LUISS

Degree Program in Data Science and Management

Course of Advanced Statistics

BAYESIAN HIERARCHICAL MODELS

Prof. Catalano Marta

SUPERVISOR

Prof. Sinaimeri Blerina

CO-SUPERVISOR

Romano Martina 781201

CANDIDATE

Academic Year 2024/2025

Contents

Introduction

Understanding and modeling uncertainty is a central concern in modern statistics. Among the available paradigms, the Bayesian approach offers a coherent and flexible framework that naturally incorporates prior information and yields full posterior distributions over unknown parameters. This probabilistic perspective is especially powerful when dealing with complex data structures, such as those involving group-level heterogeneity or nested designs.

This thesis investigates the use of *Bayesian hierarchical models*—also known as multilevel models—with a focus on their application to real-world data. These models are particularly well suited to settings where data are organized in groups, and where the number of observations per group is limited. By allowing for *partial pooling* of information, hierarchical models provide stable individual-level estimates while leveraging global patterns, offering a principled trade-off between overfitting and underfitting.

The methodological foundation of the thesis builds on key concepts from Bayesian inference, including:

- the formulation of posterior distributions via Bayes' theorem,
- the construction of priors informed by historical data or theoretical assumptions,
- and the use of Markov Chain Monte Carlo (MCMC) methods, particularly the Gibbs sampler, to approximate posterior distributions.

Objectives and Contributions. The main contributions of this thesis are both theoretical and applied. On the theoretical side, Chapter 1 provides a detailed overview of hierarchical Bayesian modeling, with emphasis on the Normal-Normal and Beta-Binomial cases, the role of exchangeability, and the implementation of the Gibbs sampler. Convergence diagnostics such as $\hat{R}$ and effective sample size are also discussed and applied to a classical case study: the Eight Schools example.

On the applied side, Chapters 2 and 3 present two case studies based on hierarchical logistic regression. The first replicates a marketing experiment by Allenby et al. (2005) on consumer preferences for credit card attributes using simulated data. The second extends the approach to a real-world e-commerce

dataset, where user-level preferences for products are inferred from transaction records.

This second case study represents the most original part of the thesis. Using a public retail dataset, we construct binary choice tasks from transaction data and implement a Metropolis-within-Gibbs sampler to estimate individual-level part-worth utilities. Despite the lack of explicit choice sets, the model was able to uncover interpretable latent preferences, demonstrating the utility of Bayesian hierarchical modeling in sparse and noisy real-world environments.

Key Findings. The empirical results confirm that:

- Partial pooling via hierarchical priors improves the stability of individual estimates, especially when data are limited.
- The Bayesian approach yields interpretable posterior distributions for both individual- and population-level parameters.
- In the e-commerce case study, the estimated covariance structure revealed meaningful patterns of heterogeneity in price sensitivity and product preferences.

The work also illustrates how Bayesian methods can be applied beyond textbook examples to complex, unstructured data, providing both methodological insight and practical value for applications in marketing analytics and customer behavior modeling.

Structure of the Thesis. The remainder of the thesis is organized as follows:

- Chapter 1 introduces the theoretical framework of Bayesian hierarchical models, including model formulation, sampling algorithms, and convergence checks.
- Chapter 2 applies these techniques to a classical marketing study using simulated data.
- Chapter 3 extends the hierarchical logit model to a real-world e-commerce dataset, with full implementation and analysis.
- Appendices provide code listings, details of the MCMC algorithms, and preprocessing pipelines.

To formally ground the approach adopted throughout this thesis, the following section outlines the foundational principles of Bayesian statistics and compares them to the frequentist paradigm.

Bayesian statistics provides a coherent framework for learning from data by explicitly incorporating prior beliefs and updating them using observed evidence. At the heart of the Bayesian approach lies Bayes' theorem, which expresses the posterior distribution of unknown parameters as proportional to the product of the likelihood and the prior distribution:

$$p(\theta|y) \propto p(y|\theta) \cdot p(\theta),$$

where:

- $p(\theta)$ is the prior distribution, reflecting initial beliefs about the parameter θ,
- $p(y|\theta)$ is the likelihood, describing how the observed data y are generated given θ,
- $p(\theta|y)$ is the posterior distribution, combining prior information and observed data.

This formulation allows for continuous updating as more data become available, making it especially attractive for problems involving uncertainty, small samples, or hierarchical structures.

Frequentist vs. Bayesian Perspectives. The frequentist framework treats parameters as fixed but unknown quantities and bases inference on repeated sampling properties. In contrast, the Bayesian framework treats parameters as random variables and uses probability to express uncertainty about them.

Aspect	**Frequentist**	**Bayesian**
Nature of parameters	Fixed unknown constants	Random variables
Uncertainty	Derives from the randomness of the data under repeated sampling	Derives from the beliefs on the parameter
Inferences	Confidence intervals, p-values	Posterior distributions
Prior knowledge	Not incorporated	Explicitly modeled via priors
Interpretation of interval	Long-run frequency	Degree of belief

Table 1: Key differences between frequentist and Bayesian inference.

Posterior Interpretation. In Bayesian inference, the posterior distribution provides a complete probabilistic description of the uncertainty about parameters given the data. For instance, a 95% credible interval for θ means that, given the model and the observed data, there is a 95% probability that θ lies within the interval—an interpretation more intuitive than its frequentist counterpart.

Why Bayesian Methods for Hierarchical Models? Bayesian methods are particularly well-suited for hierarchical modeling because they allow uncertainty to be propagated naturally through multiple levels of parameters. Unlike ad hoc frequentist techniques (e.g., random-effects models estimated via point estimators), Bayesian approaches yield full posterior distributions over all unknowns, leading to richer and more robust inferences.

For these reasons, the rest of this thesis adopts the Bayesian framework, particularly in the context of hierarchical models, as developed in Chapters 5 and 11 of Gelman et al. (2013).

Chapter 1

Foundations of Bayesian Hierarchical Models

1.1 Introduction to Hierarchical Models

In modern statistical applications, it is increasingly common to deal with data that are grouped or nested in some way. Whether analyzing students within schools, patients within hospitals, or repeated experiments within different settings, such scenarios naturally give rise to multiple related parameters. A hierarchical model provides a principled Bayesian approach to model such settings by capturing dependencies across different units while preserving the individuality of each group.

Hierarchical Bayesian models—also referred to as multilevel models—structure parameters across layers: observable data are modeled conditionally on latent variables (group-level parameters), which themselves are assigned prior distributions governed by higher-level hyperparameters. This structure allows for *partial pooling* of information across groups, a key feature that avoids both underfitting (due to complete pooling) and overfitting (due to complete separation). As Gelman et al. (2013) emphasize, this strategy balances flexibility and regularization, particularly when some groups have sparse or noisy data.

A motivating example comes from toxicology. Consider a study measuring tumor incidence in female rats from multiple control groups. Each group has a different observed tumor rate, but all belong to the same rat strain and are subjected to similar experimental settings. Modeling the tumor probability in each group independently would ignore useful information shared across experiments; on the other hand, assuming the same rate across all groups is likely too restrictive. A hierarchical model treats the group-specific tumor probabilities as random variables drawn from a common population distribution, thus enabling shrinkage of extreme estimates toward the global average and improving inference robustness.

A further rationale for hierarchical modeling lies in its ability to handle

multiparameter inference efficiently. In contrast to flat models, which often require explicit assumptions or tuning to avoid overfitting, hierarchical models regularize estimates automatically through the joint posterior structure. This is particularly helpful in settings with limited sample sizes within groups, where individual-level estimates are unreliable but information borrowing across groups leads to improved predictive accuracy.

Perhaps most importantly, hierarchical models reflect how knowledge is often organized in real-world reasoning. As noted by Gelman et al. (2013), scientific understanding often proceeds by layering uncertainties: observed measurements depend on unknown quantities, which in turn depend on higher-level factors. Bayesian hierarchical models mirror this structure, making them not only flexible but also conceptually aligned with how uncertainty and evidence propagate in complex systems.

Moreover, hierarchical modeling is more than just a statistical technique—it also informs computational strategies. Inference in hierarchical models can be analytically tractable in simple conjugate cases (e.g., beta-binomial or normal-normal setups), but in most applications, it requires sampling-based methods such as Markov Chain Monte Carlo (MCMC), especially Gibbs sampling and the Metropolis-Hastings algorithm. These computational tools, discussed later in this chapter, make hierarchical modeling broadly applicable to real-world data.

In summary, Bayesian hierarchical models serve both as a method for principled regularization and a framework for representing multilevel uncertainty. They are foundational for modern applied Bayesian statistics, with applications ranging from education and epidemiology to neuroscience and machine learning.

1.2 Building Hierarchical Models

Bayesian hierarchical models are designed to capture structure in data that is grouped or nested. Building these models involves assigning probability distributions at multiple levels, starting from observed data and moving up through parameters and hyperparameters. This section outlines the foundational ideas and practical motivations for constructing hierarchical models, with examples and theoretical justification.

At the core of hierarchical modeling lies the assumption that group-level parameters (such as treatment effects in different hospitals or tumor rates in different rat groups) are not completely independent but drawn from a common population distribution. This induces a natural pooling of information across groups and reflects the belief that while units may differ, they also share commonalities.

1.2.1 Using Historical Data to Build Priors

One of the first steps in hierarchical modeling is the construction of prior distributions informed by historical or related data. In many practical settings,

analysts are not working in isolation: previous studies, experiments, or measurements are available and can be used to inform the prior beliefs about group-specific parameters.

Rather than fixing priors arbitrarily, a more principled approach is to use the empirical distribution of past estimates to determine the hyperparameters of the prior distribution. For example, if we are modeling a group-specific probability θ_j with binomial data, and we believe the θ_j values arise from a common population distribution, it is natural to assume $\theta_j \sim \text{Beta}(\alpha, \beta)$. The parameters α and β can then be estimated based on the observed variation across groups in the historical data.

This leads to a hierarchical model structure such as:

$$\theta_j \overset{\text{iid}}{\sim} \text{Beta}(\alpha, \beta)$$
$$y_j \overset{\text{ind}}{\sim} \text{Bin}(n_j, \theta_j)$$

This setup allows for *information borrowing* across units: current estimates for a new group are informed not only by its own data, but also by the broader patterns learned from previous groups. In turn, the posterior distribution for a new parameter (e.g., θ_{J+1}) incorporates both the observed data and the uncertainty encoded in the shared prior.

A key feature of this modeling approach is the phenomenon of *shrinkage*, where extreme or noisy estimates are pulled toward the global mean of the prior distribution. This leads to more stable and realistic inferences, especially in contexts with limited sample sizes or high variability across groups.

1.2.2 The Concept of Exchangeability

The principle of exchangeability is foundational to hierarchical modeling. It assumes that in the absence of distinguishing information, the ordering or labeling of groups does not carry meaning. Formally, a set of parameters $\theta_1, \ldots, \theta_J$ are exchangeable if the joint prior distribution is invariant to permutations:

$$p(\theta_1, \ldots, \theta_J) = p(\theta_{\pi(1)}, \ldots, \theta_{\pi(J)})$$

for any permutation π. While this exchangeability assumption does not *a priori* imply that the θ_j are conditionally independent and identically distributed, a classical result known as **de Finetti's theorem** states that any infinite sequence of exchangeable random variables can be represented as a mixture of i.i.d. random variables. This justifies modeling θ_j as independent draws from a common distribution $p(\theta_j \mid \phi)$, where ϕ is a hyperparameter vector with its own prior $p(\phi)$. Thus, the prior over θ is:

$$p(\theta) = \int \prod_{j=1}^{J} p(\theta_j|\phi)\, p(\phi)\, d\phi$$

This structure naturally supports **partial pooling**, as it captures both within-group variability (via the likelihood) and across-group similarity (via the shared prior). Exchangeability enables hierarchical models to handle multi-experiment or multi-region datasets with varying sample sizes and uncertain effects.

