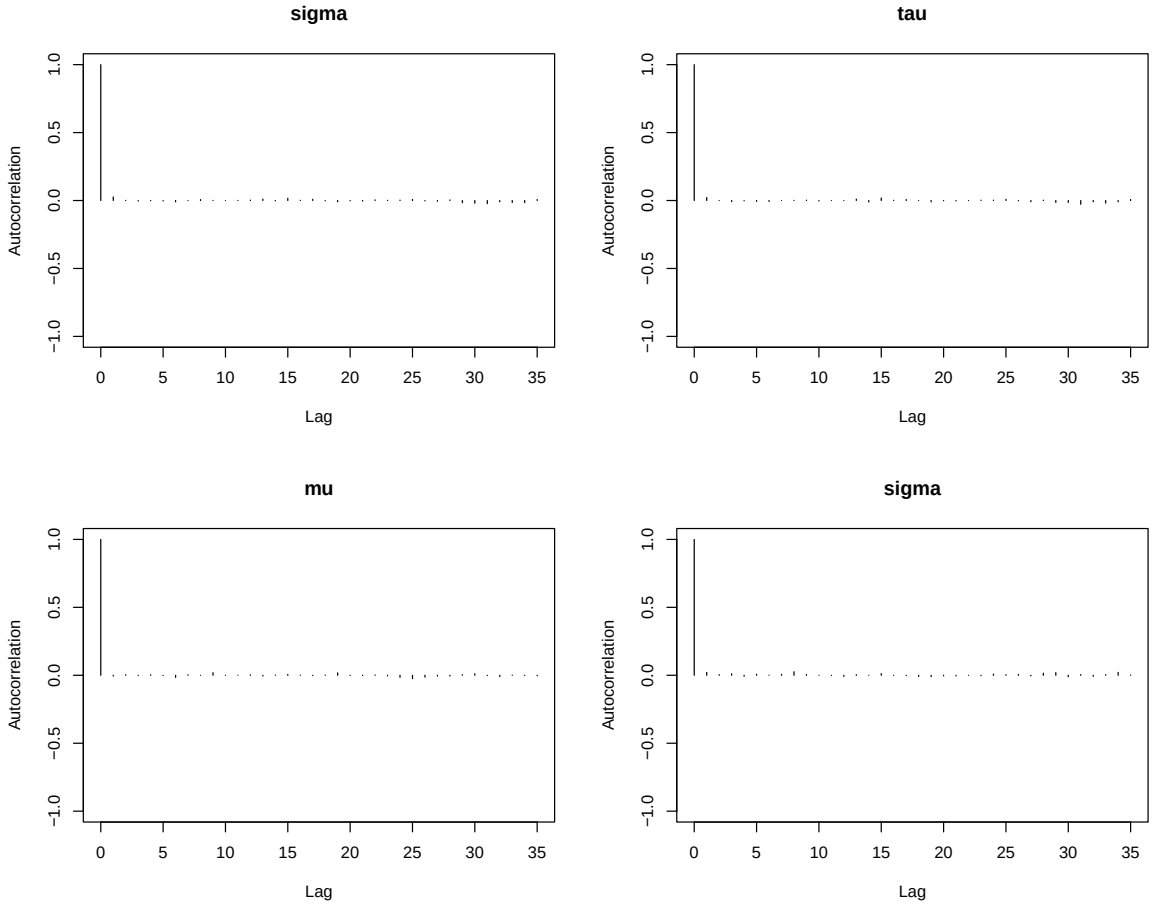


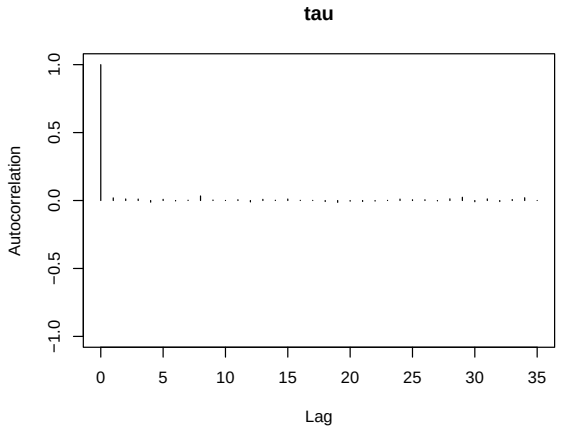

```
plot(mcmc_out)
```




```
coda::autocorr.plot(mcmc_out)
```


mu
sigma
tau
mu
Autocorrelation
Lag
1.0
0.5
0.0
−0.5
−1.0
0
5
10
15
20
25
30
35

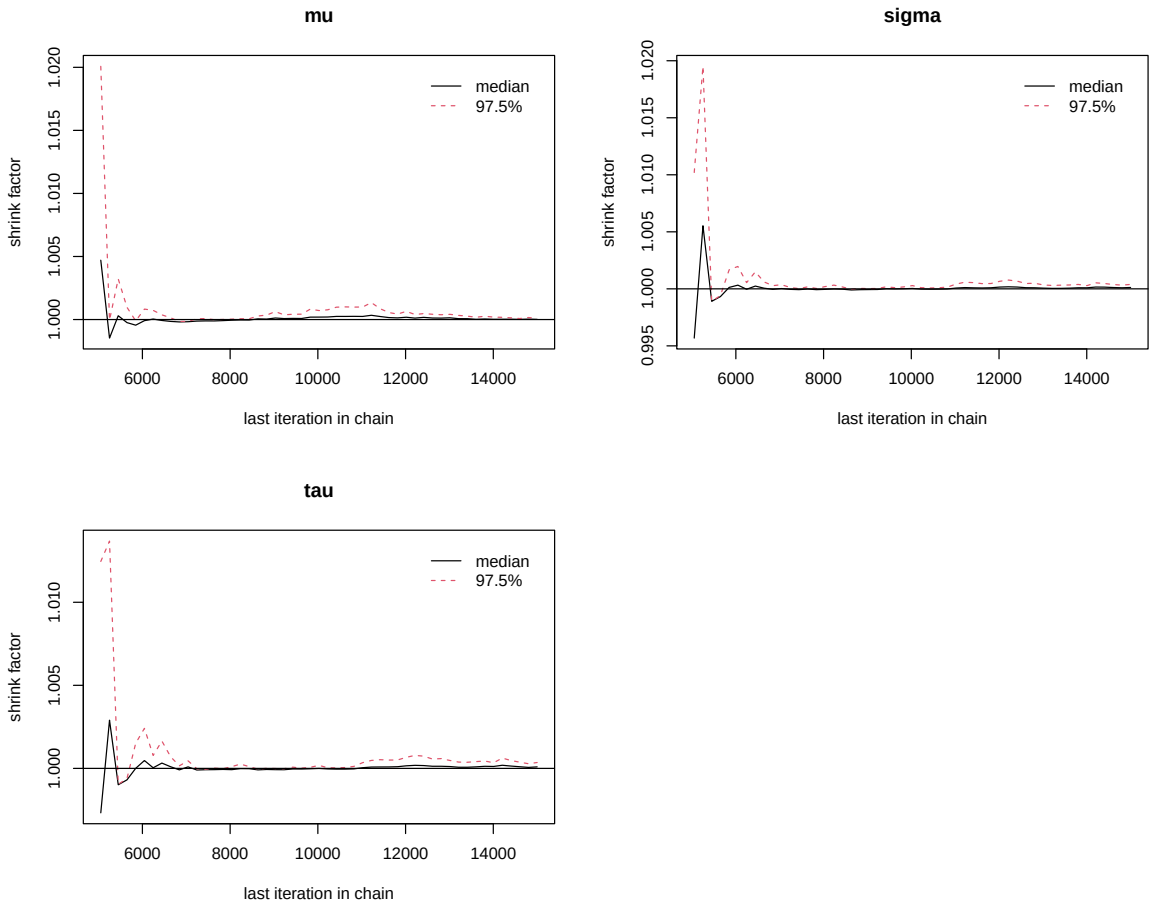

sigma
tau
mu
sigma
Autocorrelation
Lag
1.0
0.5
0.0
−0.5
−1.0
0
5
10
15
20
25
30
35




```
coda::gelman.diag(mcmc_out)
```

```
Potential scale reduction factors:

      Point est. Upper C.I.
mu             1          1
sigma          1          1
tau            1          1

Multivariate psrf

1
```

```
coda::gelman.plot(mcmc_out)
```




```
coda::effectiveSize(mcmc_out)
```

```
      mu    sigma      tau
29633.15 28986.98 27882.13
```

```
coda::autocorr.diag(mcmc_out)
```

```
                mu        sigma          tau
Lag 0    1.000000000  1.000000000  1.000000000
Lag 1    0.002311656  0.022351301  0.020145155
Lag 5    0.001561614  0.001301518 -0.001232817
```

```
Lag 10 -0.013150412 -0.001596771 -0.003168911
Lag 50  0.002270423  0.008500384  0.005792332
```

This Chapter borrows materials from Chapter 6 in Hoff (2009).

7 The Multivariate Gaussian Model

Up to this point, most of our models have been **univariate**: each observational unit contributed a single measurement. In many applications, however, each unit contributes **multiple related measurements**. Examples include:

- repeated measurements on the same person,
- several biomarkers measured on the same patient,
- multiple exam scores for the same student,
- several financial returns observed on the same day.

In such settings, the variables are usually **not independent**. We therefore need a model that can describe not only the behaviour of each variable separately, but also the **dependence structure among variables**.

The multivariate Gaussian model is one of the most important models in Bayesian statistics because it

- jointly models means, variances, and covariances,
- leads to tractable posterior distributions under convenient priors,
- provides a foundation for Gibbs sampling in higher dimensions,
- supports missing-data imputation,
- underlies many later models, including hierarchical normal models, latent Gaussian models, and Gaussian process models.

> **Note**
>
> The multivariate Gaussian model is to multivariate data what the ordinary Gaussian model is to univariate data.

7.1 Why a multivariate model?

So far, we have mostly considered scalar parameters such as

$$\theta \in \mathbb{R}.$$

Now we consider a random vector

$$\mathbf{Y} = (Y_1, \dots, Y_p) \in \mathbb{R}^p.$$

In a multivariate model, we are interested not only in

- the mean of each variable,
- the variance of each variable,

but also in

- the covariance between variables,
- the correlation structure,
- linear combinations of variables,
- prediction of one variable from others.

For example, if a student has both a pre-test score and a post-test score, then we may want to know:

- the average pre-test score,
- the average post-test score,
- whether the post-test mean is larger than the pre-test mean,
- how strongly the two scores are associated.

This is exactly the type of problem the multivariate Gaussian model is designed for.

Suppose each student takes a reading comprehension test **before** and **after** a particular instructional method. For student i, let

$$\mathbf{Y}_i = \begin{pmatrix} Y_{i1} \\ Y_{i2} \end{pmatrix} = \begin{pmatrix} \text{pre-instruction score} \\ \text{post-instruction score} \end{pmatrix}.$$

The population quantities of interest include the mean vector

$$\theta = E[\mathbf{Y}] = \begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix},$$

and the covariance matrix

$$\Sigma = \mathrm{Cov}(\mathbf{Y}) = \begin{pmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{pmatrix}.$$

From these, we can study:

- the difference in population means, $\theta_2 - \theta_1$,
- the variability in each test,
- the correlation between the two scores.

This kind of example is one of the main motivating examples in Chapter 7 in Hoff (2009).

A random vector

$$\mathbf{Y} \in \mathbb{R}^p$$

is said to follow a **multivariate Gaussian** (or multivariate normal) distribution if

$$\mathbf{Y} \sim \mathcal{N}_p(\mu, \Sigma),$$

where

- $\mu \in \mathbb{R}^p$ is the **mean vector**, and
- $\Sigma \in \mathbb{R}^{p \times p}$ is the **covariance matrix**.

We often write

$$E[\mathbf{Y}] = \mu \qquad \text{and} \qquad \text{Cov}(\mathbf{Y}) = \Sigma.$$

To understand those quantities, we need to unpack the mean vector and covariance matrix.

The mean vector is

$$\mu = \left(\mu_1, \mu_2, ..., \mu_p\right).$$

Each component μ_j is the population mean of the jth variable.

The covariance matrix is

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1p} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{p1} & \sigma_{p2} & \cdots & \sigma_p^2 \end{pmatrix}.$$

Interpretation:

- the diagonal entries σ_j^2 are variances,
- the off-diagonal entries σ_{jk} are covariances.

The covariance between Y_j and Y_k tells us whether the variables tend to move together.

A scale-free summary is the correlation

$$\rho_{jk} = \frac{\sigma_{jk}}{\sqrt{\sigma_j^2 \sigma_k^2}}.$$

This value lies between -1 and 1.

A valid covariance matrix must be

- **symmetric**, and
- **positive definite**.

Positive definiteness guarantees, among other things, that all variances are positive and that all correlations are valid.

7.2 The density function

If $\mathbf{Y} \sim \mathcal{N}_p(\mu, \Sigma)$, then its density is

$$p(\mathbf{y}) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{y}-\mu)^\top \Sigma^{-1}(\mathbf{y}-\mu)\right\}.$$

This formula looks more complicated than the univariate Gaussian density, but it is really doing the same two jobs:

1. the determinant term $|\Sigma|^{1/2}$ controls the overall scale;
2. the quadratic form

$$(\mathbf{y}-\mu)^\top \Sigma^{-1}(\mathbf{y}-\mu)$$

measures how far $\mathbf{y}$ is from the centre, taking the covariance structure into account.

Mahalanobis distance

The quantity

$$(\mathbf{y}-\mu)^\top \Sigma^{-1}(\mathbf{y}-\mu)$$

is called the **Mahalanobis distance**. It is the multivariate analogue of

$$\frac{(y-\mu)^2}{\sigma^2}$$

from the univariate Gaussian model.

7.3 Matrix quantities you need to know

To work with the multivariate Gaussian density, it is helpful to remember three matrix ideas.

For a square matrix $\mathbf{A}$, the quantity $|\mathbf{A}|$ is its **determinant**. Roughly speaking, the determinant measures the size or volume associated with the matrix transformation.

For an invertible square matrix $\mathbf{A}$, the inverse $\mathbf{A}^{-1}$ satisfies

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I},$$

where $\mathbf{I}$ is the identity matrix.

For a vector $\mathbf{x}$ and matrix $\mathbf{A}$, the quantity

$$\mathbf{x}^\top \mathbf{A}\mathbf{x}$$

is a **quadratic form**. In the Gaussian density, it measures distance in a way that depends on both scale and correlation.

7.4 Geometric intuition

The multivariate Gaussian distribution has **elliptical contours**.

- If the variables are independent and have equal variances, the contours are circles.
- If the variables are independent but have different variances, the contours are axis-aligned ellipses.
- If the variables are correlated, the ellipses tilt.

This geometric picture is very useful because it helps us understand what covariance actually does.

```
library(MASS)
library(ggplot2)

set.seed(8310)

mu <- c(0, 0)
```

```r
Sigma <- matrix(c(1, 0.8,
                  0.8, 1), 2, 2)

samples <- MASS::mvrnorm(2000, mu = mu, Sigma = Sigma)
df <- data.frame(x = samples[, 1], y = samples[, 2])

ggplot(df, aes(x = x, y = y)) +
  geom_point(alpha = 0.20, size = 0.8) +
  stat_density_2d(color = "blue", linewidth = 0.8) +
  labs(
    title = "Bivariate Gaussian distribution",
    subtitle = "Positive correlation produces tilted elliptical contours",
    x = expression(Y[1]),
    y = expression(Y[2])
  ) +
  theme_classic()
```

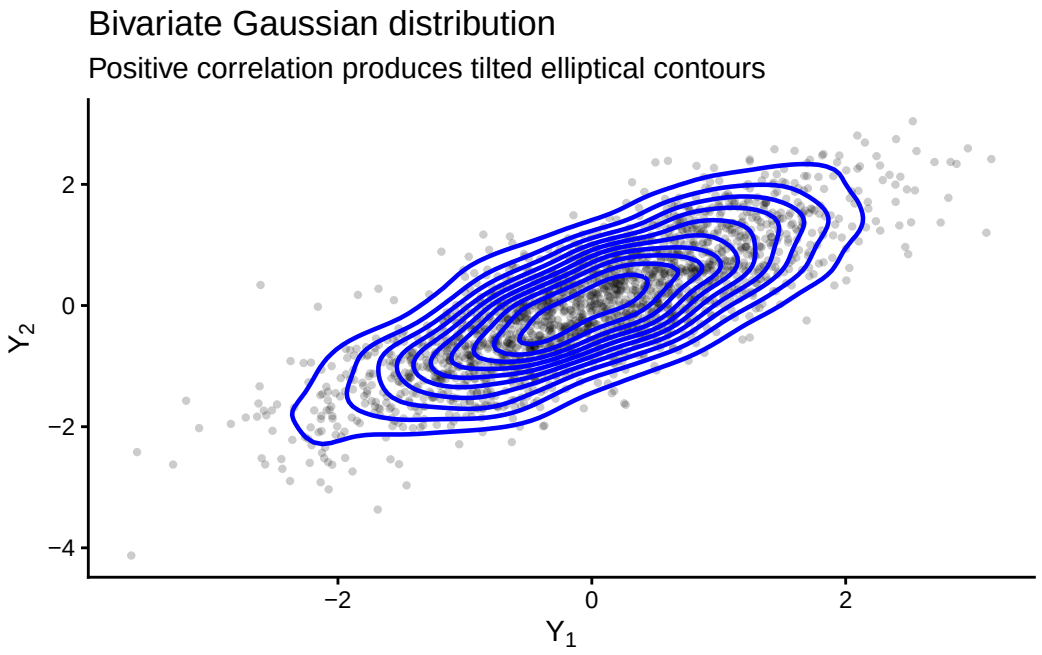


Figure 7.1: A bivariate Gaussian distribution with positive correlation.

The next plot shows how correlation changes the orientation of the distribution.

```r
library(MASS)
library(ggplot2)

set.seed(8310)

make_samples <- function(rho, n = 1500) {
  Sigma <- matrix(c(1, rho, rho, 1), 2, 2)
  out <- MASS::mvrnorm(n, mu = c(0, 0), Sigma = Sigma)
  data.frame(
    x = out[, 1],
    y = out[, 2],
    rho = factor(
      paste0("Correlation = ", rho),
      levels = c("Correlation = -0.7", "Correlation = 0", "Correlation = 0.7")
    )
  )
}

df_rho <- rbind(
  make_samples(-0.7),
  make_samples(0.0),
  make_samples(0.7)
)

ggplot(df_rho, aes(x = x, y = y)) +
  geom_point(alpha = 0.20, size = 0.8) +
  stat_density_2d(color = "blue", linewidth = 0.7) +
  facet_wrap(~ rho, nrow = 1) +
  labs(
    title = "Effect of correlation on a bivariate Gaussian distribution",
    x = expression(Y[1]),
    y = expression(Y[2])
  ) +
  theme_classic()
```

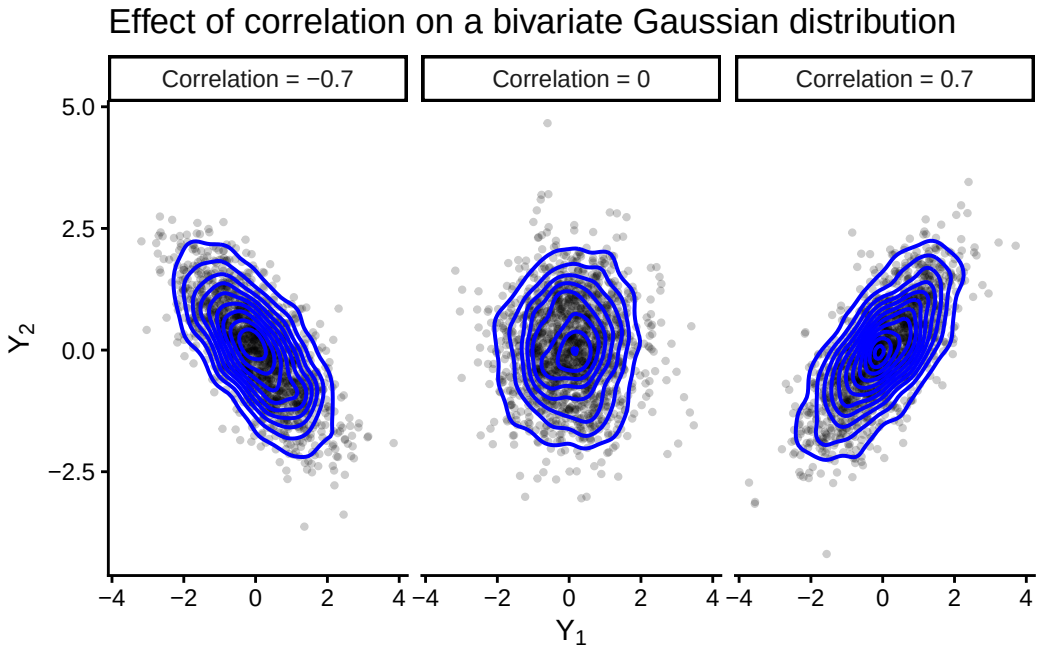


Figure 7.2: Three bivariate Gaussian distributions with the same means and variances but different correlations.

7.5 Marginal distributions

One of the most important properties of the multivariate Gaussian distribution is that **all marginal distributions are Gaussian**.

If

$$\mathbf{Y} \sim \mathcal{N}_p(\mu, \Sigma),$$

then each component satisfies

$$Y_j \sim \mathcal{N}(\mu_j, \sigma_j^2).$$

More generally, if we take any subset of the components of $\mathbf{Y}$, the resulting subvector still has a multivariate Gaussian distribution.

> **Note**
>
> The multivariate Gaussian family is **closed under marginalization**.

This is especially useful in Bayesian analysis because it lets us study subsets of variables without leaving the Gaussian framework.

7.6 Conditional distributions

An equally important property is that **conditional distributions are also Gaussian**.

Suppose we partition the random vector as

$$\mathbf{Y} = \begin{pmatrix} \mathbf{Y}_1 \\ \mathbf{Y}_2 \end{pmatrix},$$

and partition the mean and covariance accordingly:

$$\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \qquad \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}.$$

Then

$$\mathbf{Y}_1 \mid \mathbf{Y}_2 = \mathbf{y}_2 \sim \mathcal{N}\big(\mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{y}_2 - \mu_2),\ \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}\big).$$

7.6.1 Interpretation of the conditional formula

The conditional mean

$$\mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{y}_2 - \mu_2)$$

adjusts the mean of $\mathbf{Y}_1$ according to the observed value of $\mathbf{Y}_2$.

The conditional covariance

$$\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

is smaller than the marginal covariance in the sense that conditioning reduces uncertainty.

This is exactly the mechanism that later makes missing-data imputation possible.

> **Key takeaway**
>
> The multivariate Gaussian family is
>
> - closed under marginalization, and
> - closed under conditioning.
>
> These two properties explain much of its usefulness in Bayesian statistics.

Suppose

$$\begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix} \sim \mathcal{N}_2 \left[\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0.8 \\ 0.8 & 1 \end{pmatrix} \right].$$

Then

$$Y_1 \mid Y_2 = y_2 \sim \mathcal{N}(0.8 y_2,\ 1 - 0.8^2) = \mathcal{N}(0.8 y_2,\ 0.36).$$

So if Y_2 is observed to be large and positive, the conditional mean of Y_1 is also shifted upward.

The next figure illustrates this idea by plotting the conditional mean line.

```
library(ggplot2)

y2_grid <- seq(-3, 3, length.out = 200)
cond_df <- data.frame(
  y2 = y2_grid,
  cond_mean = 0.8 * y2_grid
)

ggplot(cond_df, aes(x = y2, y = cond_mean)) +
  geom_line(linewidth = 1.1, color = "blue") +
  geom_abline(intercept = 0, slope = 0.8, linetype = "dashed", color = "grey40") +
  labs(
    title = "Conditional mean of Y1 given Y2",
    x = expression(Y[2]),
    y = expression(E(Y[1] ~ "|" ~ Y[2]))
  ) +
  theme_classic()
```

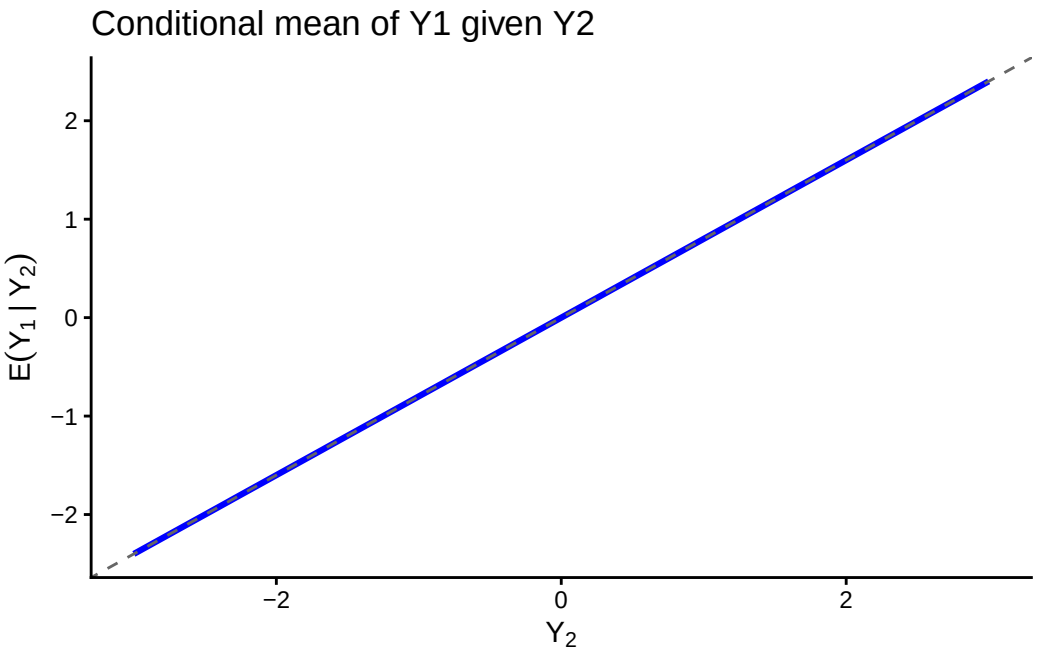


Figure 7.3: The conditional mean of Y1 given Y2 in a bivariate Gaussian distribution.

7.7 Why this matters for Bayesian data analysis

The multivariate Gaussian model is not merely a probability distribution to memorize. It is useful because it supports several key Bayesian tasks.

1. Joint estimation of means and dependence

The model lets us jointly estimate

- multiple population means,
- multiple variances,
- covariances and correlations.

This is often more informative than fitting separate univariate models, because separate univariate analyses ignore the dependence structure.

2. Conjugate and semiconjugate priors

We can show that if

$$\mathbf{Y}_1, \ldots, \mathbf{Y}_n \mid \theta, \Sigma \overset{\text{i.i.d.}}{\sim} \mathcal{N}_p(\theta, \Sigma),$$

and we place a multivariate Gaussian prior on θ, then the conditional posterior of θ given Σ remains multivariate Gaussian. This is the multivariate analogue of the normal-normal update from the univariate case.

3. Gibbs sampling

If we combine a Gaussian prior for the mean vector with an inverse-Wishart prior for the covariance matrix, then the full conditional distributions take convenient forms. This makes Gibbs sampling possible.

4. Missing-data imputation

Because conditional Gaussian distributions are again Gaussian, missing entries in a multivariate observation can be imputed from their conditional distribution given the observed entries. This is a major application in Section 7.5 in Hoff (2009).

7.8 A semiconjugate prior for the mean vector

Suppose

$$\mathbf{Y}_1, \ldots, \mathbf{Y}_n \mid \theta, \Sigma \overset{\text{i.i.d.}}{\sim} \mathcal{N}_p(\theta, \Sigma),$$

and suppose we place the prior

$$\theta \sim \mathcal{N}_p(\mu_0, \Lambda_0).$$

One can show that the conditional posterior distribution of θ given the data and Σ is again multivariate Gaussian:

$$\theta \mid \mathbf{Y}_1, \ldots, \mathbf{Y}_n, \Sigma \sim \mathcal{N}_p(\mu_n, \Lambda_n),$$

where

$$\Lambda_n = (\Lambda_0^{-1} + n\Sigma^{-1})^{-1}$$

and

$$\mu_n = (\Lambda_0^{-1} + n\Sigma^{-1})^{-1}(\Lambda_0^{-1}\mu_0 + n\Sigma^{-1}\bar{\mathbf{y}}).$$

Interpretation

This formula is the direct multivariate analogue of the univariate Gaussian update.

- posterior precision = prior precision + data precision,
- posterior mean = weighted average of prior mean and sample mean.

So the same logic we learned in the univariate model continues to hold, but now in matrix form.