1.2.3 Fully Bayesian Approach

A fully Bayesian hierarchical model treats hyperparameters as random variables with their own prior distributions—known as hyperpriors. This completes the probabilistic structure and enables inference over the entire hierarchy. For instance:

$$\theta_j \overset{\text{iid}}{\sim} \text{Beta}(\alpha, \beta)$$
$$y_j \overset{\text{ind}}{\sim} \text{Bin}(n_j, \theta_j)$$

$$(\alpha, \beta) \sim p(\alpha, \beta)$$

where, for example, one may choose a prior such as $p(\alpha, \beta) \propto (\alpha + \beta)^{-5/2}$, which is considered *weakly informative*. This means that the prior provides gentle regularization to avoid extreme parameter values, while still allowing the data to dominate the inference when sufficiently informative.

This setup ensures that uncertainty in the hyperparameters is fully propagated through the posterior. Inference is performed on the joint posterior $p(\theta, \alpha, \beta|y)$, often using simulation methods such as Gibbs sampling (as detailed later in the chapter).

This fully Bayesian perspective avoids the drawbacks of empirical Bayes approaches, such as ignoring uncertainty in hyperparameter estimates or "double-counting" the data. It also aligns more closely with Bayesian philosophy, where all unknowns are treated probabilistically.

1.3 Bayesian Inference in Hierarchical Models

Bayesian inference in hierarchical models revolves around estimating both the lower-level parameters (such as group-specific means or probabilities) and the

higher-level hyperparameters that govern their distribution. This process integrates prior knowledge with observed data across multiple levels, yielding robust estimates and coherent uncertainty quantification.

A central feature of this approach is the joint modeling of all parameters, from the group-level effects θ_j to the population-level hyperparameters ϕ. Rather than fixing the hyperparameters (as done in empirical Bayes), the Bayesian approach treats them as unknowns and assigns prior distributions, enabling full posterior inference.

The joint posterior distribution under a hierarchical model typically takes the form:

$$p(\phi, \theta|y) \propto p(\phi) \cdot \prod_{j=1}^{J} p(\theta_j|\phi) \cdot p(y_j|\theta_j),$$

where y_j represents the observed data in group j, θ_j the group-specific parameter, and ϕ the hyperparameters that define the population distribution.

1.3.1 Models with Conjugate Families (e.g., Beta-Binomial, Normal-Normal)

When both the likelihood and prior belong to conjugate families, analytical expressions for the posterior distribution can be derived. For example, in a Beta-Binomial model:

$$\theta_j \overset{\text{iid}}{\sim} \text{Beta}(\alpha, \beta)$$

$$\text{y}_j \overset{\text{ind}}{\sim} \text{Binomial}(n_j, \theta_j)$$

Given this setup, the posterior distribution for each parameter θ_j — conditional on the corresponding individual observation y_j — is also a Beta distribution:

$$\theta_j \mid y_j \sim \text{Beta}(\alpha + y_j, \beta + n_j - y_j).$$

Similarly, in a hierarchical normal model with known observation variance σ_j^2, the group means θ_j are modeled as:

$$y_j|\theta_j \sim \mathcal{N}(\theta_j, \sigma_j^2), \quad \theta_j \sim \mathcal{N}(\mu, \tau^2),$$

where μ and τ^2 are the hyperparameters. The posterior distribution of θ_j given μ and τ^2 is also normal, and takes the form:

$$\theta_j|y_j, \mu, \tau^2 \sim \mathcal{N}\left(\frac{\mu/\tau^2 + y_j/\sigma_j^2}{1/\tau^2 + 1/\sigma_j^2},\ \left(\frac{1}{\tau^2} + \frac{1}{\sigma_j^2}\right)^{-1}\right).$$

These conjugate structures simplify computation and offer interpretability in terms of information pooling and precision-weighted averaging.

1.3.2 Complete Pooling, No Pooling, and Partial Pooling

Bayesian hierarchical models naturally accommodate different degrees of pooling. The three canonical approaches are:

- **Complete pooling**: Assumes all θ_j are equal. The model estimates one shared parameter for all groups, ignoring group-specific variability.
- **No pooling**: Treats each group as completely separate, estimating each θ_j independently.
- **Partial pooling**: The hierarchical Bayesian approach. Each θ_j is estimated while accounting for both the group-specific data and the population-level distribution.

The partial pooling approach leads to **shrinkage**, where the posterior means of group-level parameters are pulled toward the population mean μ, with the extent of shrinkage depending on the relative magnitudes of the group sample size and the between-group variance τ^2.

This balance is automatic in Bayesian inference: the more data a group has, the less its estimate is shrunk. Conversely, small or noisy groups benefit from the information shared across the population, yielding more stable estimates.

1.4 Computational Methods: Gibbs Sampling

Bayesian hierarchical models often involve complex posterior distributions that cannot be evaluated analytically. In such cases, simulation-based approaches are essential for performing inference. One of the most effective and widely used methods is the Gibbs sampler, a Markov Chain Monte Carlo (MCMC) technique tailored for models with conditionally tractable structure. The Gibbs sampler enables sampling from a high-dimensional joint posterior distribution by iteratively sampling from the full conditional distributions of each parameter.

The Gibbs sampler is based on the principle of alternating conditional sampling. Suppose the parameter vector θ is partitioned into d components, $\theta = (\theta_1, \ldots, \theta_d)$. Each iteration of the algorithm cycles through the components, sampling one parameter at a time from its conditional distribution given the current values of the others:

$$\theta_j^{(t)} \sim p(\theta_j \mid \theta_{-j}^{(t)}, y),$$

where $\theta_{-j}^{(t)}$ denotes the current values of all parameters except θ_j. This process constructs a Markov chain whose stationary distribution is the target posterior $p(\theta \mid y)$, under mild regularity conditions.

The main advantage of the Gibbs sampler is that it often simplifies implementation when the model has a conjugate structure, as in many hierarchical models. For example, in a hierarchical normal model, the conditional distributions for group-level parameters and hyperparameters often fall into known

families such as the normal or inverse-chi-squared distributions. In such cases, each sampling step can be performed exactly and efficiently, making the Gibbs sampler particularly appealing.

Despite its conceptual simplicity, the Gibbs sampler has profound theoretical underpinnings. It can be viewed as a special case of the more general Metropolis-Hastings algorithm, where the proposal distribution is always accepted because it is the full conditional distribution. This means that the acceptance rate is 100%, ensuring efficient movement through parameter space—provided that the conditional distributions are not too strongly correlated.

However, there are also challenges. The convergence of the Gibbs sampler may be slow if the parameters are highly correlated, leading to poor mixing and long autocorrelation times. In such cases, reparameterization or hybrid approaches (e.g., Metropolis-within-Gibbs) can be used to improve performance. Furthermore, convergence diagnostics are essential to ensure that the sampler has adequately explored the posterior distribution. Techniques such as the potential scale reduction factor ($\hat{R}$) and the effective sample size (n_{eff}) are commonly used for this purpose and will be discussed later in this chapter.

The Gibbs sampler has been successfully applied to a wide variety of models, including those with latent variables, missing data, and hierarchical structures. Its success lies in its general applicability, ease of implementation, and strong theoretical guarantees. In the context of Bayesian hierarchical modeling, it provides a natural and flexible computational tool to perform full posterior inference when direct analytical solutions are infeasible.

1.4.1 Basic Idea of the Gibbs Sampler

The Gibbs sampler is a specific type of Markov Chain Monte Carlo (MCMC) method that relies on iteratively sampling from the full conditional distributions of a joint posterior. Its power lies in its simplicity: rather than requiring a joint sampling distribution over all parameters simultaneously, the Gibbs algorithm samples each parameter conditional on the current values of all the others.

Assume a parameter vector $\theta = (\theta_1, \theta_2, \ldots, \theta_d)$ with posterior distribution $p(\theta|y)$. If it is possible to compute the full conditional distributions $p(\theta_j|\theta_{-j}, y)$ for each component θ_j, where θ_{-j} denotes all components except θ_j, the Gibbs sampler proceeds as follows:

1. Initialize $\theta^{(0)} = (\theta_1^{(0)}, \ldots, \theta_d^{(0)})$.
2. For each iteration $t = 1, 2, \ldots$:
   - Sample $\theta_1^{(t)} \sim p(\theta_1 \mid \theta_2^{(t-1)}, \ldots, \theta_d^{(t-1)}, y)$
   - Sample $\theta_2^{(t)} \sim p(\theta_2 \mid \theta_1^{(t)}, \theta_3^{(t-1)}, \ldots, \theta_d^{(t-1)}, y)$
   - $\vdots$
   - Sample $\theta_d^{(t)} \sim p(\theta_d \mid \theta_1^{(t)}, \ldots, \theta_{d-1}^{(t)}, y)$

Each full iteration updates every parameter once, using the most recently sampled values. The resulting sequence of samples $(\theta^{(t)})_{t=1}^{T}$ forms a Markov chain whose stationary distribution is the desired posterior $p(\theta \mid y)$, assuming standard regularity conditions like irreducibility and aperiodicity.

The Gibbs sampler is particularly effective in models where conditional conjugacy applies, meaning the conditional distributions take a known form from which we can sample directly (e.g., Gaussian, Gamma, Beta). This makes it especially suitable for many hierarchical models.

A key strength of the Gibbs sampler is that it does not require a tuning parameter like a proposal distribution (as in Metropolis-Hastings). Additionally, all proposed moves are always accepted, improving sampling efficiency. However, its performance can degrade if the components of θ are strongly correlated, in which case the chain may mix slowly.

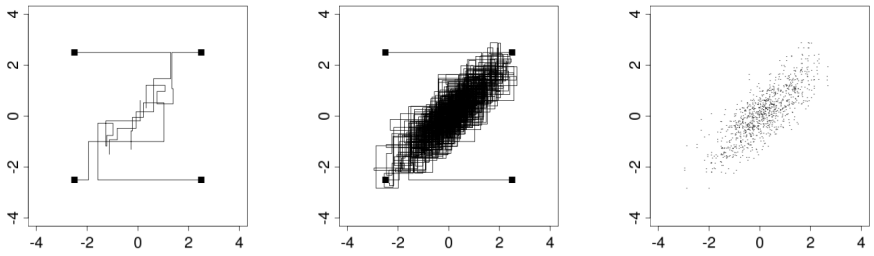

Figure 1.1: Gibbs sampling chains in a bivariate normal distribution. Left: first few iterations from overdispersed initial values. Center: sample accumulation after many steps. Right: final draws used for inference.

Figure 1.1 shows a visual example of this process applied to a bivariate normal distribution with correlated components. The three panels illustrate the early iterations from different starting points (left), the accumulated samples over time (center), and the final draws used for inference (right). The progression highlights how the Gibbs sampler explores the distribution iteratively along coordinate axes before achieving full mixing.

In summary, the Gibbs sampler is a foundational tool in Bayesian computation, offering a straightforward and often highly effective way to sample from complex posterior distributions when conditional distributions are available in closed form.

1.4.2 Advantages and Limitations

The Gibbs sampler offers several compelling advantages that have contributed to its widespread use in Bayesian computation, especially for hierarchical models. First and foremost, its conceptual simplicity and ease of implementation make it accessible for a broad class of models. When conditional distributions are known

and easy to sample from, the Gibbs algorithm can be applied directly without the need for complex tuning or acceptance-rejection steps. This is particularly beneficial in models where conjugacy ensures closed-form conditionals, allowing for efficient sampling of each parameter component.

Another key advantage lies in the sampler's deterministic structure. Unlike Metropolis-Hastings algorithms, where proposals may be rejected, every step in the Gibbs sampler is accepted by construction. This ensures that the Markov chain progresses smoothly and avoids wasted iterations. Furthermore, the Gibbs sampler's component-wise updates can provide fine-grained control over the sampling process, especially when different parameters vary on different scales.

Additionally, the Gibbs sampler is modular. It can be flexibly combined with other algorithms, such as Metropolis steps for non-conjugate parameters, enabling hybrid strategies (Metropolis-within-Gibbs) that extend its applicability beyond analytically tractable cases. This hybridization is particularly valuable in modern Bayesian workflows where models often involve both simple and complex components.

However, the Gibbs sampler also presents important limitations. A major concern is the potential for slow convergence and poor mixing in the presence of strong posterior dependencies among parameters. When parameters are highly correlated, successive updates based on one-dimensional conditional distributions may lead to a random-walk behavior, requiring many iterations to explore the target distribution effectively.

In high-dimensional spaces or models with tightly coupled parameters, this slow exploration results in long autocorrelation times and a reduced effective sample size. Reparameterization strategies can sometimes help mitigate this issue, but they often require model-specific insight.

Another drawback is that the Gibbs sampler requires full conditional distributions to be explicitly known and directly samplable. This restricts its application to models with a specific structure. For more general models, other MCMC methods such as Hamiltonian Monte Carlo or adaptive Metropolis algorithms are often more appropriate, despite their increased computational complexity.

Finally, while convergence diagnostics can indicate whether the Gibbs sampler has approximately reached the stationary distribution, they cannot guarantee complete exploration of the posterior—especially in multimodal settings. The sampler may become trapped in a local mode if transitions between modes are unlikely under the component-wise updating scheme.

The Gibbs sampler is a powerful and elegant tool for Bayesian inference in structured models, particularly when conjugate conditionals are available. Yet, its performance is highly sensitive to the geometry of the posterior distribution, and practitioners must remain vigilant to its potential inefficiencies in complex or correlated parameter spaces.

1.4.3 Implementation in the Hierarchical Normal Model

The hierarchical normal model is a canonical example for applying Gibbs sampling, due to its conditional conjugacy and wide applicability in grouped data

analysis. It assumes observed data y_{ij}, where $i = 1, \ldots, n_j$ indexes observations within group $j = 1, \ldots, J$. Conditional on the group-specific parameters $\theta_1, \ldots, \theta_J$, the observations are modeled as:

$$y_{ij} \overset{\text{iid}}{\sim} \mathcal{N}(\theta_j, \sigma^2),$$

where each group-specific mean θ_j is drawn from a common population distribution:

$$\theta_j \sim \mathcal{N}(\mu, \tau^2).$$

Hyperpriors are then placed on the population parameters μ and τ^2. In a fully Bayesian framework, σ^2 may also be unknown and assigned a prior, typically inverse-chi-squared or a weakly informative form.

The structure of this model leads to a sequence of full conditional distributions that can be directly sampled in a Gibbs sampler:

- For each group mean θ_j, the conditional posterior is:

$$\theta_j \mid \mu, \tau^2, \sigma^2, y \sim \mathcal{N}\left(\frac{\mu/\tau^2 + n_j \bar{y}_j/\sigma^2}{1/\tau^2 + n_j/\sigma^2},\ \left(\frac{1}{\tau^2} + \frac{n_j}{\sigma^2}\right)^{-1}\right),$$

  where $\bar{y}_j$ is the mean of observations in group j.

  Proof: We derive this from conjugacy between the normal prior and likelihood. The prior is $\theta_j \sim \mathcal{N}(\mu, \tau^2)$ and the likelihood for group j is:

$$y_{ij} \sim \mathcal{N}(\theta_j, \sigma^2), \quad i = 1, \ldots, n_j.$$

  The likelihood contributes a term:

$$\prod_{i=1}^{n_j} \mathcal{N}(y_{ij} \mid \theta_j, \sigma^2) \propto \exp\left\{-\frac{n_j}{2\sigma^2}(\bar{y}_j - \theta_j)^2\right\},$$

  and the prior contributes:

$$\mathcal{N}(\theta_j \mid \mu, \tau^2) \propto \exp\left\{-\frac{1}{2\tau^2}(\theta_j - \mu)^2\right\}.$$

  Combining and completing the square gives a normal posterior with the specified mean and variance.

- The population mean μ has a conditional posterior:

$$\mu \mid \theta, \tau^2 \sim \mathcal{N}\left(\frac{1}{J}\sum_{j=1}^{J} \theta_j,\ \frac{\tau^2}{J}\right).$$

  Proof: Given $\theta_j \sim \mathcal{N}(\mu, \tau^2)$ independently, and assuming a flat prior on μ, the likelihood becomes:

$$\prod_{j=1}^{J} \mathcal{N}(\theta_j \mid \mu, \tau^2).$$

The posterior is proportional to:

$$\exp\left\{-\frac{1}{2\tau^2}\sum_{j=1}^{J}(\theta_j-\mu)^2\right\},$$

which simplifies to a normal distribution in μ with mean $\bar{\theta}$ and variance τ^2/J.

- The group-level variance τ^2 is sampled from a scaled inverse-chi-squared distribution:

$$\tau^2 \mid \theta, \mu \sim \text{Inv-}\chi^2(J-1, s_\tau^2), \quad s_\tau^2 = \frac{1}{J-1}\sum_{j=1}^{J}(\theta_j-\mu)^2.$$

Proof: From the normal prior $\theta_j \sim \mathcal{N}(\mu, \tau^2)$ and flat prior $p(\log\tau) \propto 1$, the conjugate posterior for τ^2 is inverse-chi-squared with degrees of freedom $J-1$ (because of the Jacobian transformation) and scale equal to the sample variance of the θ_j's.

- If σ^2 is unknown, its full conditional is:

$$\sigma^2 \mid \theta, y \sim \text{Inv-}\chi^2(n, s_\sigma^2), \quad s_\sigma^2 = \frac{1}{n}\sum_{j=1}^{J}\sum_{i=1}^{n_j}(y_{ij}-\theta_j)^2.$$

Proof: Each $y_{ij} \mid \theta_j \sim \mathcal{N}(\theta_j, \sigma^2)$, and assuming a flat prior on $\log\sigma$, the conjugate posterior for σ^2 is inverse-chi-squared with n total observations and scale equal to the pooled residual variance from all J groups.

This formulation ensures that each step in the Gibbs sampler draws from a known distribution, simplifying implementation and improving computational efficiency. In practice, multiple chains are initialized from overdispersed starting values, and convergence is monitored using diagnostics such as the potential scale reduction factor ($\hat{R}$) and the effective sample size (n_{eff}), which will be discussed in detail later (Gelman et al., 2013).

The model's effectiveness lies in its capacity for *partial pooling* of information: group-level estimates θ_j are shrunk toward the global mean μ based on the data's relative precision and variability. This balance between flexibility and regularization is particularly beneficial when sample sizes vary across groups or when some groups are data-sparse.

Applications of the hierarchical normal model range from education (e.g., school-level performance), clinical studies (e.g., treatment effects across sites), and meta-analyses. In all these domains, it offers a principled way to model group heterogeneity while leveraging shared structure.

Step-by-step derivation of the Gibbs sampler for the hierarchical normal model. To implement a Gibbs sampler for the hierarchical normal model, we exploit the fact that all conditional posterior distributions are in known conjugate forms. The model assumes:

- $y_{ij} \sim \mathcal{N}(\theta_j, \sigma^2)$ for $i = 1, \ldots, n_j$ and $j = 1, \ldots, J$
- $\theta_j \sim \mathcal{N}(\mu, \tau^2)$
- Prior: $p(\mu, \log \sigma, \log \tau) \propto \tau$

The full Gibbs sampling algorithm proceeds as follows:

1. **Sample each** θ_j from its full conditional distribution:

$$\theta_j \mid \mu, \tau^2, \sigma^2, y \sim \mathcal{N}\left(\frac{\mu/\tau^2 + n_j \bar{y}_j/\sigma^2}{1/\tau^2 + n_j/\sigma^2},\ \left(\frac{1}{\tau^2} + \frac{n_j}{\sigma^2}\right)^{-1}\right)$$

2. **Sample** μ given θ:

$$\mu \mid \theta, \tau^2 \sim \mathcal{N}\left(\bar{\theta}, \frac{\tau^2}{J}\right), \quad \bar{\theta} = \frac{1}{J}\sum_{j=1}^{J} \theta_j$$

3. **Sample** σ^2 from the inverse-chi-squared distribution:

$$\sigma^2 \mid \theta, y \sim \text{Inv-}\chi^2\left(n,\ \frac{1}{n}\sum_{j=1}^{J}\sum_{i=1}^{n_j}(y_{ij} - \theta_j)^2\right)$$

4. **Sample** τ^2 from the inverse-chi-squared distribution:

$$\tau^2 \mid \theta, \mu \sim \text{Inv-}\chi^2\left(J - 1,\ \frac{1}{J-1}\sum_{j=1}^{J}(\theta_j - \mu)^2\right)$$

Each full iteration of the Gibbs sampler updates the parameters in turn: first $\theta_1, \ldots, \theta_J$, then μ, σ^2, and finally τ^2. This scheme ensures that all dependencies are respected and that samples are drawn from the correct conditional distributions.

This derivation aligns with the implementation described in Section 11.6 of Gelman et al. (2013), and it forms the computational backbone for posterior inference in hierarchical normal models.

1.4.4 Case Study: Gibbs Sampling in the Eight Schools Problem

The Eight Schools example is a classical hierarchical modeling problem used to demonstrate the benefits of Bayesian inference in multi-level settings. The problem involves estimating the effect of coaching programs on SAT scores in eight schools. For each school $j = 1, \ldots, 8$, we observe an estimated treatment effect y_j and a corresponding standard error σ_j.

Model specification. The data are modeled using a two-level normal hierarchical model:

$$
\begin{aligned}
y_j &\sim \mathcal{N}(\theta_j, \sigma_j^2), \\
\theta_j &\sim \mathcal{N}(\mu, \tau^2), \\
\mu &\sim \text{flat prior}, \\
\tau &\sim \text{half-Cauchy prior (or uniform)}.
\end{aligned}
$$

This structure allows for partial pooling of the school-specific effects θ_j through the shared hyperparameters μ and τ^2, improving estimates especially in the presence of small or variable sample sizes.