> **Note**
>
> The multivariate Gaussian prior is convenient because it matches the likelihood structure of the Gaussian sampling model.

7.9 The inverse-Wishart distribution

In the univariate Gaussian model, a convenient prior for the variance parameter is the inverse-gamma distribution. In the multivariate case, the analogous prior for the covariance matrix is the **inverse-Wishart distribution**.

In Hoff (2009), it explains the motivation of the Wishart distribution by thinking about empirical covariance matrices. If $\mathbf{z}_1, \ldots, \mathbf{z}_n$ are mean-zero multivariate vectors, then

$$\sum_{i=1}^{n} \mathbf{z}_i \mathbf{z}_i^\top = \mathbf{Z}^\top \mathbf{Z}$$

is a sum-of-squares matrix. The Wishart distribution is the multivariate analogue of the gamma distribution for such matrix-valued quantities.

If

$$\Sigma \sim \text{Inverse-Wishart}(\nu_0, \mathbf{S}_0^{-1}),$$

then

- ν_0 acts like a prior sample size or degrees-of-freedom parameter,
- $\mathbf{S}_0$ controls the prior scale.

A larger ν_0 corresponds to stronger prior information about the covariance structure.

> **Intuition**
>
> The inverse-Wishart prior plays the same role for covariance matrices that the inverse-gamma prior plays for variances in the univariate Gaussian model.

7.10 Conditional posterior of the covariance matrix

Conditional on θ, one can show that the covariance matrix has the full conditional distribution

$$\Sigma \mid \mathbf{Y}_1, \ldots, \mathbf{Y}_n, \theta \sim \text{Inverse-Wishart}(\nu_0 + n, \mathbf{S}_n^{-1}),$$

where

$$\mathbf{S}_n = \mathbf{S}_0 + \mathbf{S}_\theta,$$

and

$$\mathbf{S}_\theta = \sum_{i=1}^{n} (\mathbf{Y}_i - \theta)(\mathbf{Y}_i - \theta)^\top.$$

Interpretation

This mirrors the univariate case as well:

- posterior degrees of freedom = prior degrees of freedom + sample size,
- posterior scale = prior sum-of-squares information + data sum-of-squares information.

This is a very natural update rule.

7.11 Gibbs sampling for the mean and covariance

Because the two full conditional distributions are available in closed form, we can construct a Gibbs sampler for

$$p(\theta, \Sigma \mid \mathbf{Y}_1, \ldots, \mathbf{Y}_n).$$

The Gibbs sampler alternates between:

1. sampling θ from its multivariate Gaussian full conditional,
2. sampling Σ from its inverse-Wishart full conditional.

So each iteration looks like:

$$\theta^{(s+1)} \sim p(\theta \mid \Sigma^{(s)}, \mathbf{Y}),$$

$$\Sigma^{(s+1)} \sim p(\Sigma \mid \theta^{(s+1)}, \mathbf{Y}).$$

This gives a Markov chain whose stationary distribution is the joint posterior.

> **Note**
>
> This is one of the most important examples of Gibbs sampling in Bayesian statistics, because it shows how matrix-valued parameters can still be handled tractably.

7.12 A simulated Gibbs-sampling example

The next chunk uses simulated bivariate Gaussian data to illustrate the structure of a Gibbs sampler for (θ, Σ).

```
library(MASS)
library(ggplot2)

set.seed(8310)

# Simulated bivariate data
n <- 50
theta_true <- c(48, 54)
Sigma_true <- matrix(c(180, 95,
                       95, 240), 2, 2)

Y <- MASS::mvrnorm(n, mu = theta_true, Sigma = Sigma_true)
ybar <- colMeans(Y)

# Prior hyperparameters
mu0 <- c(50, 50)
Lambda0 <- matrix(c(625, 312.5,
                    312.5, 625), 2, 2)
nu0 <- 4
S0 <- Lambda0
```

```r
# Storage
S <- 4000
THETA <- matrix(NA, nrow = S, ncol = 2)
SIGMA <- array(NA, dim = c(2, 2, S))

# Initial value
Sigma <- cov(Y)

for (s in 1:S) {
  # Update theta
  Lambda_n <- solve(solve(Lambda0) + n * solve(Sigma))
  mu_n <- Lambda_n %*% (solve(Lambda0) %*% mu0 + n * solve(Sigma) %*% ybar)
  theta <- as.numeric(MASS::mvrnorm(1, mu = mu_n, Sigma = Lambda_n))

  # Update Sigma
  S_theta <- matrix(0, 2, 2)
  for (i in 1:n) {
    d <- matrix(Y[i, ] - theta, ncol = 1)
    S_theta <- S_theta + d %*% t(d)
  }
  S_n <- S0 + S_theta
  Sigma <- solve(rWishart(1, df = nu0 + n, Sigma = solve(S_n))[,,1])

  THETA[s, ] <- theta
  SIGMA[, , s] <- Sigma
}
```

Traceplots for the posterior mean components

```r
trace_df <- data.frame(
  iter = 1:S,
  theta1 = THETA[, 1],
  theta2 = THETA[, 2]
)

p1 <- ggplot(trace_df, aes(x = iter, y = theta1)) +
  geom_line(linewidth = 0.4) +
  labs(x = "Iteration", y = expression(theta[1]), title = expression("Traceplot of " * theta
  theme_classic()

p2 <- ggplot(trace_df, aes(x = iter, y = theta2)) +
  geom_line(linewidth = 0.4) +
```

```
  labs(x = "Iteration", y = expression(theta[2]), title = expression("Traceplot of " * theta
  theme_classic()

p1
p2
```

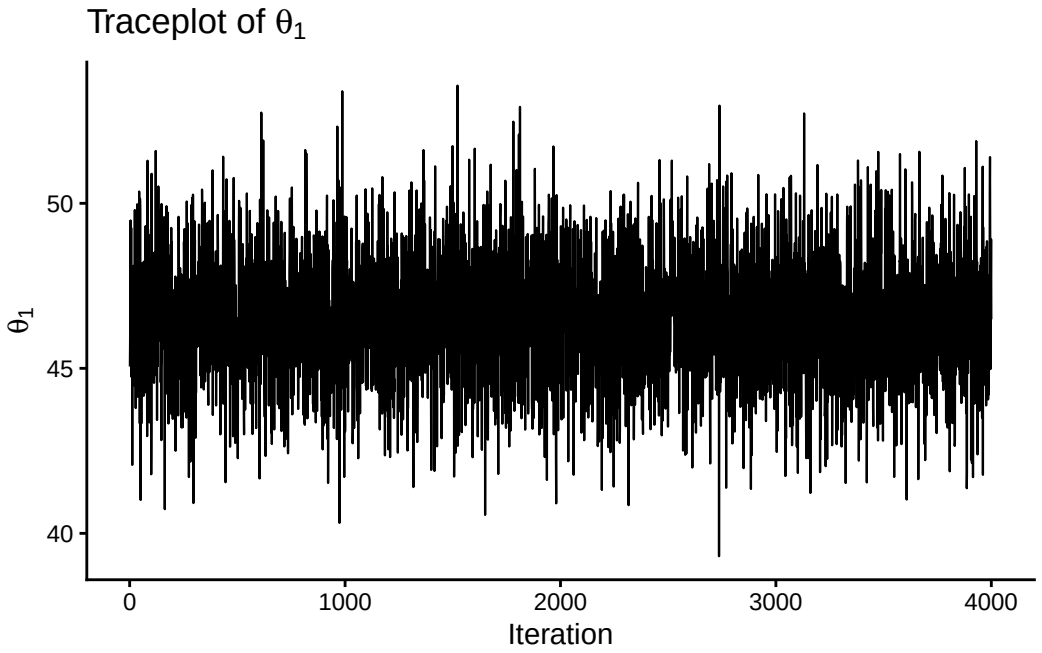


Figure 7.4: Traceplots of the two components of the posterior mean vector from a Gibbs sampler.

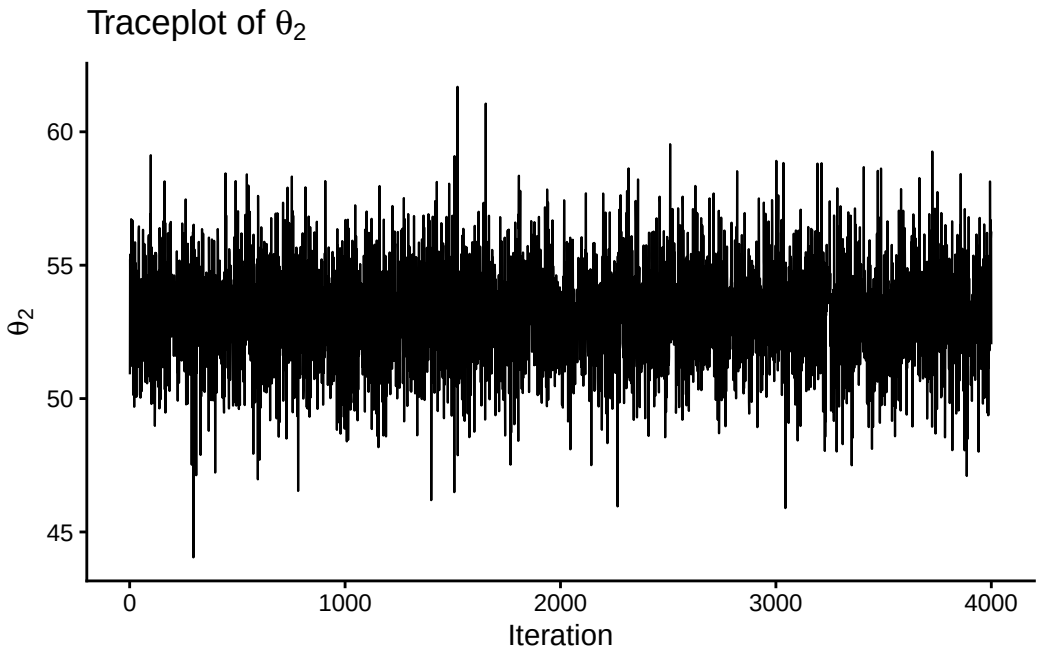


Figure 7.5: Traceplots of the two components of the posterior mean vector from a Gibbs sampler.

You can also look at the traceplots for the covariance matrix entries, but we will not show them here. Also, the acf plot for the mean vector components shows some autocorrelation, which is expected in a Gibbs sampler, as well as the effective sample size.

7.13 Missing data and imputation

A major application of the multivariate Gaussian model is missing-data imputation.

Suppose some components of $\mathbf{Y}_i$ are missing. If the data are **missing at random**, then the observed part of each vector still contributes information about the mean and covariance, and the missing part can be sampled from its conditional distribution given the observed part.

This is attractive because:

- we do not discard incomplete observations,
- we account for uncertainty about the missing values,
- we preserve dependence among variables.

Hoff (2009) emphasizes that this is much better than either:

- deleting incomplete cases, or
- plugging in fixed values such as column means.

> **Important Bayesian lesson**
>
> Missing data can often be treated as additional unknown quantities rather than nuisances to be discarded.

7.14 Why this chapter matters for the rest of the course

This chapter is a bridge between basic Bayesian models and more sophisticated Bayesian workflows.

Once we can work with multivariate Gaussian distributions, we can move naturally to

- hierarchical normal models,
- Bayesian regression,
- mixed-effects models,
- latent Gaussian models,
- Gaussian copulas and Gaussian processes.

So this chapter is foundational not only for multivariate data analysis, but for much of modern Bayesian modelling.

7.15 Summary

The multivariate Gaussian distribution is one of the cornerstones of Bayesian statistics.

Main ideas:

- the mean vector controls location,
- the covariance matrix controls spread and dependence,
- the geometry is elliptical,
- marginal distributions are Gaussian,
- conditional distributions are Gaussian,
- convenient priors lead to tractable full conditional distributions,
- Gibbs sampling becomes feasible,
- missing data can be imputed naturally.

In short,

$$\begin{aligned} &\text{Multivariate Gaussian model} \\ \Longrightarrow\ &\text{joint modelling } + \text{ tractable Bayesian computation.} \end{aligned}$$

This Chapter borrows materials from Chapter 7 in Hoff (2009).

8 Bayesian methods in Machine Learning: Regularization, Prediction, and Uncertainty

In this lecture, we explore several ways in which Bayesian statistics connects naturally with machine learning. The main message is that many ideas in modern machine learning have very natural Bayesian interpretations.

In particular, we will focus on three themes:

- regularization as prior modelling,
- prediction with uncertainty,
- Bayesian thinking in modern machine learning.

8.1 Why connect Bayesian statistics and machine learning?

Many machine learning methods are designed to do one or more of the following:

- estimate complicated models,
- make predictions,
- avoid overfitting,
- quantify uncertainty.

Bayesian methods are also designed to address these goals, but from a **probabilistic perspective**.

A Bayesian analysis starts with a model for the data, specifies prior distributions on unknown parameters, and then uses the posterior distribution to update beliefs and make predictions.

> **Note**
>
> A useful way to think about the connection is:
>
> - machine learning often emphasizes **prediction and optimization**;
> - Bayesian statistics emphasizes **probability models and uncertainty quantification**.

In many cases, the two perspectives are closely related.

8.2 Roadmap

In this lecture we will study:

1. Bayesian linear regression as a probabilistic prediction model,
2. regularization as prior modeling,
3. posterior predictive uncertainty,
4. why these ideas matter in modern machine learning.

8.3 Bayesian linear regression as probabilistic machine learning

Suppose we observe data

$$(y_1, \mathbf{x}_1), \dots, (y_n, \mathbf{x}_n),$$

where each $\mathbf{x}_i \in \mathbb{R}^p$ is a vector of predictors.

A standard linear regression model is

$$y_i = \mathbf{x}_i^\top \beta + \varepsilon_i, \qquad \varepsilon_i \sim \mathcal{N}(0, \sigma^2).$$

Equivalently,

$$y_i \mid \beta, \sigma^2 \sim \mathcal{N}(\mathbf{x}_i^\top \beta, \sigma^2).$$

In matrix form,

$$\mathbf{y} \mid \beta, \sigma^2 \sim \mathcal{N}_n(\mathbf{X}\beta, \sigma^2 \mathbf{I}_n).$$

8.3.1 Frequentist view versus Bayesian view

In ordinary least squares regression, we estimate β by a single point estimate:

$$\hat{\beta} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}.$$

This gives one "best" coefficient vector.