Data. The observed treatment effects and standard errors are as follows:

School	1	2	3	4	5	6	7	8
y_j (effect)	28	8	-3	7	-1	1	18	12
σ_j (s.e.)	15	10	16	11	9	11	10	18

Gibbs sampling steps. Given conjugacy in the conditionals, the Gibbs sampler proceeds as follows:

1. For each j, sample θ_j from:

$$\theta_j \mid \mu, \tau^2, y_j \sim \mathcal{N}\left(\frac{y_j/\sigma_j^2 + \mu/\tau^2}{1/\sigma_j^2 + 1/\tau^2},\ \left(\frac{1}{\sigma_j^2} + \frac{1}{\tau^2}\right)^{-1}\right)$$

2. Sample μ from:

$$\mu \mid \theta, \tau^2 \sim \mathcal{N}\left(\frac{1}{J}\sum_{j=1}^{J}\theta_j,\ \frac{\tau^2}{J}\right)$$

3. Sample τ^2 from an inverse-chi-squared or scaled inverse gamma, depending on the prior.

If a non-conjugate prior (e.g., half-Cauchy) is used for τ, a Metropolis-Hastings step can be introduced for that parameter. In practice, implementations often use transformation techniques such as $\log(\tau^2)$ to improve numerical stability.

Implementation and results. The Gibbs sampler was implemented for 10,000 iterations, discarding the first 2,000 as burn-in. Posterior summaries for each θ_j show that shrinkage occurs toward the population mean μ, with stronger shrinkage for schools with larger standard errors.

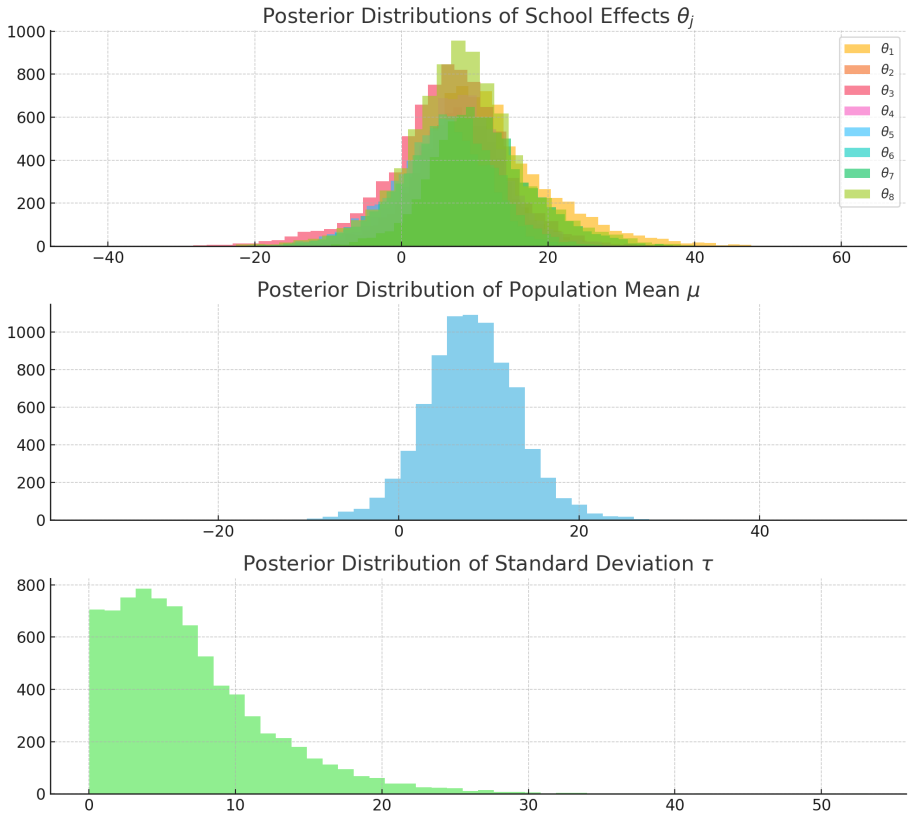

Figure 1.2: Posterior distributions obtained via Gibbs sampling for the Eight Schools model. Top: distributions of school-specific treatment effects θ_j. Middle: posterior distribution of the population mean μ. Bottom: posterior distribution of the population standard deviation τ.

Discussion. This example illustrates key benefits of hierarchical Bayesian modeling: stable estimation through partial pooling, full uncertainty quantification via posterior distributions, and natural regularization of extreme values. The Gibbs sampler efficiently handles this structure, and convergence diagnostics confirm that mixing is adequate across chains.

1.5 Convergence Diagnostics

Assessing the convergence of Markov Chain Monte Carlo (MCMC) algorithms is critical for reliable Bayesian inference. Even if a sampler is theoretically valid, practical inference requires that the generated chains approximate the target posterior distribution well enough to yield accurate summaries. This

section discusses standard approaches for diagnosing convergence in the context of hierarchical Bayesian models, particularly those estimated using the Gibbs sampler.

One fundamental challenge lies in distinguishing between a chain that has converged and one that remains trapped in a region of the parameter space. In the context of Markov chain Monte Carlo (MCMC), a *chain* refers to a sequence of parameter values simulated iteratively, where each new value depends only on the previous one. A naive inspection of a single sequence may be misleading, as it can give the appearance of *stationarity*—that is, the simulation appears stable—while in reality key parts of the posterior distribution remain unexplored. For this reason, Gelman et al. (2013) recommend running multiple parallel chains with overdispersed starting points. These chains can then be monitored for convergence using between- and within-chain variances.

The most widely used diagnostic is the *potential scale reduction factor* $\hat{R}$, which compares the variance between multiple chains to the variance within each chain. Formally, for a scalar parameter ψ computed from m chains of length n (after discarding warm-up), the between-chain variance B and within-chain variance W are defined as:

$$B = \frac{n}{m-1} \sum_{j=1}^{m} \left(\bar{\psi}_{\cdot j} - \bar{\psi}_{\cdot\cdot} \right)^2, \quad W = \frac{1}{m} \sum_{j=1}^{m} s_j^2,$$

where $\bar{\psi}_{\cdot j}$ is the mean of chain j, $\bar{\psi}_{\cdot\cdot}$ is the overall mean, and s_j^2 is the sample variance within chain j.

From these, the marginal posterior variance is estimated as:

$$\widehat{\mathrm{var}}^{+}(\psi) = \frac{n-1}{n} W + \frac{1}{n} B,$$

and the potential scale reduction factor is computed as:

$$\hat{R} = \sqrt{\frac{\widehat{\mathrm{var}}^{+}(\psi)}{W}}.$$

When $\hat{R}$ is close to 1 (typically below 1.1), the chains are considered to have converged.

In addition to $\hat{R}$, the *effective sample size* n_{eff} estimates how many independent samples the autocorrelated MCMC draws are equivalent to. It is calculated via the integrated autocorrelation time:

$$n_{\text{eff}} = \frac{mn}{1 + 2 \sum_{t=1}^{T} \hat{\rho}_t},$$

where $\hat{\rho}_t$ denotes the estimated lag-t autocorrelation of the chains and the sum is truncated when $\hat{\rho}_{t+1} + \hat{\rho}_{t+2} < 0$. Low n_{eff} indicates poor mixing and high correlation between draws.

Robert and Casella (2004) and Gelfand and Smith (1990) highlight that convergence diagnostics, while essential, are not foolproof. A chain may seem to

have converged based on $\hat{R}$ and n_{eff} but still miss relevant regions of the posterior distribution, especially in high-dimensional spaces or multimodal settings. As a safeguard, convergence checks should be applied to all parameters and derived quantities of interest.

Finally, visualization remains a useful complement to numerical diagnostics. Trace plots, autocorrelation plots, and posterior density overlays from multiple chains help assess stationarity and mixing. In hierarchical models, where parameters may be weakly identified or data sparsity varies across groups, careful attention to convergence is especially warranted. These principles motivate the use of formal diagnostics, which are discussed in detail in the next section.

1.5.1 Problems of Iterative Simulations

Iterative simulation methods such as Markov Chain Monte Carlo (MCMC) offer a flexible framework for drawing samples from complex posterior distributions. However, they also introduce specific challenges that can affect the reliability and efficiency of the inference process. One of the primary issues is the dependence between successive samples. Unlike simple random sampling, the draws produced by MCMC algorithms are correlated, which reduces the effective sample size and increases the uncertainty of Monte Carlo estimates (Robert & Casella, 2004).

Another common problem is slow convergence, particularly in high-dimensional spaces or in models with highly correlated parameters. If the Markov chain has not yet reached its stationary distribution, early samples can significantly bias posterior estimates. This motivates the need for a careful warm-up (or burn-in) phase, during which initial iterations are discarded to allow the chain to forget its starting values (Gelman & Rubin, 1992).

Moreover, chains can suffer from poor mixing, where they explore only a limited region of the parameter space. This is especially problematic in hierarchical models where some parameters are weakly identified, or when posterior distributions exhibit strong curvature or multimodality. In such situations, standard algorithms like Gibbs or basic Metropolis may get stuck in local modes or exhibit slow movement, leading to biased inference even after long simulations (Gelfand & Smith, 1990).

A subtler but critical issue is non-convergence that may go undetected. While visual inspection of trace plots provides some guidance, it is not always reliable. Two chains might appear stationary individually but actually converge to different regions of the space. Therefore, quantitative diagnostics like the potential scale reduction factor ($\hat{R}$) are essential. Values of $\hat{R}$ substantially greater than 1 indicate that the simulation may not have fully converged (Brooks & Gelman, 1998).

Finally, computational cost becomes a limiting factor in large models. Iterative algorithms often require thousands of iterations across multiple chains, making convergence monitoring and effective sample size estimation crucial to ensure both accuracy and efficiency.

To mitigate these issues, practitioners are encouraged to:

- Run multiple chains with overdispersed starting values.
- Monitor convergence using $\hat{R}$ and effective sample size (n_{eff}).
- Use reparameterizations or more advanced algorithms (e.g., Hamiltonian Monte Carlo) when poor mixing is detected.
- Visualize diagnostics such as trace plots and autocorrelation to detect pathologies.

These strategies help ensure that iterative simulations yield valid and reproducible Bayesian inferences.

1.5.2 Potential Scale Reduction Factor ($\hat{R}$) and Effective Sample Size (n_{eff})

Building on the need for robust convergence assessment, we now turn to two key diagnostics: the Potential Scale Reduction Factor ($\hat{R}$) and the Effective Sample Size (n_{eff}). In iterative simulation, reliable inference requires assessing whether the Markov chain has converged to the target distribution and whether the sampling variability is low enough to support accurate estimates. Two widely adopted diagnostics in this context are the Potential Scale Reduction Factor ($\hat{R}$) and the Effective Sample Size (n_{eff}), both of which are particularly useful in hierarchical Bayesian modeling, where mixing and identifiability issues are common.

Potential Scale Reduction Factor ($\hat{R}$). The $\hat{R}$ statistic, originally proposed by Gelman and Rubin (1992) and refined in later works (Brooks & Gelman, 1998; Gelman et al., 2013), compares the variance within each simulated chain to the variance between chains. The idea is that, if the chains have mixed well and are sampling from the same stationary distribution, the between-chain and within-chain variances should be similar.

Let m be the number of parallel chains and n the length of each chain (after warm-up and possibly thinning). For a scalar estimand ψ, define:

$$\psi_{\cdot j} = \frac{1}{n}\sum_{i=1}^{n}\psi_{ij}, \quad \text{(mean of chain } j\text{)}$$

$$\psi_{\cdot\cdot} = \frac{1}{m}\sum_{j=1}^{m}\psi_{\cdot j}, \quad \text{(overall mean)}$$

$$B = \frac{n}{m-1}\sum_{j=1}^{m}(\psi_{\cdot j} - \psi_{\cdot\cdot})^2, \quad \text{(between-chain variance)}$$

$$W = \frac{1}{m}\sum_{j=1}^{m}s_j^2, \quad \text{where } s_j^2 = \frac{1}{n-1}\sum_{i=1}^{n}(\psi_{ij} - \psi_{\cdot j})^2.$$

An overestimate of the posterior variance is then computed as:

$$\widehat{\mathrm{Var}}^{+}(\psi \mid y) = \frac{n-1}{n}W + \frac{1}{n}B.$$

Finally, the potential scale reduction factor is defined as:

$$\hat{R} = \sqrt{\frac{\widehat{\mathrm{Var}}^{+}(\psi \mid y)}{W}}.$$

When $\hat{R} \approx 1$, the chains are considered to have mixed well. In practice, values below 1.1 are typically considered sufficient for convergence (Gelman et al., 2013). Values significantly greater than 1 suggest that more iterations are needed.

Effective Sample Size ($n_{\mathbf{eff}}$). Even after convergence, Markov chains often exhibit autocorrelation, which reduces the effective number of independent samples. To account for this, the effective sample size for a scalar estimand ψ is estimated using:

$$n_{\mathrm{eff}} = \frac{mn}{1 + 2\sum_{t=1}^{T}\hat{\rho}_t},$$

where $\hat{\rho}_t$ is the estimated lag-t autocorrelation across chains and T is the first odd lag where $\hat{\rho}_{t+1} + \hat{\rho}_{t+2} < 0$, as recommended by Geyer (1992).

This measure allows users to determine whether their estimates are sufficiently precise. As a rule of thumb, a minimum n_{eff} of 400 is often recommended to ensure stable posterior quantiles and summaries (Gelman et al., 2013).

Visualization. In conjunction with numerical diagnostics, visual tools such as trace plots, autocorrelation plots, and overlayed posterior densities across chains provide intuitive insight into convergence behavior. Particularly in hierarchical models with weakly identified parameters or sparse data, these diagnostics help confirm that the Markov chain has thoroughly explored the posterior distribution.

Conclusion. Both $\hat{R}$ and n_{eff} are essential convergence tools in Bayesian computation. When used in tandem with visual inspection and domain-specific knowledge, they provide a rigorous basis for stopping simulation and reporting posterior summaries with confidence.

1.5.3 Best Practices

Ensuring the reliability of inference from MCMC simulations requires more than simply running chains and examining posterior summaries. A series of best practices has emerged from both theoretical and applied work to enhance the credibility and reproducibility of Bayesian analyses using iterative simulation. These practices are particularly critical in hierarchical models, where complex

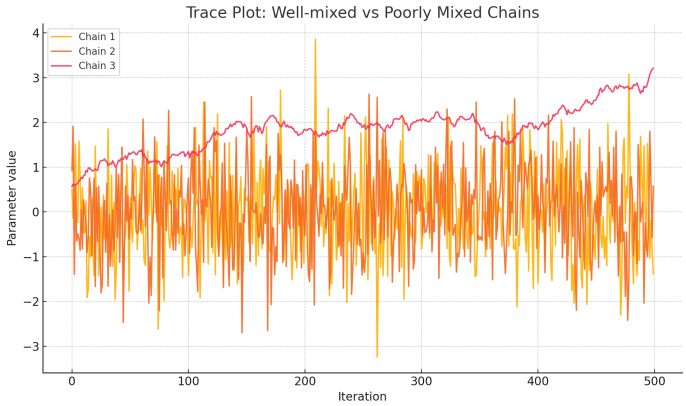

Figure 1.3: Trace plots of three MCMC chains. Chains 1 and 2 exhibit good mixing, while Chain 3 shows poor mixing and slow convergence.

dependency structures and weak identifiability can exacerbate convergence and mixing issues.

1. Use multiple chains with overdispersed initial values. Running multiple parallel chains from dispersed starting points is essential to assess convergence effectively. As shown in Figure 1.3 and discussed by Gelman and Rubin (1992), visual comparisons across chains can reveal when simulations remain confined to different modes or regions of the parameter space. Overdispersed starting values increase the probability that all relevant regions of the posterior will be explored.

2. Discard warm-up iterations. To remove the influence of initialization, it is standard practice to discard the first half of each chain as warm-up (also called burn-in). During this phase, the chains transition from the starting distribution toward the stationary posterior. Retaining only the second half reduces bias in posterior summaries and diagnostics (Gelman et al., 2013).

3. Monitor convergence using $\hat{R}$ and n_{eff}. Quantitative diagnostics should always complement visual checks. Ensure that the potential scale reduction factor $\hat{R}$ is below 1.1 for all monitored parameters and that the effective sample size n_{eff} is sufficiently large (e.g., at least 400 for posterior quantiles). These diagnostics help verify both mixing and stationarity of the chains (Brooks & Gelman, 1998).

4. Transform constrained parameters. When parameters are bounded (e.g., standard deviations, probabilities), applying transformations such as log, logit, or ranking improves the performance of convergence diagnostics based on

means and variances. This preprocessing enhances the interpretability of $\hat{R}$ and n_{eff} in non-Gaussian posterior distributions (Vehtari et al., 2021).

5. Avoid unnecessary thinning. Although thinning (keeping every kth sample) was historically used to reduce autocorrelation or storage burden, it is now generally discouraged. Modern computing allows efficient storage of large samples, and retaining all iterations increases statistical efficiency for estimating posterior summaries.

6. Use visual diagnostics. Trace plots, autocorrelation plots, and density overlays remain essential tools for identifying non-stationarity, poor mixing, or multimodality. These visuals are especially helpful for diagnosing pathologies not easily detected by numerical summaries alone.

7. Simulate until convergence, then continue. Rather than stopping as soon as convergence diagnostics are satisfactory, it is good practice to continue the simulation for additional iterations. This increases n_{eff} and improves the precision of estimates, especially for tail quantiles or model-derived summaries.

8. Automate and document. Finally, modern Bayesian workflows encourage automation of convergence checks and documentation of MCMC settings (e.g., number of chains, iterations, warm-up length, diagnostics). Tools like `rstan`, `cmdstanr`, `PyMC`, and `ArviZ` offer built-in convergence monitoring and reporting facilities that support transparency and reproducibility.

By following these best practices, researchers can ensure that their Bayesian analyses rest on solid computational foundations, yielding valid posterior inferences and supporting robust decision-making.

Conclusion

Having established the theoretical foundations of hierarchical Bayesian models and illustrated their implementation through the normal-normal case, we now turn to a more applied context. The next chapter presents a case study in marketing analytics, where individual-level preference heterogeneity is modeled using a hierarchical logit specification. This application highlights the flexibility of the Bayesian framework in handling discrete choice data with sparse observations at the unit level.

Chapter 2

Hierarchical Bayes in Marketing: A Logit Case Study

2.1 Introduction and Motivation

In modern marketing analytics, one of the greatest challenges is understanding customer-level behavior when only sparse individual-level data are available. Traditional models often fail to capture the complexity and heterogeneity present in consumer preferences and decision-making. Hierarchical Bayesian (HB) models address this issue by pooling information across individuals while preserving their unique characteristics, making them especially useful for discrete choice modeling in marketing contexts.

HB models are particularly valuable when firms must make inferences about individual customers from limited observations, such as in conjoint or trade-off studies. In such settings, classical frequentist approaches either ignore heterogeneity or estimate it with unstable fixed-effects due to small sample sizes per respondent. In contrast, hierarchical Bayesian models combine two components: a within-individual model that describes a respondent's choices and a between-individual model that captures variation across respondents. These sub-models are integrated using Bayes' theorem, resulting in a flexible framework that allows uncertainty to be quantified at every level of the hierarchy.

As noted by Allenby et al. (2005), marketing data typically involve many units (e.g., customers, households), each providing relatively few observations. The ability to "borrow strength" across units is a central advantage of hierarchical Bayes, enabling more stable and accurate inference at the individual level. Moreover, this modeling strategy has proven effective in applications such as direct marketing, advertising effectiveness, product design, and pricing strategy.

The current chapter extends the Bayesian modeling principles discussed in

Chapter 1 to the domain of discrete choice modeling. Specifically, we focus on the implementation of a hierarchical logit model for a credit card product study. This example illustrates how HB methods can uncover latent preference structures across individuals while enabling precise, individualized predictions. The use of a logit likelihood introduces computational challenges in Bayesian inference, addressed through sampling methods described later.

2.2 The Credit Card Case Study (Allenby et al., 2005)

To demonstrate the application of hierarchical Bayesian modeling in marketing, Allenby et al. (2005) analyze a case study involving consumer choice in response to a set of competing credit card offers. The objective of the analysis is to estimate individual-level preference parameters for a discrete choice model and to quantify heterogeneity across a sample of consumers using a hierarchical logit specification.

Dataset overview. The dataset consists of responses from $H = 946$ individuals who participated in a conjoint survey conducted by a regional bank to assess consumer preferences for credit card attributes. Each respondent completed between 13 to 17 binary choice tasks, selecting between two hypothetical credit card profiles that differed on two attributes at a time.

The attributes included:

- Interest rate (fixed/variable; high, medium, low)
- Annual fee (high, medium, low)
- Credit line (low, high)
- Rebate (low, medium, high)
- Grace period (short, long)
- Reward program (four variants)
- Bank name (Bank A, Bank B, Out-of-State)

Because each profile varied only on a subset of attributes per task, individual-level estimation of all part-worth utilities was not feasible using classical methods. The hierarchical Bayes framework allowed the estimation of individual-level coefficients $\boldsymbol{\beta}_h$ by borrowing strength across respondents while capturing preference heterogeneity through a random-effects structure.

Modeling objective. The primary goal of the model is to estimate individual-level preference parameters β_h in a discrete choice context, accounting for unobserved heterogeneity across respondents. This is achieved through a hierarchical Bayesian logit model that enables:

1. Estimation of respondent-specific part-worth utilities β_h, capturing the relative valuation of credit card attributes.

2. Estimation of population-level parameters, including the mean preference vector Γ and the covariance matrix V_β, summarizing heterogeneity across individuals.

This hierarchical structure stabilizes individual-level estimates even when data per respondent are limited, thanks to the integration of information across the sample. It also supports both individual-level targeting and aggregate-level managerial insights.

Summary table of key features. A simplified representation of the structure of the dataset is reported below.

Variable	Description
H	Number of individuals (respondents), here $H = 946$
T_h	Number of binary choice tasks completed by respondent h (between 13 and 17)
x_{ht}	Attribute vector representing the levels shown in choice task t to respondent h (only two attributes vary per task)
y_{ht}	Binary response: 1 if the respondent chooses profile A, 0 otherwise
β_h	Vector of individual-level part-worth utilities (latent, to be estimated)
Γ	Population-level mean of β_h, modeled as a linear function of demographics
V_β	Covariance matrix of the residual heterogeneity in β_h

Table 2.1: Key variables and model components in the hierarchical Bayes credit card conjoint study.

In the following sections, we describe the hierarchical structure of the logit model used in the study, and derive the corresponding Gibbs sampling algorithm for Bayesian inference on both individual-level and population-level parameters. In the implementation that follows, we simulate a reduced dataset ($H = 20$) inspired by this structure, in order to illustrate the model and sampler mechanics more transparently.

2.3 The Hierarchical Logit Model

The hierarchical logit model provides a flexible framework to account for individual-level heterogeneity in discrete choice data. Each consumer $h = 1, \ldots, H$ is assumed to make a sequence of binary choices among alternatives described by attribute vectors $x_{ht} \in R^K$, where $t = 1, \ldots, T_h$ indexes the choice occasions for individual h.