In Bayesian linear regression, we instead treat β as unknown and assign it a prior distribution, such as

$$\beta \sim \mathcal{N}_p(\mathbf{0}, \tau^2 \mathbf{I}_p).$$

Then we obtain the posterior distribution

$$p(\beta \mid \mathbf{y}, \mathbf{X}).$$

So instead of a single estimate, we get a full distribution over plausible values of the regression coefficients.

> **Note**
>
> This is one of the key Bayesian ideas in machine learning:
>
> > Instead of learning one parameter value, we learn a distribution over parameter values.

8.3.2 Why is this useful?

A posterior distribution over β gives us:

- point estimates such as posterior means,
- interval estimates such as credible intervals,
- predictive distributions for new observations,
- uncertainty quantification.

This is especially useful when:

- sample sizes are small,
- predictors are correlated,
- overfitting is a concern,
- prediction uncertainty matters.

8.4 Regularization as prior modeling

One of the most important connections between Bayesian statistics and machine learning is that **regularization** can often be interpreted as a **prior distribution**.

8.4.1 Why regularization?

In machine learning, flexible models can easily overfit the data. To control complexity, we often add a penalty term.

For example, ridge regression solves

$$\hat{\beta}_{\text{ridge}} = \arg\min_{\beta} \left\{ \sum_{i=1}^{n} (y_i - \mathbf{x}_i^\top \beta)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}.$$

The second term penalizes large coefficients.

8.4.2 Bayesian interpretation of ridge regression

Suppose we use the Gaussian prior

$$\beta_j \sim \mathcal{N}(0, \tau^2), \qquad j = 1, \dots, p,$$

independently.

Then the log prior density is proportional to

$$-\frac{1}{2\tau^2} \sum_{j=1}^{p} \beta_j^2.$$

Under a Gaussian likelihood, the log posterior is proportional to

$$-\frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \mathbf{x}_i^\top \beta)^2 - \frac{1}{2\tau^2} \sum_{j=1}^{p} \beta_j^2.$$

Maximizing the posterior is therefore equivalent to minimizing a penalized least squares criterion. In particular, the Gaussian prior corresponds to an L_2 penalty.

> **!** Important
>
> A Gaussian prior on coefficients corresponds to **ridge-type shrinkage**.

So a prior is not only a statement about beliefs. It is also a way to control model complexity.

8.4.3 Bayesian interpretation of the lasso

The lasso solves

$$\hat{\beta}_{\text{lasso}} = \arg\min_{\beta} \left\{ \sum_{i=1}^{n} (y_i - \mathbf{x}_i^\top \beta)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\}.$$

This corresponds to a Laplace prior,

$$p(\beta_j) \propto \exp(-\lambda |\beta_j|).$$

So:

- Gaussian prior ⇒ ridge-type shrinkage,
- Laplace prior ⇒ lasso-type shrinkage.

This is a powerful connection, because it shows that many machine learning penalties can be understood as prior distributions.

8.4.4 Why shrinkage is Bayesianly natural

From a Bayesian point of view, shrinkage happens because the prior says that very large coefficients are unlikely. The posterior combines this prior information with the evidence in the data.

This often improves prediction, especially in noisy or high-dimensional settings.

> **Note**
>
> In machine learning language: regularization controls overfitting.
> In Bayesian language: priors regularize estimation.

8.5 A simple simulation: ordinary least squares versus ridge-style shrinkage

The following example illustrates the effect of shrinkage in a small regression problem.

```r
library(ggplot2)

set.seed(8310)

n <- 40
p <- 8

X <- matrix(rnorm(n * p), nrow = n, ncol = p)
beta_true <- c(2, -1.5, 1, 0, 0, 0, 0, 0)
y <- X %*% beta_true + rnorm(n, sd = 2)
y <- as.numeric(y)

# OLS estimate
beta_ols <- solve(t(X) %*% X, t(X) %*% y)

# Ridge estimate
lambda <- 10
beta_ridge <- solve(t(X) %*% X + lambda * diag(p), t(X) %*% y)

coef_df <- data.frame(
  index = factor(1:p),
  True = beta_true,
  OLS = as.numeric(beta_ols),
  Ridge = as.numeric(beta_ridge)
)

coef_long <- rbind(
  data.frame(index = coef_df$index, value = coef_df$True, method = "True"),
  data.frame(index = coef_df$index, value = coef_df$OLS, method = "OLS"),
  data.frame(index = coef_df$index, value = coef_df$Ridge, method = "Ridge")
)

ggplot(coef_long, aes(x = index, y = value, fill = method)) +
  geom_col(position = "dodge") +
  labs(
    title = "OLS and ridge estimates",
    x = "Coefficient index",
    y = "Value",
    fill = NULL
  ) +
  theme_classic()
```

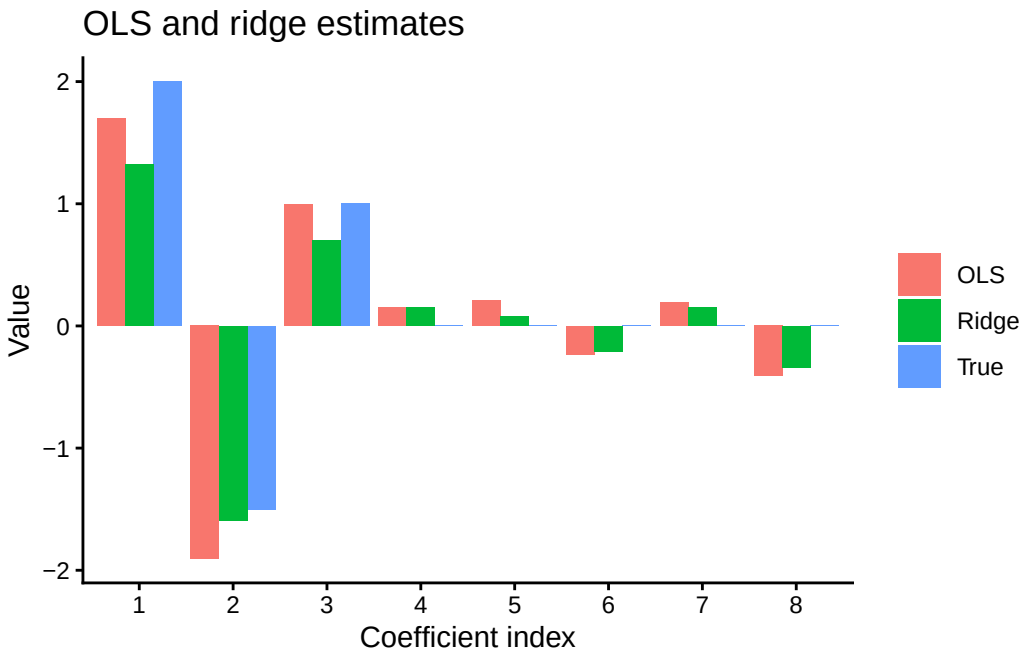


Figure 8.1: Comparison of ordinary least squares and ridge regression coefficient estimates in a small simulated example.

8.6 Discussion of the plot

In many small or noisy datasets, the ordinary least squares estimates can be unstable. Ridge shrinkage tends to pull the estimates toward zero. This can reduce variance and improve predictive performance, even if it introduces some bias.

This is a good example of a more general machine learning principle:

> A small amount of bias can be worthwhile if it substantially reduces variance.

8.7 Prediction and uncertainty

A major strength of Bayesian methods is that prediction is naturally probabilistic.

Suppose we have a new predictor vector $\mathbf{x}_{\text{new}}$. In a non-Bayesian regression analysis, we may plug in an estimate and compute

$$\hat{y}_{\text{new}} = \mathbf{x}_{\text{new}}^{\top} \hat{\beta}.$$

In a Bayesian analysis, we instead use the **posterior predictive distribution**

$$p(y_{\text{new}} \mid \mathbf{y}) = \int p(y_{\text{new}} \mid \beta, \sigma^2)\, p(\beta, \sigma^2 \mid \mathbf{y})\, d\beta\, d\sigma^2.$$

This distribution accounts for both:

- randomness in future observations,
- uncertainty about the parameters.

8.8 Why posterior predictive distributions matter

Posterior predictive distributions are useful because they let us answer questions such as:

- What values are plausible for a future outcome?
- How uncertain is our prediction?
- How do different models compare in predictive performance?

This is especially important in decision-making problems, where uncertainty is not a nuisance but a core part of the problem.

> **!** Important
>
> Machine learning models often emphasize accurate prediction.
> Bayesian models emphasize **accurate prediction with calibrated uncertainty**.

8.9 A simple predictive simulation

The next example illustrates posterior predictive thinking in a regression setting. For simplicity, we simulate from a Gaussian approximation.

```
library(MASS)
library(ggplot2)

set.seed(8310)

n <- 50
x <- runif(n, -2, 2)
y <- 1 + 2 * x + rnorm(n, sd = 1)
```

```r
X <- cbind(1, x)
beta_hat <- solve(t(X) %*% X, t(X) %*% y)
sigma2_hat <- sum((y - X %*% beta_hat)^2) / (n - 2)
V_beta <- sigma2_hat * solve(t(X) %*% X)

# Simulate approximate posterior draws
S <- 4000
beta_draws <- MASS::mvrnorm(S, mu = as.numeric(beta_hat), Sigma = V_beta)

x_new <- 1.0
X_new <- c(1, x_new)

y_new_draws <- numeric(S)
for (s in 1:S) {
  mu_new <- sum(X_new * beta_draws[s, ])
  y_new_draws[s] <- rnorm(1, mean = mu_new, sd = sqrt(sigma2_hat))
}

pred_df <- data.frame(y_new = y_new_draws)

ggplot(pred_df, aes(x = y_new)) +
  geom_histogram(aes(y = after_stat(density)),
                 bins = 40,
                 fill = "grey80",
                 color = "white") +
  labs(
    title = "Posterior predictive distribution",
    subtitle = "Prediction for a new response at x = 1",
    x = expression(y[new]),
    y = "Density"
  ) +
  theme_classic()
```

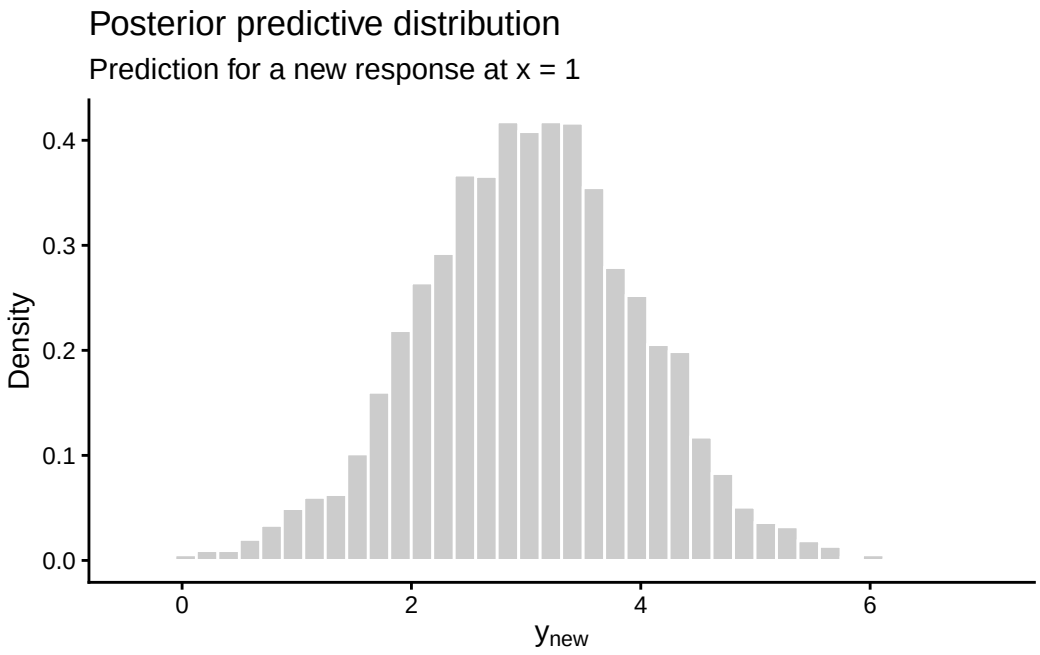


Figure 8.2: Posterior predictive distribution for a new observation in a simple regression example.

8.10 What do we learn from this?

This histogram is not just a point prediction. It is a **distribution** for the future observation.

From it, we can compute:

- predictive means,
- predictive intervals,
- probabilities of exceeding a threshold.

This is often much more useful than a single fitted value.

8.11 Bayesian ideas in modern machine learning

The ideas we have seen so far are not isolated topics. They reappear in many modern machine learning methods.

8.11.1 1. Priors as regularizers

We have already seen that priors can act like penalties. This idea extends far beyond ridge and lasso.

Examples:

- Gaussian priors ⇒ shrinkage,
- Laplace priors ⇒ sparsity,
- hierarchical priors ⇒ adaptive shrinkage.

8.11.2 2. Prediction with uncertainty

Many machine learning models make predictions, but not all provide uncertainty in a principled way.

Bayesian models naturally provide:

- predictive distributions,
- credible intervals,
- probabilities of events.

This is particularly important in applications such as:

- medicine,
- policy,
- finance,
- scientific prediction.

8.11.3 3. Model complexity and overfitting

Bayesian methods can control complexity through the prior. Instead of just asking

> Which parameter values fit the data best?

Bayesian analysis asks

> Which parameter values are plausible after combining prior structure with the observed data?

This can improve stability and generalization.

8.11.4 4. Modern computation

In simple models, posterior distributions may be available in closed form. In more complicated models, Bayesian machine learning relies on computational methods such as:

- MCMC,
- Hamiltonian Monte Carlo,
- variational inference.

This is one reason Bayesian computation has become such an important part of modern statistics and machine learning.

8.12 Bayesian versus machine learning language

It is often helpful to translate between the two languages.

Machine learning language	Bayesian language
Regularization	Prior distribution
Training loss + penalty	Negative log posterior
Prediction	Posterior predictive distribution
Parameter estimate	Posterior summary
Ensemble / uncertainty	Posterior distribution over functions or parameters

> **Note**
>
> These are not exactly the same in every setting, but the parallels are often very strong.

8.13 What Bayesian methods add

It is worth emphasizing that Bayesian methods do not replace machine learning. Rather, they provide a probabilistic framework for many of the same goals.

Bayesian methods add:

- principled uncertainty quantification,
- coherent updating,
- interpretable regularization through priors,
- predictive distributions instead of only point predictions.

8.14 Summary

In this lecture, we explored several important links between Bayesian statistics and machine learning.