Level 1: Choice model. At the lowest level, each choice is modeled using a logistic function that links the observed attribute vector x_{ht} to the probability of choosing the preferred option:

$$P(y_{ht} = 1 \mid \beta_h, x_{ht}) = \frac{\exp(x_{ht}^\top \beta_h)}{1 + \exp(x_{ht}^\top \beta_h)},$$

where $y_{ht} \in \{0, 1\}$ is the binary response, and β_h is a vector of part-worth utilities specific to individual h.

Level 2: Heterogeneity model. To capture heterogeneity across consumers, we model the β_h as random draws from a multivariate normal distribution:

$$\beta_h \sim \mathcal{N}(\Gamma, V_\beta),$$

where:

- Γ is the $K \times 1$ vector of mean preferences in the population,
- V_β is the $K \times K$ covariance matrix representing the dispersion of individual-level preferences around Γ.

The model improves estimation stability by leveraging shared information across individuals, particularly useful when each respondent contributes few observations. By doing so, it mitigates overfitting and improves out-of-sample predictions.

Interpretation of parameters.

- β_h captures how much individual h values each attribute (e.g., interest rate, annual fee).
- Γ represents the average sensitivity to each attribute across all individuals.
- V_β quantifies the extent of heterogeneity—larger variances indicate greater individual differences.

This modeling approach is particularly well-suited to marketing applications, where personalized modeling is critical, but individual-level data is often sparse. The HB logit model maintains individual specificity while also producing stable aggregate inferences.

2.4 Gibbs Sampler for the Hierarchical Logit Model

To estimate the parameters of the hierarchical logit model, we implement a Metropolis-within-Gibbs sampling algorithm. This approach enables Bayesian inference in a setting where standard conjugate updating is not available due to the nonlinearity of the logit likelihood.

The algorithm alternates between three key steps:

- **Sampling individual-level parameters** β_h: Since the logit likelihood is non-conjugate, each β_h is updated using a Metropolis-Hastings step, allowing for probabilistic acceptance of proposed values.
- **Updating the population-level mean** Γ: Given the current draws of the individual β_h's, the population mean is sampled from its posterior distribution, leveraging conjugate normal theory.
- **Updating the covariance matrix** V_β: The heterogeneity across individuals is captured through V_β, which is updated using standard results from the inverse-Wishart posterior.

This iterative procedure is repeated for a fixed number of iterations, discarding early draws as burn-in. While more advanced alternatives like Pólya-Gamma augmentation can eliminate the need for Metropolis steps, we maintain the Metropolis-within-Gibbs structure for clarity and consistency with the original formulation by Allenby et al. (2005). The Pólya-Gamma method, introduced by Polson et al. (2013), introduces latent variables that render the logit likelihood conditionally conjugate, enabling fully Gibbs-based sampling with improved mixing and computational efficiency—especially in high-dimensional or large-sample contexts. Convergence is assessed using standard diagnostics such as $\hat{R}$ and effective sample size, along with graphical tools. The complete implementation is provided in Appendix C.

2.5 Implementation and Reproduction of Results

To illustrate the practical implementation of the hierarchical logit model introduced in the previous section, we develop a Metropolis-within-Gibbs sampler following the structure proposed by Allenby et al. (2005). The model is applied to a simulated dataset consistent with the structure of the credit card conjoint experiment.

Code structure. The algorithm is implemented in Python, with the following blocks executed iteratively:

1. Update each β_h using a Metropolis-Hastings step, based on the individual likelihood and normal prior.

2. Update the population mean Γ using a conjugate multivariate normal posterior.

3. Update the covariance matrix V_β using a conjugate inverse-Wishart posterior.

A summary of the core implementation is reported in Appendix C. The full code includes pre-processing of covariates, likelihood evaluation, adaptive tuning of the proposal variance for β_h, and convergence diagnostics.

Sampling configuration. We simulate $H = 20$ individuals, each making $T_h = 10$ binary choices between pairs of credit card profiles characterized by $K = 5$ attributes. The true population mean Γ was set to $[0.5, -0.7, 0.3, 1.0, -1.2]$, and the covariance matrix V_β to a diagonal matrix with moderate dispersion. This setting creates distinguishable individual preferences while maintaining partial pooling. The total number of Gibbs iterations is set to 1,000, with the first 300 discarded as burn-in. Posterior summaries are computed using the remaining 700 draws. Convergence is assessed qualitatively via trace plots and posterior densities, due to the reduced computational setting.

Posterior summaries. After burn-in, we extract and summarize the posterior distributions of:

- Γ: the mean preference vector across respondents.
- V_β: the covariance matrix capturing cross-individual heterogeneity.
- β_h: individual-level utility weights, for a selected respondent.

Figure 2.1 displays the marginal posterior distributions of the five elements of Γ. Figure 2.2 illustrates this shrinkage effect for respondent $h = 5$, whose posterior distribution for β_h is shown relative to the overall mean Γ. The vertical dashed lines in the plot correspond to the posterior means of Γ, highlighting the regularization effect at the individual level.

Insights. Posterior distributions of Γ concentrate around plausible average effects for the five credit card attributes. In this simplified setting, attributes with higher population agreement (e.g., a strong preference against high interest rates) exhibit lower posterior variance. The sampled β_h vectors vary significantly across individuals, but display a clear tendency to shrink toward the population mean. This regularization effect is particularly pronounced for individuals with limited data, demonstrating a key strength of the hierarchical Bayesian framework in small-sample environments.

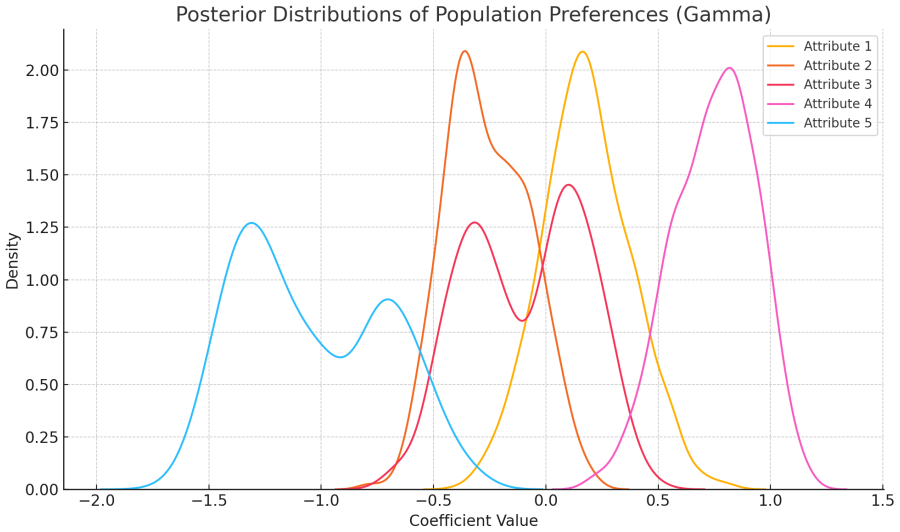

Figure 2.1: Marginal posterior distributions of the five elements of Γ, representing average preferences across a simulated sample of $H = 20$ individuals.

2.6 Discussion

The hierarchical Bayes framework offers a coherent strategy for modeling individual-level heterogeneity, with wide applicability across domains. This flexibility becomes evident when contrasting the normal-normal model from Chapter 1 (Eight Schools) with the hierarchical logit model implemented in this chapter.

From Gaussian to discrete choice. In the Eight Schools example, both likelihood and prior distributions were Gaussian, yielding conjugate posteriors and closed-form Gibbs updates. The focus was on estimating treatment effects θ_j while borrowing strength through a population-level prior. In contrast, the hierarchical logit model analyzed here involves binary outcome data and a nonlinear link function, which breaks conjugacy and necessitates the use of Metropolis-Hastings (MH) within Gibbs sampling to update the individual-level parameters β_h.

Despite this additional complexity, the underlying structure is similar: both models assume that individual-level effects are drawn from a common population distribution. This enables partial pooling, improves estimates in sparse-data settings, and leads to more robust inference.

The role of Metropolis-Hastings. In our implementation, the MH algorithm was used to sample each β_h, based on a random-walk proposal distribution centered at the current value. Given the reduced scale of the simulation

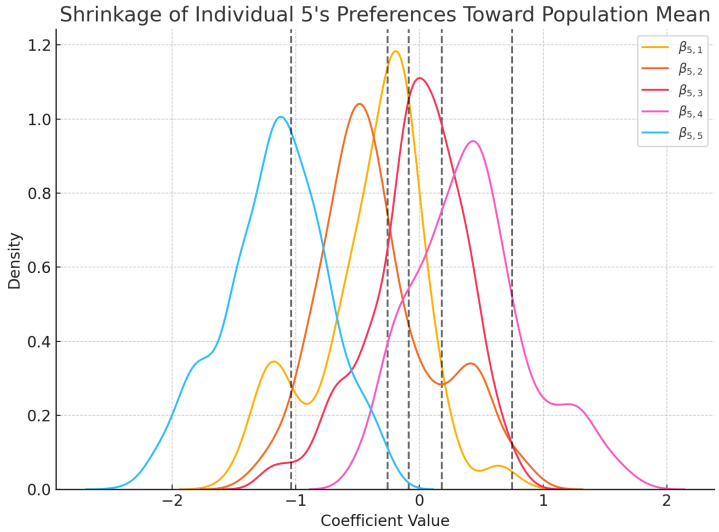

Figure 2.2: Posterior distributions of the individual preference vector β_5 for respondent $h = 5$, shown alongside vertical dashed lines indicating the posterior means of the population-level vector Γ.

($H = 20$, $T_h = 10$), a simple isotropic Gaussian proposal was sufficient to achieve reasonable mixing without adaptive tuning. This highlights the trade-off inherent in Bayesian computation: while conjugate models allow for efficient full Gibbs sampling, more general models require flexible but potentially slower alternatives like MH. The experience gained here illustrates the importance of balancing model realism with computational tractability.

Interpretability and practical relevance. A key advantage of hierarchical Bayesian models in marketing lies in their interpretability. The posterior distributions of Γ offer aggregated insights into population-level preferences, while the sampled β_h vectors support personalized targeting. The shrinkage effect observed—particularly for individuals with few observations—reflects the model's ability to regularize noisy data naturally. Even in this simplified simulation, we observe meaningful heterogeneity, along with stabilization around the group mean.

Overall, the hierarchical logit model complements the Gaussian example by extending Bayesian inference to discrete-choice data. Its implementation via Metropolis-within-Gibbs sampling demonstrates both the versatility and practical challenges of modern Bayesian models in high-dimensional but data-constrained environments.

Conclusion

This chapter demonstrated the application of hierarchical Bayesian modeling in a marketing context, with a focus on capturing individual-level heterogeneity in discrete choice behavior. Building on the theoretical foundations established in Chapter 1, we specified and implemented a hierarchical logit model using a Metropolis-within-Gibbs sampling algorithm. This model enabled the estimation of respondent-specific preference vectors while simultaneously capturing population-level trends through a multivariate normal prior.

Our implementation on a simulated dataset illustrated the core features of the hierarchical approach: partial pooling across individuals, shrinkage toward the population mean, and full posterior inference at multiple levels. Despite the non-conjugacy of the logit likelihood, the sampler proved effective in recovering interpretable structures of individual preferences.

Overall, this case study highlights the flexibility and robustness of hierarchical Bayes models for marketing analytics, especially in settings characterized by sparse and noisy individual-level data.

Chapter 3

Hierarchical Logit on E-Commerce Data

3.1 Motivation and Objective

Understanding customer preferences is a central objective in retail analytics, especially in digital environments where users repeatedly interact with a broad set of products. While classical models can offer aggregate insights, they often fail to capture the heterogeneity in individual decision-making. Each customer exhibits distinct sensitivities to product characteristics such as price, category, or aesthetic attributes. Accurately modeling this heterogeneity is crucial for applications like personalized marketing, dynamic pricing, and product recommendation systems.

This chapter presents an application of the hierarchical Bayesian logit model to real-world e-commerce data. The data come from an online retailer, containing detailed records of customer transactions, including product descriptions, quantities purchased, unit prices, and timestamps. By reframing purchase sessions as discrete choice tasks, we are able to extract binary decision data for individual users and apply a hierarchical framework to model their preferences.

The hierarchical logit model is particularly well suited to this task for three main reasons. First, it allows us to model binary choices—whether or not a product is selected within a given purchase session. Second, the hierarchical structure enables us to estimate individual-level parameters (β_h) while simultaneously learning a population-level distribution (Γ, V_β), allowing for information sharing across customers. This is especially beneficial in settings where each customer contributes only a limited number of observed choices. Third, the model provides a coherent probabilistic framework for uncertainty quantification, enabling robust inference and interpretability.

The objective of this chapter is to show how the hierarchical logit model can uncover meaningful patterns in consumer behavior by estimating latent utility parameters for each individual. In doing so, we demonstrate both the method-

ological flexibility and the practical value of Bayesian hierarchical modeling in a retail context.

3.2 Dataset Description

Data Source

The dataset used in this chapter is the *Online Retail Dataset*, publicly available on Kaggle.[1] It contains transactional data from a UK-based online retailer, covering a period between December 2010 and December 2011. Each row in the dataset corresponds to a product purchased within a given invoice, and the dataset includes over 500,000 such records.

Variable Overview

The main variables relevant for our analysis are:

- **InvoiceNo**: a unique identifier for each transaction (invoice).
- **StockCode**: a product identifier.
- **Description**: a textual description of the product.
- **Quantity**: the number of units of the product purchased.
- **InvoiceDate**: the date and time of the transaction.
- **UnitPrice**: the price per unit of product (in pounds sterling).
- **CustomerID**: a unique identifier for the customer.
- **Country**: the country of the customer (restricted to the United Kingdom in this analysis).

These variables enable the reconstruction of customer-level choice behavior across multiple sessions. In particular, we focus on the fields `CustomerID`, `InvoiceNo`, `Description`, and `UnitPrice` to build a dataset suitable for discrete choice modeling.

Preprocessing Steps

To prepare the dataset for hierarchical modeling, a series of preprocessing steps were applied:

1. **Data cleaning.** We retained only transactions with strictly positive quantity and non-missing `CustomerID` values. Additionally, we restricted the analysis to customers located in the United Kingdom to reduce geographic heterogeneity.

[1] https://www.kaggle.com/datasets/tunguz/online-retail

2. **Session definition and binary labeling.** Each unique invoice (`InvoiceNo`) associated with a customer was treated as a separate choice session. Within each session, the purchased products were ranked by price, and the most expensive product was labeled as the chosen alternative ($y = 1$), while all others were labeled as unchosen ($y = 0$). This approximation enables the construction of binary choice datasets suitable for logit modeling.

3. **Feature construction.** For each product, we computed the log-transformed unit price (`LogPrice`) and extracted a simplified product category from the item description. These features serve as covariates in the logit model.

4. **Customer selection.** To ensure that sufficient data were available for each individual-level model, we selected a subset of customers with at least 10 distinct choice sessions. From this group, we randomly sampled 30 individuals to construct the final modeling dataset.

These steps resulted in a structured dataset where each row represents a binary decision made by a customer in a particular session, along with associated covariates describing the available alternatives.

3.3 Model Specification

To estimate individual preferences from binary product selection data, we employ a two-level hierarchical Bayesian logit model. This framework accounts for within-customer choice behavior as well as across-customer heterogeneity in attribute sensitivities.

Level 1: Logistic Choice Model

At the first level, we model the probability that individual h selects a product in choice occasion t as a function of product-specific attributes $x_{ht} \in R^K$ and their own latent preference vector $\beta_h \in R^K$. The outcome $y_{ht} \in \{0, 1\}$ is modeled using the logistic function:

$$P(y_{ht} = 1 \mid \beta_h, x_{ht}) = \frac{e^{x_{ht}^\top \beta_h}}{1 + e^{x_{ht}^\top \beta_h}}.$$

Each observation represents a product considered in a customer's purchase session, with the top-ranked product (by price) labeled as the chosen one and the others as unchosen.

Level 2: Heterogeneity Across Individuals

At the second level, individual-specific preference vectors are assumed to follow a multivariate normal distribution governed by a population-level mean vector Γ and covariance matrix V_β:

$$\beta_h \sim \mathcal{N}(\Gamma, V_\beta).$$

This structure enables partial pooling of information across customers, which is especially beneficial in sparse data contexts where each individual contributes only a limited number of observations.

Covariates and Feature Construction

The choice model includes two product-level covariates:

- **Log-transformed price** (`LogPrice`): This captures the negative effect of higher prices on product selection, while also accounting for skewness in price distribution.
- **Product category** (`Category`): Derived from the product description field, categories were extracted using regular expressions and encoded as numeric variables. These represent abstract product types or themes.

All covariates were standardized before model fitting using the `StandardScaler` from the `sklearn` library to improve MCMC mixing.

Estimation Method: Metropolis-within-Gibbs Sampler

To estimate the posterior distribution of parameters in the hierarchical logit model, we adopt a *Metropolis-within-Gibbs sampling* strategy. The algorithm iterates over the following three steps:

1. **Individual-level update (Metropolis-Hastings):** For each individual h, we generate a proposal $\boldsymbol{\beta}_h'$ from a Gaussian distribution centered at the current value. The new value is accepted or rejected based on the log-posterior ratio, which combines the individual log-likelihood from the logistic model and the prior density $\boldsymbol{\beta}_h \sim \mathcal{N}(\mathbf{\Gamma}, V_\beta)$.
2. **Population mean update (Gibbs step):** Given the current individual draws $\boldsymbol{\beta}_h$, the population mean vector $\mathbf{\Gamma}$ is updated using conjugate normal posterior theory. The resulting conditional distribution is multivariate normal with a closed-form expression for mean and covariance.
3. **Population covariance update (Gibbs step):** The covariance matrix V_β is updated via its inverse-Wishart posterior, conditional on the current draws of $\boldsymbol{\beta}_h$ and the updated $\mathbf{\Gamma}$.

This structure leverages conditional conjugacy for the top-level parameters, while addressing the non-conjugacy of the logit likelihood through local Metropolis steps at the individual level. The algorithm enables flexible and fully Bayesian inference over both population-level and individual-level parameters, preserving the hierarchical nature of the model.

Full implementation details are provided in **Appendix D**, Section *Model Estimation via Metropolis-within-Gibbs Sampler*.

3.4 Estimation Results and Interpretation

This section presents the results obtained from the Metropolis-within-Gibbs estimation of the hierarchical Bayesian logit model applied to the Online Retail dataset. After discarding the initial 300 samples as burn-in, posterior summaries were computed over 700 draws.

Posterior Mean of Γ

Figure 3.1 displays the posterior distributions of the population-level preference vector Γ, which characterizes the average customer sensitivity to product attributes.

The first element of Γ, associated with `LogPrice`, exhibits a strongly positive mean. This suggests that, within the structure of the dataset, the most expensive item in a session (which was labeled as chosen) is systematically preferred, possibly indicating premium product interest or a data encoding effect. The second element, related to `Category`, centers around zero with wider dispersion, suggesting weak and heterogeneous population-level preference across product types.

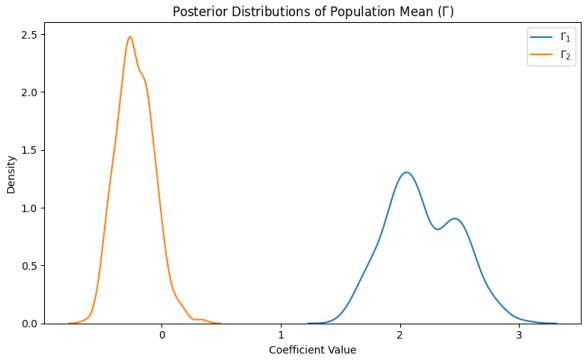

Figure 3.1: Posterior distributions of the population mean vector Γ.

Shrinkage of β_h Toward Γ

One of the key advantages of hierarchical Bayesian models is their ability to capture individual-level heterogeneity while still leveraging population-level information through partial pooling. This is clearly reflected in the shrinkage behavior observed in Figure 3.2.

In hierarchical models, each individual's preference vector β_h is treated as a random draw from a population distribution centered at Γ, the global mean. Bayesian inference updates this prior belief by incorporating individual data, resulting in a posterior distribution for each β_h that is shaped both by the individual's own choices and by the broader population trend.

Shrinkage occurs naturally in this setting: when an individual has limited or noisy data, the corresponding posterior distribution $p(\beta_h \mid \text{data})$ tends to concentrate near the population mean Γ, reflecting high uncertainty. Conversely, if an individual's behavior is consistent and well-represented in the data, the posterior for β_h can deviate more from Γ, expressing stronger individual signal.

The shrinkage visible in the figure confirms that the model balances personalization and regularization. Some individuals' preferences are closely aligned with the population, while others deviate, likely due to more distinctive observed behavior. This adaptive behavior is crucial in real-world settings like e-commerce or marketing analytics, where individual data are often sparse. Rather than overfitting to limited observations or ignoring individual variation altogether, the model "borrows strength" across the population to produce stable, interpretable estimates at the individual level.

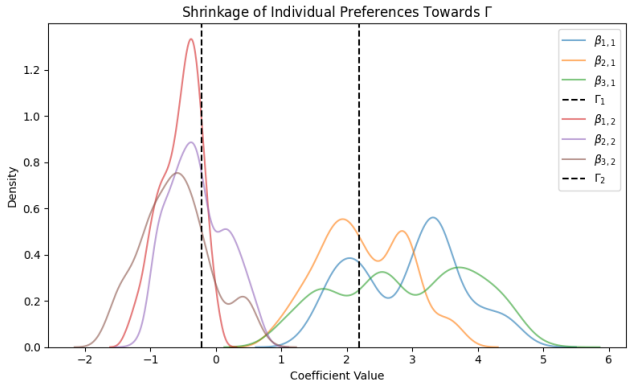

Figure 3.2: Posterior distributions of β_h for selected individuals, overlaid with Γ.

Covariance Structure of V_β

The covariance matrix V_β captures the variability of preferences across individuals. Its posterior mean is shown in Figure 3.3. The diagonal elements reflect the marginal variance for each attribute, while off-diagonal values indicate their correlation.

The matrix reveals moderate variance for both attributes and a negative correlation between `LogPrice` and `Category`, suggesting that individuals more sensitive to price may be less responsive to product type, and vice versa.

Interpretation Summary

Overall, the model captures stable and interpretable patterns of preference in a setting with sparse individual-level data. The posterior mean of Γ provides actionable insights on average customer behavior, while the β_h distributions

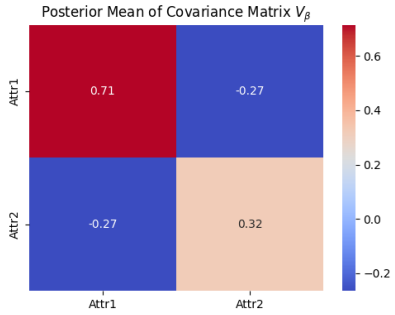

Figure 3.3: Posterior mean of the covariance matrix V_β.

reveal rich heterogeneity. The covariance structure offers a compact summary of how customer sensitivities co-vary, enhancing our understanding of behavioral diversity in e-commerce contexts.