Main ideas:

- Bayesian linear regression is a probabilistic predictive model.
- Regularization can often be interpreted as prior modeling.
- Posterior predictive distributions quantify uncertainty in predictions.
- Many modern machine learning ideas have natural Bayesian counterparts.

In short,

$$\text{Bayesian statistics} \iff \text{probabilistic machine learning.}$$

9 Modern Bayesian Computation and Models

In the previous lecture, we discussed several connections between Bayesian statistics and machine learning (ML), focusing on

- Bayesian linear regression,
- regularization as prior modelling,
- and posterior predictive uncertainty.

In this lecture, we continue that theme by looking at two major directions in modern Bayesian statistics:

- **modern Bayesian computation**, especially variational inference;
- **modern Bayesian modelling**, especially Gaussian processes.

The main message of this lecture is:

> Modern Bayesian statistics is not only about priors and posteriors. It is also about how to compute them efficiently and how to build flexible probabilistic models.

9.1 Roadmap

In this lecture we will study:

1. why classical MCMC is not always enough,
2. variational inference as optimization-based Bayesian approximation,
3. Gaussian processes as priors over functions,
4. why these ideas matter in modern statistics and ML

Question: Why do we need modern Bayesian computation?

So far in this course, we have focused heavily on models where posterior inference is tractable or can be handled using Gibbs sampling.

This works very well in many classical Bayesian models. However, in modern applications, we often encounter:

- high-dimensional parameter spaces,
- nonconjugate models,

- large datasets,
- complex posterior geometry.

In such situations, exact posterior sampling may be slow or difficult.

> **Note**
>
> The main computational challenge in Bayesian statistics is:
> How do we approximate a posterior distribution accurately enough, but also efficiently enough, for modern problems?

9.2 A short review of MCMC

MCMC methods approximate posterior distributions by generating dependent samples

$$\phi^{(1)}, \dots, \phi^{(S)}$$

from a MC whose stationary distribution is the target posterior.

Then posterior expectations are approximated by

$$E[g(\phi) \mid y] \approx \frac{1}{S} \sum_{s=1}^{S} g(\phi^{(s)}).$$

This is a powerful idea, but MCMC can become computationally expensive when:

- the chain mixes slowly,
- the dimension is large,
- the likelihood is costly to evaluate,
- or we need to analyze many datasets quickly.

9.3 Beyond classical MCMC

Modern Bayesian computation includes methods such as

- Hamiltonian MC,
- variational inference,
- stochastic gradient MCMC,
- approximate Bayesian computation.

In this lecture, we focus on **variational inference**, because it is one of the clearest examples of the connection between Bayesian computation and ML

9.4 Variational inference

9.4.1 Core idea

Suppose the true posterior is

$$p(\theta \mid y),$$

but this distribution is hard to compute or sample from directly.

Instead of drawing MCMC samples, variational inference approximates the posterior by choosing a simpler distribution

$$q(\theta)$$

from a family of tractable distributions, and then finding the member of that family $\mathcal{Q}$ that is "closest" to the posterior.

The approximation problem becomes

$$q^*(\theta) = \arg\min_{q \in \mathcal{Q}} \mathrm{KL}(q(\theta) \,\|\, p(\theta \mid y)).$$

So variational inference turns posterior approximation into an **optimization problem**.

> **!** Important
>
> MCMC is based on **sampling**.
> Variational inference is based on **optimization**.

Why is this useful?

Variational inference can be **much faster than MCMC**, especially in large or complicated models. This is one reason it is widely used in ML and large-scale Bayesian modelling.

The tradeoff is that the **approximation may be biased**, because we are restricting attention to a smaller family of distributions.

The Kullback–Leibler (KL) divergence from q to p is

$$\mathrm{KL}(q \,\|\, p) = \int q(\theta) \log \frac{q(\theta)}{p(\theta \mid y)} \, d\theta.$$

This quantity is always nonnegative, and it equals zero only if $q = p$ almost everywhere.

So minimizing KL divergence tries to make q as close as possible to the posterior.

9.4.2 The ELBO

Directly minimizing

$$\mathrm{KL}(q(\theta) \,\|\, p(\theta \mid y))$$

is not always convenient, because the posterior normalization constant may be unknown.

Instead, variational inference often maximizes the **evidence lower bound** (ELBO),

$$\mathrm{ELBO}(q) = E_q[\log p(y, \theta)] - E_q[\log q(\theta)].$$

Maximizing the ELBO is equivalent to minimizing the KL divergence.

Interpretation of the ELBO

The ELBO has two parts:

- a **fit** term, which encourages q to place mass where the joint density $p(y, \theta)$ is large;
- an **entropy** term, which discourages q from collapsing too much.

This balance is similar in spirit to many machine learning optimization problems, where we balance fit and complexity.

9.4.3 Mean-field variational inference

A common simplification is to assume that the approximation factorizes:

$$q(\theta) = \prod_{j=1}^{p} q_j(\theta_j).$$

This is called the **mean-field approximation**.

It makes optimization easier, but it can underestimate posterior dependence.

> **Warning**
>
> A common limitation of variational inference is that it may underestimate posterior uncertainty, especially when the true posterior has strong dependence.

9.4.4 Variational inference versus MCMC

The table below summarizes the broad comparison.

Feature	MCMC	Variational inference
Main idea	Sampling	Optimization
Accuracy	Often high	Approximate
Speed	Can be slow	Often faster
Output	Samples from posterior	Approximate distribution
Uncertainty quantification	Usually strong	May be underestimated

9.4.5 A simple variational approximation example

The following code illustrates a very simple variational-style approximation idea. We compare a target density with a Gaussian approximation.

This is not a full general-purpose VI algorithm. The goal is only to build intuition.

```
library(ggplot2)

theta_grid <- seq(-4, 4, length.out = 1000)

# Example target density: a non-Gaussian posterior-like shape
target_unnorm <- exp(-0.5 * (theta_grid - 1)^2) * (1 + 0.3 * sin(3 * theta_grid))
target_unnorm[target_unnorm < 0] <- 0
target_density <- target_unnorm / sum(target_unnorm) / (theta_grid[2] - theta_grid[1])

# A simple Gaussian approximation
q_density <- dnorm(theta_grid, mean = 0.9, sd = 0.8)

df_vi <- data.frame(
  theta = rep(theta_grid, 2),
  density = c(target_density, q_density),
  curve = factor(rep(c("Target density", "Gaussian approximation"),
                     each = length(theta_grid)))
```

```
)

ggplot(df_vi, aes(x = theta, y = density, color = curve, linetype = curve)) +
  geom_line(linewidth = 1) +
  labs(
    title = "Variational inference intuition",
    subtitle = "Approximate the target posterior by a simpler distribution",
    x = expression(theta),
    y = "Density",
    color = NULL,
    linetype = NULL
  ) +
  theme_classic()
```

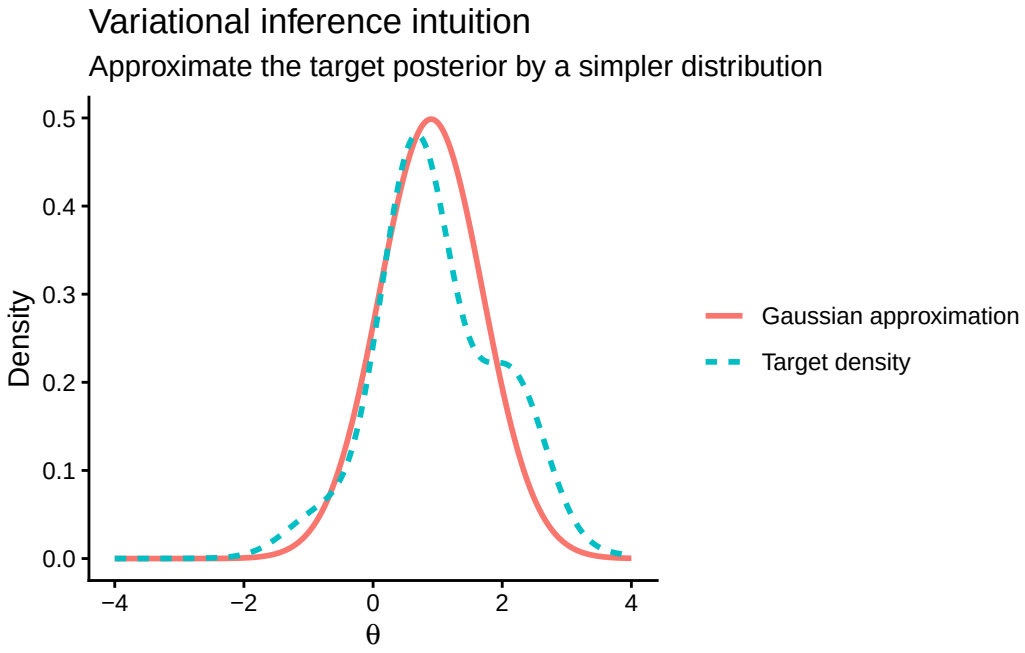


Figure 9.1: A target density and a simple Gaussian approximation.

9.4.6 Takeaway from variational inference

Variational inference is attractive because it scales well and turns Bayesian inference into optimization. This makes it very natural for modern machine learning.

At the same time, we should remember:

- it is an approximation,
- its quality depends on the approximation family,
- and it may not capture all posterior dependence or uncertainty.

9.5 Gaussian processes

We now turn to a different modern Bayesian idea: **Gaussian processes** (GP).

Whereas variational inference is mainly about computation, GPs are mainly about modelling.

9.5.1 Motivation

In regression, we often assume a parametric relationship such as

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i.$$

But what if the true relationship is nonlinear and we do not want to commit to a specific parametric form?

A GP provides a Bayesian way to model an unknown function flexibly.

9.5.2 Core idea

In GP regression, we write

$$y_i = f(x_i) + \varepsilon_i, \qquad \varepsilon_i \sim \mathcal{N}(0, \sigma^2),$$

and then place a prior directly on the unknown function f.

A **GP prior** says that for any collection of inputs

$$x_1, \dots, x_n,$$

the corresponding function values

$$(f(x_1), \dots, f(x_n))^\top$$

follow a multivariate Gaussian distribution.

In symbols,

$$f \sim \mathcal{GP}(m(\cdot), k(\cdot, \cdot)),$$

where

- $m(x)$ is the mean function,
- $k(x, x')$ is the covariance kernel.

> **Note**
>
> A GP is an infinite-dimensional generalization of the multivariate Gaussian distribution.

This is exactly why GP fit naturally after our earlier lecture on multivariate Gaussian models.

9.5.3 The kernel function

The kernel

$$k(x, x')$$

determines how strongly the function values at x and x' are related.

A common choice is the squared exponential kernel:

$$k(x, x') = \alpha^2 \exp\left(-\frac{(x - x')^2}{2\ell^2}\right),$$

where

- α^2 controls vertical variability,
- ℓ controls smoothness.

Large ℓ leads to smoother functions. Small ℓ allows more rapid local variation.

9.5.4 Why Gaussian processes are Bayesian

GP regression is Bayesian because:

- we specify a prior over functions,
- data update this prior to a posterior over functions,
- prediction is based on the posterior predictive distribution.

So the output is not just one estimated curve. It is a posterior distribution over plausible curves.

9.5.5 Simulating prior draws from a GP

The following example illustrates the idea of a prior over functions.

```
library(MASS)
library(ggplot2)

set.seed(8310)

x_grid <- seq(-3, 3, length.out = 100)

kernel_se <- function(x1, x2, alpha = 1, ell = 1) {
  alpha^2 * exp(-(outer(x1, x2, "-")^2) / (2 * ell^2))
}

K <- kernel_se(x_grid, x_grid, alpha = 1, ell = 1)
K <- K + 1e-8 * diag(length(x_grid))  # numerical stability

gp_draws <- MASS::mvrnorm(5, mu = rep(0, length(x_grid)), Sigma = K)

gp_df <- do.call(rbind, lapply(1:5, function(j) {
  data.frame(x = x_grid, y = gp_draws[j, ], draw = factor(paste("Draw", j)))
}))

ggplot(gp_df, aes(x = x, y = y, color = draw)) +
  geom_line(linewidth = 0.9) +
  labs(
    title = "Gaussian process prior draws",
    subtitle = "A Gaussian process defines a prior distribution over functions",
    x = "x",
    y = "f(x)",
    color = NULL
  ) +
  theme_classic()
```

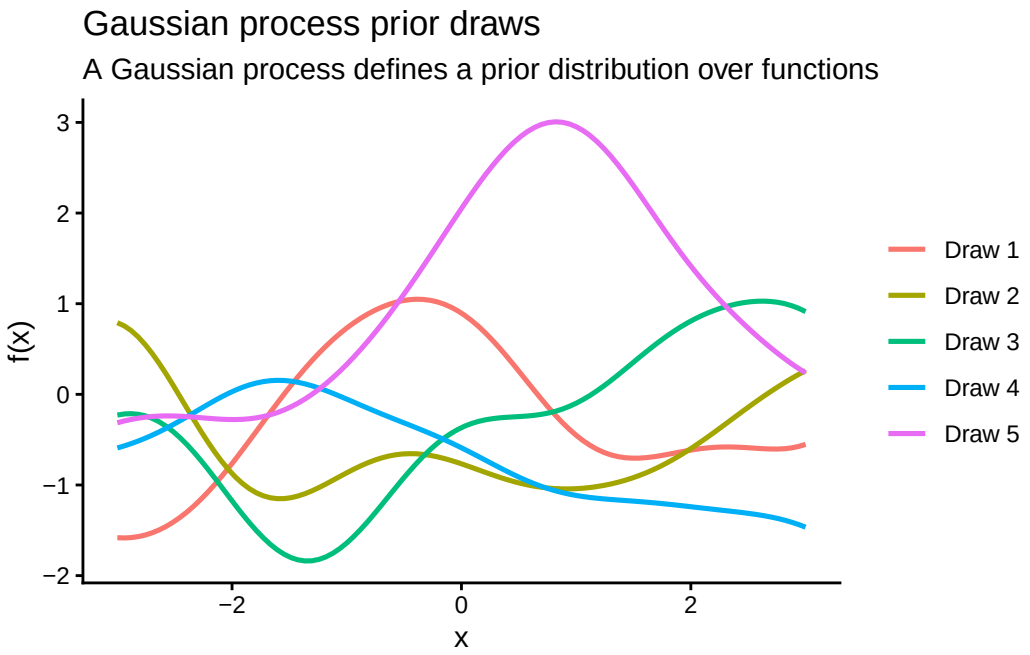


Figure 9.2: Draws from a Gaussian process prior.

9.5.6 Why Gaussian processes matter

GPs are important because they combine:

- flexible nonlinear regression,
- uncertainty quantification,
- elegant Bayesian updating.

They are widely used in:

- spatial statistics,
- Bayesian optimization,
- surrogate modeling,
- computer experiments,
- probabilistic machine learning.

9.5.7 GPs and multivariate Gaussian distributions

A Gaussian process is built from multivariate Gaussian distributions. If we evaluate the unknown function at finitely many input points, then the resulting vector is multivariate Gaussian.

So the structure is:

$$f \sim \mathcal{GP}(m, k) \quad \Longrightarrow \quad (f(x_1), \ldots, f(x_n))^\top \sim \mathcal{N}_n(\mathbf{m}, \mathbf{K}).$$

This is one reason the multivariate Gaussian distribution is so central in modern Bayesian modeling.

9.6 Bigger picture: modern Bayesian statistics

We can now summarize two big themes.

Theme 1: Bayesian computation

Modern Bayesian methods often need computational tools beyond closed-form conjugate analysis.

Examples include:

- MCMC,
- Hamiltonian Monte Carlo,
- variational inference.

Theme 2: Bayesian modeling

Modern Bayesian statistics also provides very flexible modeling tools.

Examples include:

- hierarchical models,
- Gaussian processes,
- latent variable models,
- Bayesian neural networks.

These ideas allow us to model uncertainty in complicated problems while still keeping a probabilistic interpretation.

Machine learning perspective	Bayesian perspective

9.6.1 Bayesian and machine learning: a broader view

At this point, we can see that Bayesian statistics and machine learning overlap in several deep ways.

Machine learning perspective	Bayesian perspective
Optimization	Posterior approximation
Regularization	Prior modeling
Flexible function fitting	Nonparametric Bayes / Gaussian processes
Predictive modeling	Posterior predictive inference
Ensemble uncertainty	Distribution over parameters or functions

> **Note**
>
> Bayesian statistics does not replace ML Rather, it provides a principled probabilistic framework for many ML goals.

Take home message

If you remember only a few things from this lecture, they should be:

1. **Modern Bayesian inference often requires approximation.**
   Variational inference is one major example.
2. **Modern Bayesian models can be highly flexible.**
   Gaussian processes are one major example.
3. **Bayesian methods provide more than point estimates.**
   They provide uncertainty quantification for parameters, functions, and predictions.

9.7 Summary

In this lecture, we studied two important modern Bayesian ideas.

Variational inference

- turns posterior approximation into optimization,
- is often faster than MCMC,

- but may introduce approximation bias.

Gaussian processes

- define priors over functions,
- extend multivariate Gaussian ideas to **infinite-dimensional** settings,
- provide flexible nonlinear regression with uncertainty quantification.

In short,

$$\text{Modern Bayesian statistics} = \text{probabilistic modeling} + \text{modern computation}.$$

9.8 Looking back on the course

This course has moved from:

- basic Bayesian probability and prior-posterior updating,
- conjugate models,
- MC approximation,
- Gibbs sampling and diagnostics,
- multivariate Gaussian models,

to ideas that connect directly to modern ML and modern Bayesian data analysis.

The underlying message has stayed the same:

> Bayesian statistics is a coherent way to model uncertainty, learn from data, and make predictions.

Part I

Project

Bayesian Methods Beyond the Course

Weight: 20%
Due date: May 1, 2026
Last update: April 16, 2026

Objective

The goal of this project is to explore a Bayesian topic that goes beyond the core course material and to demonstrate

- conceptual understanding,
- computational implementation,
- critical thinking, and
- clear communication.

You are encouraged to go somewhat beyond what we covered in class. The emphasis is not only on fitting a Bayesian model, but also on **evaluating, comparing, and interpreting Bayesian methods thoughtfully**.

Collaboration Policy

- **Master's students** may work in groups of up to **2 students**.
- **PhD students must work independently**.

All submissions must clearly indicate the names of contributors and, when applicable, briefly describe each member's contribution.

Group Information

Group	Member	Topic
1	Wenpu Ma	Bayesian Linear Regression Analysis with Application in Diabetes Progression Data
2	Ruihang Han	Bayesian Poisson regression for count data
3	Zilong Zhang	Bayesian Autoregressive Time Series Modelling
4	Kalen Jinnah	Dirichlet Process
5	Zhe Zhong, Michael Dixon	Gaussian Process
6	Yang Zong	Hierarchical Bayesian Regression
7	Taiwo Ayeni	Bayesian Logistic Regression with Application in Framingham Heart Study
8	Andrew Krause	Metropolis-Hastings algorithm
9	Felix Sarpong, Akomanyi-Addo Alleswell-Ato	Bayesian Finite Mixture Modeling
10	Fadil-Rahman Ibrahim	Sequential Monte Carlo/Particle Filter

Page Limit

- The main report is limited to **12 pages**.
- This limit **excludes references and appendices**.
- Appendices may include:
  - additional derivations,
  - extra figures,
  - supplementary results, and
  - code.

The main text should focus on **clarity, key ideas, and main results**.

Project Options

You may choose **one** of the following directions.

Option 1: Method Exploration

Study a Bayesian method that was not fully covered in class.

Possible topics include:

- Metropolis–Hastings algorithm
- Hamiltonian Monte Carlo (HMC)
- Variational Bayes
- Bayesian model selection
- Bayesian nonparametrics (for example, Dirichlet process models)
- Gaussian processes
- Hierarchical Bayesian models

Option 2: Applied Bayesian Analysis

Conduct a full Bayesian analysis on a real dataset, going beyond simple conjugate models.

Possible topics include:

- Bayesian regression (linear or logistic)
- Hierarchical models for grouped data
- Mixture models
- Time series models
- Spatial models

Option 3: Comparison Study

Compare two Bayesian approaches, models, or prior choices.

Possible comparisons include:

- Gibbs vs. Metropolis–Hastings

- MCMC vs. variational inference
- JAGS vs. Stan
- Bayesian vs. frequentist analysis
- informative prior vs. weakly informative prior
- hierarchical vs. non-hierarchical models

You are also welcome to propose your own topic. If you are unsure whether your idea is appropriate, please discuss it with the instructor.

Required Components

Your project should be written as a short report containing the following components.

1. Introduction and Background [10%]

- What is the problem or question?
- Why is a Bayesian approach appropriate?
- What background does the reader need in order to understand your project?

2. Model and Method [15%]

- Clearly describe the model:

$$p(y \mid \theta), \qquad p(\theta)$$

- Explain the inferential or computational method being used.
- Define the main quantities of interest.

3. Computation [20%]

- Implement your analysis using:
  - R (custom code), and/or
  - JAGS, Stan, or another Bayesian software tool.
- Clearly explain the computational procedure.
- Include enough code and explanation for the analysis to be reproducible.

4. Results and Inference [15%]

Present the main results of your analysis, such as

- posterior summaries,
- credible intervals,
- posterior probabilities, and
- model-based conclusions.

5. Diagnostics and Evaluation [10%]

Include appropriate diagnostics and discuss the quality of the fit or computation.

Examples include:

- trace plots,
- convergence diagnostics,
- posterior predictive checks, and
- discussion of model limitations.

6. Comparison Component [10%]

Your project must include **at least one meaningful comparison**, for example:

- between two methods,
- between two models, or
- between two prior choices.

You should explain

- what is being compared,
- why the comparison is meaningful, and
- what conclusion you draw.

7. Simulation Study or Empirical Evaluation [10%]

You must include **one** of the following:

- a simulation study under controlled settings, or
- a systematic empirical evaluation.

Your study should

- state a clear goal,
- vary at least one factor (such as sample size, prior choice, or noise level), and
- summarize findings using plots or tables.

8. Interpretation and Discussion [5%]

- Explain the results in clear, plain language.
- Discuss what you learned from the project.
- Comment on strengths, weaknesses, or possible extensions.

9. Reflection [5%]

Briefly discuss

- what worked well,
- what was difficult, and
- what you would improve if you had more time.

Key Requirement

Your project must answer at least one **decision question**, such as:

- Which model is better, and why?
- How sensitive are the results to the prior?
- Does the method scale well?
- What are the practical limitations of the approach?

Your conclusions should be supported by evidence from your analysis.

Grading Rubric

Component	Excellent	Good	Needs Improvement	Points
Introduction and Background	Problem is clearly motivated and the Bayesian context is well explained	Motivation is adequate but could be sharper	Motivation is unclear or incomplete	10
Model and Method	Model and method are clearly specified and well justified	Mostly clear, with minor gaps	Important pieces are missing or unclear	15
Computation	Code is correct, reproducible, and well explained	Mostly correct with minor issues	Major errors or weak reproducibility	20
Results and Inference	Results are clearly presented and interpreted appropriately	Results are mostly clear	Results are incomplete or poorly explained	15
Diagnostics and Evaluation	Diagnostics are appropriate and thoughtfully discussed	Diagnostics are present but limited	Little or no meaningful diagnostics	10
Comparison Component	Comparison is meaningful and conclusions are well supported	Comparison is present but limited	Comparison is weak or missing	10
Simulation / Empirical Evaluation	Evaluation is well designed and clearly presented	Evaluation is adequate	Evaluation is weak, incomplete, or missing	10
Interpretation and Discussion	Strong interpretation and insight	Some interpretation, but limited depth	Minimal or unclear interpretation	5
Reflection	Thoughtful and specific	Adequate but brief	Minimal	5
Total				**100**

Deliverables

Submit the following:

1. **PDF report**
2. **Source file** (`.Rmd` or `.qmd`)
3. Any additional code files needed to reproduce the analysis

Your submission must be reproducible.

Recommended Report Structure

A typical project report may be organized as follows:

1. Introduction
2. Background
3. Model / Method
4. Computation
5. Results
6. Diagnostics / Evaluation
7. Comparison or Simulation Study
8. Discussion and Reflection

Bonus (up to +5%)

You may earn bonus points for work that goes meaningfully beyond the basic requirements, such as:

- implementing your own MCMC algorithm,
- comparing multiple models in a thoughtful way,
- using advanced tools such as Stan or PyMC, or
- providing especially strong insight or creativity in the analysis.

⚠ Important Notes

- You do **not** need to produce a perfect or publishable analysis.
- Start with a simple version, then extend it.
- A clear and well-executed project is better than an overly ambitious but incomplete one.
- Clarity, reasoning, and interpretation matter as much as technical complexity.

Summary

This project is about learning by exploring Bayesian ideas beyond the classroom.

$$\text{Model} \longrightarrow \text{Computation} \longrightarrow \text{Evaluation} \longrightarrow \text{Insight}$$

Focus on

- understanding,
- computation,
- comparison, and
- communication.

Good luck. This is your opportunity to explore a Bayesian topic that genuinely interests you!

Part II

Appendix

Introduction to R

R

For conducting analyses with data sets of hundreds to thousands of observations, calculating by hand is not feasible and you will need a statistical software. **R** is one of those. **R** can also be thought of as a high-level programming language. In fact, **R** is one of the top languages to be used by data analysts and data scientists. There are a lot of analysis packages in **R** that are currently developed and maintained by researchers around the world to deal with different data problems. Most importantly, **R** is free! In this section, we will learn how to use **R** to conduct basic statistical analyses.

IDE

Rstudio

RStudio is an integrated development environment (IDE) designed specifically for working with the **R** programming language. It provides a user-friendly interface that includes a source editor, console, environment pane, and tools for plotting, debugging, version control, and package management. RStudio supports both **R** and Python and is widely used for data analysis, statistical modeling, and reproducible research. It also integrates seamlessly with tools like **R** Markdown, Shiny, and Quarto, making it popular among data scientists, statisticians, and educators.

Visual Studio Code (VS Code)

VS Code is a versatile code editor that supports multiple programming languages, including **R**. With the **R** extension for VS Code, users can write and execute **R** code, access **R**'s console, and utilize features like syntax highlighting, code completion, and debugging. While not as specialized as RStudio for **R** development, VS Code offers a lightweight alternative with extensive customization options and support for various programming tasks.

Positron

Positron IDE is the next-generation integrated development environment developed by Posit, the company behind RStudio. Designed to be a modern, extensible, and language-agnostic IDE, Positron builds on the strengths of RStudio while supporting a broader range of languages and workflows, including **R**, Python, and Quarto.

RStudio Layout

RStudio consists of several panes: - **Source**: Where you write scripts and markdown documents. - **Console**: Where you type and execute **R** commands. - **Environment/History**: Shows your variables and command history. - **Files/Plots/Packages/Help/Viewer**: For file management, viewing plots, managing packages, accessing help, and viewing web content.

R Scripts

R scripts are plain text files containing **R** code. You can create a new script in RStudio by clicking `File > New File > R Script`.

R Help

Use `?function_name` or `help(function_name)` to access help for any **R** function. For example:

```
?mean
help(mean)
```

R Packages

Packages extend **R**'s functionality. There are thousands of packages available in **R** ecosystem. You may install them from different sources.

With Comprehensive R Archive Network (CRAN)

CRAN is the primary repository for **R** packages. It contains thousands of packages that can be easily installed and updated.

Install a package with:

```
install.packages("package_name")
```

With Bioconductor

Bioconductor is a repository for bioinformatics packages in **R**. It provides tools for the analysis and comprehension of high-throughput genomic data.

Install Bioconductor packages using the `BiocManager` package:

```
BiocManager::install("package_name")
```

From GitHub

Many of the authors of **R** packages host their work on GitHub. You can install these packages using the `devtools` package:

```
devtools::install_github("username/package_name")
```

Load a package

Once a package is installed, you need to load it into your **R** session to use its functions:

```
library(package_name)
```

Alternatively, you may use a function in the package with `package_name::function_name()` without loading the entire package.

R Markdown

R Markdown allows you to combine text, code, and output in a single document. Create a new **R** Markdown file in RStudio via `File > New File > R Markdown...`.

Recently, the posit team has developed a new version of the **R** Markdown called quarto document, with the file extension `.qmd`. It is still under rapid development.

Vectors

Vectors are the most basic data structure in **R**.

```
x <- c(1, 2, 3, 4, 5)
x
```

```
[1] 1 2 3 4 5
```

You can perform operations on vectors:

```
x * 2
```

```
[1]  2  4  6  8 10
```

Data Sets

Data frames are used for storing data tables. Create a data frame:

```
df <- data.frame(Name = c("Alice", "Bob"), Score = c(90, 85))
df
```

```
   Name Score
1 Alice    90
2   Bob    85
```

You can import data from files using `read.csv()` or `read.table()`.

This appendix is adapted from Why R?.

Introduction to JAGS and BUGS language

Why learn JAGS?

So far, we have written Monte Carlo and Gibbs samplers directly in R. This is useful for understanding the mechanics of Bayesian computation, but it *becomes inconvenient when models are more complicated.*

JAGS stands for **Just Another Gibbs Sampler**. It is a program for fitting Bayesian hierarchical models using **MCMC**, especially Gibbs sampling and related methods.

JAGS uses the **BUGS language**, where BUGS stands for **B**ayesian inference **U**sing **G**ibbs **S**ampling.

The basic idea is simple:

1. we describe the statistical model in BUGS syntax;
2. we provide the observed data;
3. we choose which parameters to monitor;
4. JAGS runs the MCMC algorithm and returns posterior samples.

What BUGS language does

BUGS language is a **model specification language**. It is not a general programming language like R or Python.

Its role is to describe:

- the **sampling model**
- the **prior distributions**
- deterministic relationships among parameters
- optional **derived quantities**

For example:

$$Y_i \mid \theta \sim \text{Poisson}(\theta), \qquad \theta \sim \text{Gamma}(a, b)$$

Instead of writing the Gibbs sampler ourselves, we simply describe the model and JAGS handles the sampling.

History

- BUGS was developed in the 1990s by David Spiegelhalter and colleagues at the University of Cambridge.
- JAGS was created by Martyn Plummer in the early 2000s as a more flexible and open-source alternative to BUGS.
- Both BUGS and JAGS have been widely used in academia and industry for Bayesian data analysis, especially in fields like ecology, epidemiology, and social sciences.
- JAGS is designed to be compatible with BUGS models, so most BUGS code can be run in JAGS with little or no modification.
- JAGS is often used in conjunction with R through packages like `rjags` and `jagsUI`, which provide an interface for running JAGS models from R.
- The BUGS language has influenced the development of other probabilistic programming languages, such as **Stan**, **nimble** and **PyMC3**, which also allow users to specify Bayesian models in a high-level syntax.
- JAGS continues to be maintained and updated, with the latest version being 4.3.2 as of 2026.
- `coda` package is often used to analyze MCMC output from JAGS, providing tools for convergence diagnostics and posterior summaries.

Basic structure of a JAGS model

A JAGS model is written inside a `model {}` block.

```
model {
  for (i in 1:n) {
    y[i] ~ dpois(theta)
  }

  theta ~ dgamma(a, b)
}
```

The first part describes the likelihood, and the second part specifies the prior.

BUGS syntax rules

Stochastic nodes

Random variables are written using `~`.

```
theta ~ dgamma(a, b)
y[i] ~ dpois(theta)
```

Deterministic nodes

Deterministic relationships use `<-`.

```
sigma <- 1 / sqrt(tau)
mu[i] <- alpha + beta * x[i]
```

Loops

Repeated data are handled with loops.

```
for (i in 1:n) {
  y[i] ~ dnorm(mu, tau)
}
```

Comments

Use `#` for comments.

```
# prior distribution
theta ~ dnorm(0, 0.01)
```

> ⚠ JAGS syntax
>
> Pay attention to the JAGS syntax. Sometimes it may be different from what you expect! For instance, JAGS uses **precision** rather than variance for normal distributions `dnorm(mean, precision)`. Precision is
>
> $$\tau = \frac{1}{\sigma^2}.$$
>
> or in the lecture, we used $\tilde{\sigma}^2 = 1/\sigma^2$.
> In R code, we write

```
tau <- 1 / sigma^2
```

Table 9.1: Common probability distributions in JAGS (BUGS language)

Distribution	JAGS syntax
Normal	`dnorm(mu, tau)`
Poisson	`dpois(lambda)`
Binomial	`dbin(p, n)`
Bernoulli	`dbern(p)`
Gamma	`dgamma(a, b)`
Beta	`dbeta(a, b)`
Uniform	`dunif(a, b)`

Example: Beta–Binomial model

$$Y \mid \theta \sim \text{Binomial}(n, \theta), \qquad \theta \sim \text{Beta}(a, b)$$

JAGS model:

```
model {
  y ~ dbin(theta, n)
  theta ~ dbeta(a, b)
}
```

For multiple observations:

```
model {
  for (i in 1:n) {
    y[i] ~ dbern(theta)
  }
  theta ~ dbeta(a, b)
}
```

$$Y_i \mid \theta \sim \text{Poisson}(\theta), \qquad \theta \sim \text{Gamma}(a, b)$$

JAGS:

```
model {
  for (i in 1:n) {
    y[i] ~ dpois(theta)
  }

  theta ~ dgamma(a, b)
}
```

$$Y_i \mid \theta, \tau \sim \text{Normal}(\theta, \tau)$$

Priors

$$\theta \sim \text{Normal}(\mu_0, \tau_0), \qquad \tau \sim \text{Gamma}(a, b)$$

JAGS model:

```
model {
  for (i in 1:n) {
    y[i] ~ dnorm(theta, tau)
  }

  theta ~ dnorm(mu0, tau0)
  tau ~ dgamma(a, b)

  sigma <- 1 / sqrt(tau)
}
```

Data list in R

```
data_jags <- list(
  y = y,
  n = length(y),
  a = 2,
  b = 1
)
```

Parameters to monitor

```
params <- c("theta")
```

Example JAGS workflow

```
library(jagsUI)

fit <- jags(
  data = data_jags,
  parameters.to.save = params,
  model.file = textConnection(model_string),
  n.chains = 3,
  n.iter = 5000,
  n.burnin = 1000
)

print(fit)
```

Initial values

Optional but sometimes useful.

```
inits <- function() {
  list(theta = 1)
}
```

Posterior output

Typical outputs include

- posterior means
- credible intervals
- trace plots
- density plots
- effective sample size
- $\hat{R}$ convergence diagnostic

Table 9.2: Useful functions in the BUGS/JAGS language

Function	Meaning
`log(x)`	logarithm
`exp(x)`	exponential
`sqrt(x)`	square root
`pow(x, a)`	power
`step(x)`	indicator function

Example:

```
tau <- pow(sigma, -2)
```

Indexing

Sometimes, for each of the data y_i, you may want to have a different prior mean μ_i that depends on covariates x_i. This can be done as follows

```
for (i in 1:n) {
  y[i] ~ dnorm(mu[i], tau)
  mu[i] <- alpha + beta * x[i]
}
```

In this section we fit a simple Bayesian Poisson model in **JAGS**. Suppose

$$Y_1, \ldots, Y_n \mid \theta \sim \text{i.i.d. Poisson}(\theta),$$

with prior

$$\theta \sim \text{Gamma}(a, b),$$

where the Gamma distribution is parameterized by **shape** a and **rate** b.

We will:

1. specify the model in BUGS/JAGS language;
2. prepare the data in R;
3. run JAGS using `jagsUI`;
4. extract posterior samples;
5. compare the MCMC output with the exact posterior.

Data

We use the following toy dataset:

```
y <- c(3, 2, 4, 1, 3)
n <- length(y)

a <- 2
b <- 1
sum_y <- sum(y)

y
```

```
[1] 3 2 4 1 3
```

```
sum_y
```

```
[1] 13
```

Under the Poisson–Gamma model, the posterior distribution is available analytically:

$$\theta \mid y \sim \text{Gamma!}\left(a + \sum_{i=1}^{n} y_i, ; b + n\right).$$

So in this example,

$$\theta \mid y \sim \text{Gamma}(2 + 13, ; 1 + 5) = \text{Gamma}(15, 6).$$

This makes the example useful because we can compare the JAGS output to the exact answer.

Step 1: write the JAGS model

In BUGS language, the model is

```
model_string <- "
model {

  # likelihood
  for (i in 1:n) {
    y[i] ~ dpois(theta)
  }
```

```
  # prior
  theta ~ dgamma(a, b)

}
"
cat(model_string)
```

```
model {

  # likelihood
  for (i in 1:n) {
    y[i] ~ dpois(theta)
  }

  # prior
  theta ~ dgamma(a, b)

}
```

A few notes:

- `dpois(theta)` means Poisson with mean θ;
- `dgamma(a, b)` in JAGS uses shape and rate;
- `theta` is the parameter we want to estimate.

Step 2: prepare the data and monitored parameters

JAGS needs the data in a named list

```
data_jags <- list(
  y = y,
  n = n,
  a = a,
  b = b
)

params <- c("theta")
```

We will monitor only theta.

Step 3: choose initial values

Initial values are optional here, but we include them for completeness.

```r
inits <- function() {
  list(theta = 2)
}
```

Step 4: run JAGS

```r
library(jagsUI)

set.seed(8310)

fit <- jags(
  data = data_jags,
  inits = inits,
  parameters.to.save = params,
  model.file = textConnection(model_string),
  n.chains = 3,
  n.iter = 5000,
  n.burnin = 1000,
  n.thin = 2
)
```

```
Processing function input.......

Done.

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 5
   Unobserved stochastic nodes: 1
   Total graph size: 9

Initializing model

Adaptive phase.....
Adaptive phase complete


 Burn-in phase, 1000 iterations x 3 chains
```

```
Sampling from joint posterior, 4000 iterations x 3 chains


Calculating statistics.......

Done.
```

Now print the summary:

```
print(fit)
```

```
JAGS output for model '4', generated by jagsUI.
Estimates based on 3 chains of 5000 iterations,
adaptation = 100 iterations (sufficient),
burn-in = 1000 iterations and thin rate = 2,
yielding 6000 total samples from the joint posterior.
MCMC ran for 0.001 minutes at time 2026-03-15 15:42:58.285499.

           mean    sd   2.5%    50%  97.5% overlap0 f Rhat n.eff
theta     2.491 0.642  1.411  2.435  3.902    FALSE 1    1  6000
deviance 16.957 1.247 16.067 16.481 20.584    FALSE 1    1  6000

Successful convergence based on Rhat values (all < 1.1).
Rhat is the potential scale reduction factor (at convergence, Rhat=1).
For each parameter, n.eff is a crude measure of effective sample size.

overlap0 checks if 0 falls in the parameter's 95% credible interval.
f is the proportion of the posterior with the same sign as the mean;
i.e., our confidence that the parameter is positive or negative.

DIC info: (pD = var(deviance)/2)
pD = 0.8 and DIC = 17.734
DIC is an estimate of expected predictive error (lower is better).
```

This output includes posterior means, standard deviations, credible intervals, and convergence diagnostics.

Step 5: compare with the exact posterior

The exact posterior is

$$\theta \mid y \sim \text{Gamma}(15, 6).$$

So the exact posterior mean is

$$E(\theta \mid y) = \frac{15}{6} = 2.5.$$

The exact posterior variance is

$$\text{Var}(\theta \mid y) = \frac{15}{6^2} = \frac{15}{36}.$$

We can compute the exact summaries in R:

```
shape_post <- a + sum_y
rate_post  <- b + n

exact_mean <- shape_post / rate_post
exact_var  <- shape_post / rate_post^2
exact_ci   <- qgamma(c(0.025, 0.975), shape = shape_post, rate = rate_post)

exact_mean
```

```
[1] 2.5
```

```
exact_var
```

```
[1] 0.4166667
```

```
exact_ci
```

```
[1] 1.399231 3.914937
```

The JAGS summary for theta should be very close to these values.

Step 6: extract posterior samples

The posterior samples are stored inside the fitted object.

```r
theta_post <- fit$samples[, , "theta"]
theta_post <- as.vector(theta_post)

head(theta_post)
```

```
[[1]]
Markov Chain Monte Carlo (MCMC) output:
Start = 1002
End = 1014
Thinning interval = 2
        theta deviance
[1,] 1.560872 18.94180
[2,] 2.548968 16.07122
[3,] 3.785898 18.15506
[4,] 2.465718 16.10206
[5,] 2.216374 16.38049
[6,] 2.061858 16.71422
[7,] 2.080451 16.66675

[[2]]
Markov Chain Monte Carlo (MCMC) output:
Start = 1002
End = 1014
Thinning interval = 2
        theta deviance
[1,] 2.597255 16.06616
[2,] 2.152139 16.50281
[3,] 2.207936 16.39529
[4,] 2.923875 16.25254
[5,] 2.964158 16.29960
[6,] 2.286347 16.27207
[7,] 3.593582 17.58737

[[3]]
Markov Chain Monte Carlo (MCMC) output:
Start = 1002
End = 1014
Thinning interval = 2
        theta deviance
[1,] 1.738204 17.91732
[2,] 2.491540 16.08941
[3,] 2.296215 16.25878
```

```
[4,] 1.960389 17.01161
[5,] 2.444164 16.11480
[6,] 3.173188 16.61816
[7,] 3.100759 16.49421

attr(,"class")
[1] "mcmc.list"
```

```r
length(theta_post)
```

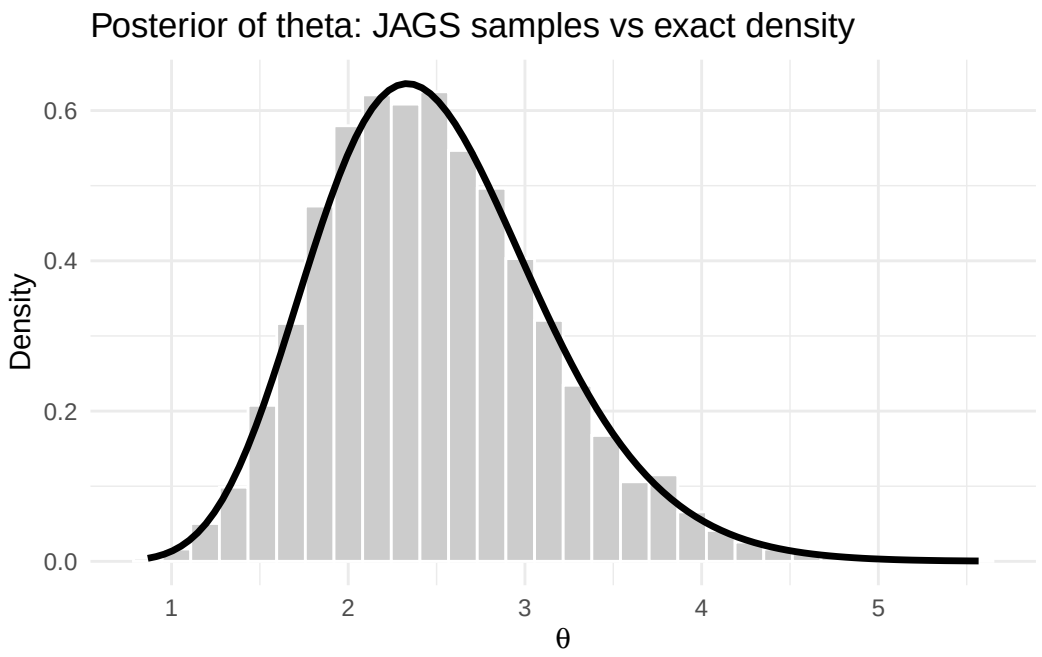
```
[1] 3
```

These are MCMC draws from the posterior distribution of θ.

Step 7: posterior histogram and exact density

Now compare the JAGS samples to the exact Gamma posterior density.

```r
library(ggplot2)
theta_post <- as.numeric(as.matrix(fit$samples)[,"theta"])

theta_df <- data.frame(theta = theta_post)
ggplot(theta_df, aes(x = theta)) +
  geom_histogram(aes(y = after_stat(density)),
                 bins = 30,
                 fill = "grey80",
                 color = "white") +
  stat_function(
    fun = dgamma,
    args = list(shape = shape_post, rate = rate_post),
    linewidth = 1.2,
    color = "black"
  ) +
  labs(
    title = "Posterior of theta: JAGS samples vs exact density",
    x = expression(theta),
    y = "Density"
  ) +
  theme_minimal()
```



The histogram of the JAGS samples should align closely with the exact posterior density.

Step 8: posterior probability of an event

Suppose we want to estimate

$$\Pr(\theta < 2.8 \mid y).$$

From the JAGS samples:

```
mean(theta_post < 2.8)
```

```
[1] 0.7071667
```

From the exact posterior:

```
pgamma(2.8, shape = shape_post, rate = rate_post)
```

```
[1] 0.702895
```

These two values should be close.

Step 9: posterior predictive simulation

Suppose $\tilde{Y}$ is a future observation from the same population. The posterior predictive distribution is

$$p(\tilde{y} \mid y) \int p(\tilde{y} \mid \theta), p(\theta \mid y), d\theta.$$

Using the JAGS samples, we can simulate from the posterior predictive distribution:

```
set.seed(8310)
y_tilde <- rpois(length(theta_post), lambda = theta_post)

head(y_tilde)
```

```
[1] 1 3 2 2 1 1
```

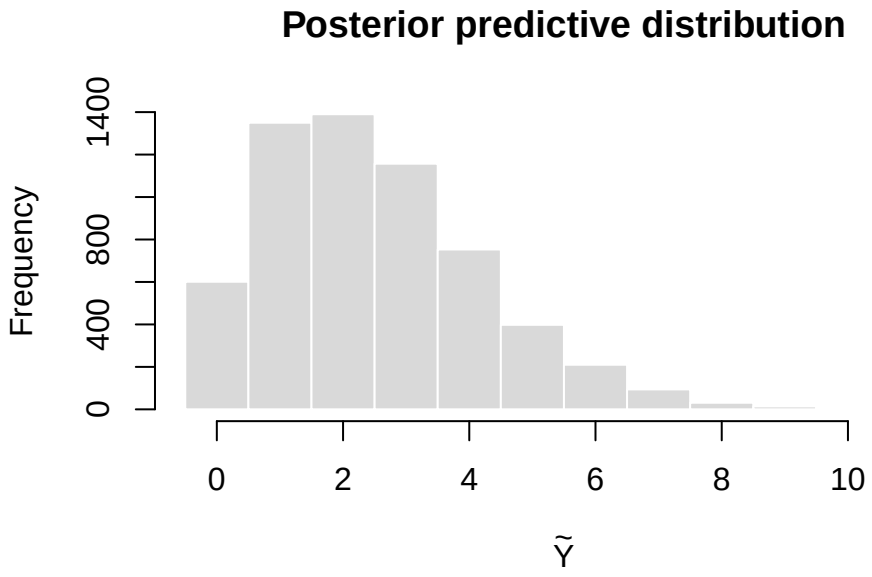
```
mean(y_tilde)
```

```
[1] 2.486
```

```
var(y_tilde)
```

```
[1] 2.957964
```

```
hist(y_tilde,
     breaks = seq(-0.5, max(y_tilde) + 0.5, by = 1),
     col = "grey85",
     border = "white",
     main = "Posterior predictive distribution",
     xlab = expression(tilde(Y)))
```



This histogram represents the distribution of a future observation $\tilde{Y}$ given the observed data y.

Interpretation

This example illustrates the full JAGS workflow:

- specify a Bayesian model in BUGS language;
- provide the data and priors;
- run MCMC in JAGS;
- summarize the posterior;
- compare with an exact answer when available.

Because the Poisson–Gamma model is conjugate, this example is especially useful for learning. The JAGS output should agree closely with the exact posterior

$$\text{Gamma}(15, 6).$$

Full code in one chunk

```
library(jagsUI)
library(coda)
```

```
Attaching package: 'coda'
```

```
The following object is masked from 'package:jagsUI':

    traceplot
```

```r
library(ggplot2)

# ================================
# Data
# ================================
y <- c(3, 2, 4, 1, 3)
n <- length(y)
a <- 2
b <- 1
sum_y <- sum(y)

# ================================
# JAGS model
# ================================
model_string <- "
model {

  for (i in 1:n) {
    y[i] ~ dpois(theta)
  }

  theta ~ dgamma(a, b)

}
"

# ================================
# Data list for JAGS
# ================================
data_jags <- list(
  y = y,
  n = n,
  a = a,
  b = b
)

params <- c("theta")
```

```r
inits <- function() {
  list(theta = 2)
}

# ================================
# Run JAGS
# ================================
set.seed(8310)

fit <- jags(
  data = data_jags,
  inits = inits,
  parameters.to.save = params,
  model.file = textConnection(model_string),
  n.chains = 3,
  n.iter = 5000,
  n.burnin = 1000,
  n.thin = 2
)
```

```
Processing function input.......

Done.

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 5
   Unobserved stochastic nodes: 1
   Total graph size: 9

Initializing model

Adaptive phase.....
Adaptive phase complete


 Burn-in phase, 1000 iterations x 3 chains


Sampling from joint posterior, 4000 iterations x 3 chains
```

```
Calculating statistics.......

Done.
```

```
print(fit)
```

```
JAGS output for model '4', generated by jagsUI.
Estimates based on 3 chains of 5000 iterations,
adaptation = 100 iterations (sufficient),
burn-in = 1000 iterations and thin rate = 2,
yielding 6000 total samples from the joint posterior.
MCMC ran for 0 minutes at time 2026-03-15 15:44:45.704652.

           mean    sd   2.5%    50%  97.5% overlap0 f Rhat n.eff
theta     2.491 0.642  1.411  2.435  3.902    FALSE 1    1  6000
deviance 16.957 1.247 16.067 16.481 20.584    FALSE 1    1  6000

Successful convergence based on Rhat values (all < 1.1).
Rhat is the potential scale reduction factor (at convergence, Rhat=1).
For each parameter, n.eff is a crude measure of effective sample size.

overlap0 checks if 0 falls in the parameter's 95% credible interval.
f is the proportion of the posterior with the same sign as the mean;
i.e., our confidence that the parameter is positive or negative.

DIC info: (pD = var(deviance)/2)
pD = 0.8 and DIC = 17.734
DIC is an estimate of expected predictive error (lower is better).
```

```
# ================================
# Exact posterior
# ================================
shape_post <- a + sum_y
rate_post  <- b + n

exact_mean <- shape_post / rate_post
exact_var  <- shape_post / rate_post^2
exact_ci   <- qgamma(c(0.025,0.975), shape = shape_post, rate = rate_post)

exact_mean
```

```
[1] 2.5
```

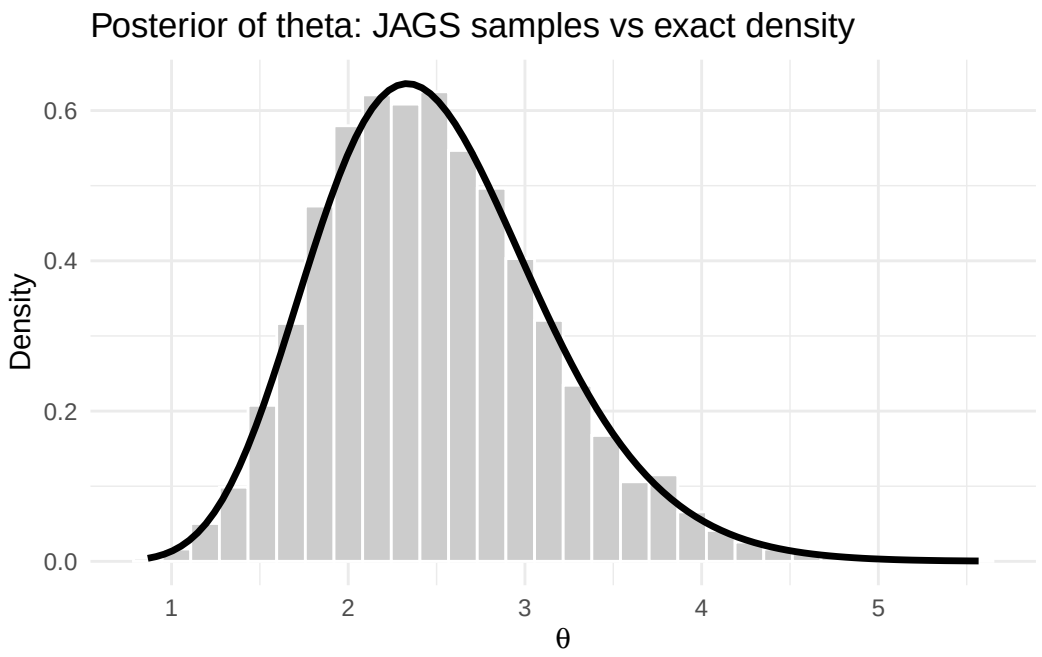
```
exact_var
```

```
[1] 0.4166667
```

```
exact_ci
```

```
[1] 1.399231 3.914937
```

```
# ================================
# Extract posterior samples
# ================================
theta_post <- as.numeric(as.matrix(fit$samples)[,"theta"])

theta_df <- data.frame(theta = theta_post)

# ================================
# Posterior distribution plot
# ================================
ggplot(theta_df, aes(x = theta)) +
  geom_histogram(aes(y = after_stat(density)),
                 bins = 30,
                 fill = "grey80",
                 color = "white") +
  stat_function(
    fun = dgamma,
    args = list(shape = shape_post, rate = rate_post),
    linewidth = 1.2,
    color = "black"
  ) +
  labs(
    title = "Posterior of theta: JAGS samples vs exact density",
    x = expression(theta),
    y = "Density"
  ) +
  theme_minimal()
```




```
# ================================
# Posterior probability example
# ================================
mean(theta_post < 2.8)
```

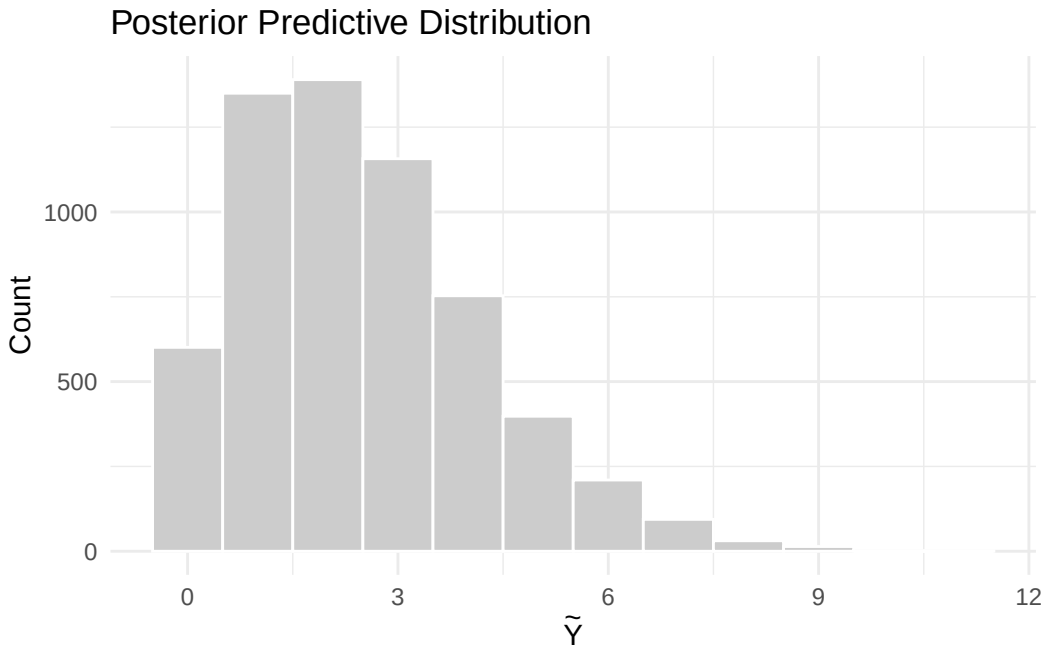
```
[1] 0.7071667
```

```
pgamma(2.8, shape = shape_post, rate = rate_post)
```

```
[1] 0.702895
```

```
# ================================
# Posterior predictive simulation
# ================================
set.seed(8310)

y_tilde <- rpois(length(theta_post), lambda = theta_post)

pred_df <- data.frame(y_tilde = y_tilde)

# ================================
```

```r
# Posterior predictive plot
# ================================
ggplot(pred_df, aes(x = y_tilde)) +
  geom_histogram(
    binwidth = 1,
    fill = "grey80",
    color = "white",
    boundary = -0.5
  ) +
  labs(
    title = "Posterior Predictive Distribution",
    x = expression(tilde(Y)),
    y = "Count"
  ) +
  theme_minimal()
```



Summary and Take home message

- JAGS performs Bayesian inference using MCMC.
- Models are written in the BUGS language.
- ~ denotes a distribution.

- `<-` denotes deterministic relationships.
- Normal distributions use **precision** instead of variance.
- JAGS separates **model**, **data**, and **parameters**.

> ⚠ Common (beginner) mistakes
>
> 1. Forgetting that normal uses **precision**
> 2. Mixing deterministic `<-` and stochastic `~`
> 3. Assuming all R syntax works inside JAGS
> 4. Forgetting to pass constants in the data list

Reference:

- JAGS User Manual
- jagsUI package documentation

References

Hoff, Peter D. 2009. *A First Course in Bayesian Statistical Methods*. Springer. https://sites.math.rutgers.edu/~zeilberg/EM20/Hoff.pdf.