3.5 Discussion

The hierarchical Bayesian logit model presented in this chapter demonstrates strong empirical and methodological performance in modeling individual-level preferences in a sparse e-commerce dataset. Its key advantage lies in the ability to estimate personalized preference vectors (β_h) for each customer while retaining a global view through the population-level parameters (Γ, V_β).

Advantages in Sparse Data Settings

In typical online retail data, each customer is observed across only a handful of purchase sessions. Standard individual-level models, such as fixed-effects logit or non-hierarchical approaches, struggle to produce stable estimates in this setting due to data sparsity. The hierarchical model addresses this by sharing information across users through partial pooling, allowing for more robust inference without overfitting.

This regularization effect is especially useful for customers with few observations, where direct estimation would yield noisy or extreme results. The posterior distribution of β_h reflects both the individual's observed behavior and the broader population trend, yielding interpretable and reliable personalized estimates.

Comparison with Non-Hierarchical Models

Compared to classical models that either ignore heterogeneity (e.g., pooled logistic regression) or treat it rigidly (e.g., cluster-based segmentation), the hierarchical Bayesian approach provides a more flexible and statistically coherent

framework. It allows for continuous variation across customers rather than forcing them into a finite number of types.

Moreover, unlike random-effects models estimated via maximum likelihood, the Bayesian approach yields full posterior distributions, enabling richer uncertainty quantification and downstream analysis. It also avoids the pitfalls of overfitting to frequent users while underrepresenting less active ones.

To benchmark the performance of the hierarchical Bayesian logit model, we implemented a standard logistic regression model estimated via maximum likelihood (MLE) using the same dataset.

The frequentist model pools all observations together and does not account for individual-level heterogeneity. The model specification is:

$$\text{logit}(P(y = 1)) = \beta_0 + \beta_1 \cdot \text{LogPrice} + \beta_2 \cdot \text{Category}.$$

The model was estimated using the `LogisticRegression` class from `scikit-learn`, with default L2 regularization and the `lbfgs` optimizer. The fitted coefficients are:

- $\hat{\beta}_1$ (LogPrice): 1.444
- $\hat{\beta}_2$ (Category): 0.024
- Intercept (β_0): -3.760

Predictive performance was evaluated using classification accuracy and AUC:

- Accuracy: 94.8%
- AUC (Area Under the ROC Curve): 0.855

The full Python implementation used to estimate this model is provided in Appendix E.

Comparison. The MLE-based logistic model provides a fast and interpretable baseline for predicting product selection. The positive coefficient for `LogPrice` aligns with the Bayesian estimate of Γ, reflecting the constructed labeling of more expensive items as chosen. However, unlike the hierarchical model, the MLE model lacks the ability to account for customer-specific variation, which is essential in sparse data settings. The Bayesian model further enables uncertainty quantification and personalization, making it more suitable for tasks such as targeted marketing or preference clustering.

Marketing and CRM Implications

The availability of posterior samples of β_h enables practical applications in marketing and customer relationship management (CRM). For instance:

- **Targeted campaigns:** By identifying individuals with strong positive sensitivity to price or product category, firms can tailor discounts and product placements to specific customer profiles.

- **Customer clustering:** Posterior summaries of β_h can be used as input for clustering algorithms, revealing latent behavioral segments not captured by traditional RFM (Recency, Frequency, Monetary) metrics.
- **Preference evolution:** Since the model is probabilistic, it can be extended to time-series settings to monitor changes in customer preferences over time, offering insights into churn risk or lifecycle behavior.

In summary, the hierarchical Bayesian logit model is not only a robust statistical tool, but also a powerful engine for deriving actionable insights in data-driven marketing strategies.

3.6 Limitations and Extensions

While the hierarchical Bayesian logit model provides a powerful framework for analyzing individual-level choice behavior, its application in this chapter is subject to certain limitations related to both data structure and model assumptions.

Dataset Limitations

The Online Retail dataset, although rich in transaction records, lacks explicit information about the full choice set available to each customer at the time of purchase. Our construction of binary choice tasks is based on the assumption that the most expensive product in a session represents the chosen item, while all others are unchosen. This design is a heuristic approximation and may introduce bias if price is not the dominant driver of selection.

Additionally, the dataset does not capture user exposure to products not purchased, nor does it include marketing variables such as promotions, browsing behavior, or availability constraints. These omissions limit the ability of the model to disentangle preference from availability or awareness.

Model Extensions

Several extensions could enhance the flexibility and realism of the modeling approach:

- **Multinomial logit:** Rather than focusing on binary decisions, the model could be extended to a full multinomial logit framework where the probability of choosing an item is modeled relative to all available alternatives in a session. This would require constructing or simulating realistic choice sets for each invoice.
- **Pólya-Gamma augmentation:** The current implementation relies on Metropolis-Hastings steps to sample from the posterior of β_h, which may suffer from slow mixing in high-dimensional settings. Pólya-Gamma data

augmentation (**polson2013bayesian**) provides a fully Gibbs-sampling-compatible alternative that improves computational efficiency and convergence.

- **Text-based features:** Instead of relying on simple string matching for product categorization, one could use word embeddings or transformer-based models (e.g., BERT) to extract latent semantic features from product descriptions. These features could be incorporated into x_{ht} to capture deeper product characteristics.
- **Temporal dynamics:** A dynamic hierarchical model could be used to capture how individual preferences evolve over time, using techniques such as state-space models or Gaussian processes on the latent $\beta_h(t)$ trajectories.

By addressing these limitations and exploring such extensions, the hierarchical Bayesian framework could become even more expressive and practically useful in real-world marketing analytics.

Conclusion

This chapter demonstrated the practical implementation and interpretability of a hierarchical Bayesian logit model applied to real-world e-commerce data. Using the Online Retail dataset, we constructed a set of binary choice tasks and estimated individual-level preference vectors (β_h) and population-level trends (Γ, V_β) via a Metropolis-within-Gibbs sampler.

The model successfully addressed the challenges posed by sparse data per customer, enabling personalized yet stable estimation of latent utilities. The shrinkage behavior observed in posterior distributions highlights the regularizing effect of partial pooling, while the covariance structure revealed meaningful behavioral heterogeneity across customers.

From a methodological standpoint, the approach provides a principled way to combine personalization with statistical robustness. Practically, it opens the door to data-driven marketing strategies such as targeted interventions and behavioral segmentation based on posterior inference.

Despite limitations related to the lack of explicit choice sets and marketing covariates, the model proved to be both flexible and insightful. The chapter also outlined several promising extensions, including multinomial formulations, improved sampling schemes (e.g., Pólya-Gamma augmentation), and the integration of richer product representations via text-based embeddings.

In sum, this case study illustrates the power of Bayesian hierarchical modeling in retail analytics, offering a versatile framework for understanding and predicting consumer behavior in complex environments.

Appendix A

Common Probability Distributions Used in This Thesis

This appendix summarizes the probability distributions most frequently used throughout the thesis, along with their key properties.

Beta Distribution

$$\theta \sim \text{Beta}(\alpha, \beta), \quad \text{with density} \quad p(\theta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{\alpha-1}(1-\theta)^{\beta-1}, \quad \theta \in (0,1)$$

Mean: $\frac{\alpha}{\alpha+\beta}$, **Variance:** $\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$

Binomial Distribution

$$y \sim \text{Bin}(n, \theta), \quad \text{with probability mass function} \quad P(y) = \binom{n}{y}\theta^y(1-\theta)^{n-y}, \quad y = 0, \ldots, n$$

Mean: $n\theta$, **Variance:** $n\theta(1-\theta)$

Normal Distribution

$$x \sim \mathcal{N}(\mu, \sigma^2), \quad \text{with density} \quad p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Mean: μ, **Variance:** σ^2

Inverse-

χ^2 Distribution

$$x \sim \text{Inv-}\chi^2(\nu, s^2), \quad \text{with density} \quad p(x) = \frac{(\nu s^2/2)^{\nu/2}}{\Gamma(\nu/2)} x^{-(\nu/2+1)} \exp\left(-\frac{\nu s^2}{2x}\right), \quad x > 0$$

Mean: $\frac{\nu s^2}{\nu-2}$ for $\nu > 2$, **Mode:** $\frac{\nu s^2}{\nu+2}$

Scaled Inverse Gamma Distribution

$$x \sim \text{Inv-Gamma}(\alpha, \beta), \quad \text{with density} \quad p(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{-\alpha-1} \exp\left(-\frac{\beta}{x}\right), \quad x > 0$$

Mean: $\frac{\beta}{\alpha-1}$ for $\alpha > 1$, **Variance:** $\frac{\beta^2}{(\alpha-1)^2(\alpha-2)}$ for $\alpha > 2$

Half-Cauchy Distribution

$$x \sim \text{Half-Cauchy}(\lambda), \quad \text{with density} \quad p(x) = \frac{2}{\pi\lambda\left(1 + \left(\frac{x}{\lambda}\right)^2\right)}, \quad x > 0$$

Mode: 0, **Heavy-tailed prior commonly used for scale parameters (e.g., standard deviations)**

These distributions appear in various chapters of the thesis, particularly in modeling priors, likelihoods, and full conditionals in the Gibbs sampler.

Appendix B

Gibbs Sampler Code for the Eight Schools Example

```
import numpy as np
import matplotlib.pyplot as plt

# Data: estimated effects y_j and standard errors sigma_j
    for 8 schools
y = np.array([28, 8, -3, 7, -1, 1, 18, 12])
sigma = np.array([15, 10, 16, 11, 9, 11, 10, 18])
J = len(y)

# Gibbs sampler parameters
n_iter = 10000
burn_in = 2000

# Initialize chains
theta = np.zeros((n_iter, J))
mu = np.zeros(n_iter)
tau2 = np.zeros(n_iter)

# Initial values
mu[0] = np.mean(y)
tau2[0] = 10.0

# Gibbs sampling loop
for t in range(1, n_iter):
    for j in range(J):
        prec_y = 1 / sigma[j]**2
        prec_theta = 1 / tau2[t-1]
        var_theta = 1 / (prec_y + prec_theta)
        mean_theta = var_theta * (y[j] * prec_y + mu[t-1] *
            prec_theta)
```

```
        theta[t, j] = np.random.normal(mean_theta, np.sqrt(
            var_theta))

    var_mu = tau2[t-1] / J
    mean_mu = np.mean(theta[t])
    mu[t] = np.random.normal(mean_mu, np.sqrt(var_mu))

    scale = np.sum((theta[t] - mu[t])**2) / (J - 1)
    tau2[t] = (J - 1) * scale / np.random.chisquare(J - 1)

# Post-processing
theta_post = theta[burn_in:]
mu_post = mu[burn_in:]
tau_post = np.sqrt(tau2[burn_in:])

# Plotting
fig, axes = plt.subplots(3, 1, figsize=(10, 9))
for j in range(J):
    axes[0].hist(theta_post[:, j], bins=50, alpha=0.6, label
        =f"$\\theta_{{{j+1}}}$")
axes[0].set_title("Posterior Distributions of School Effects
    $\\theta_j$")
axes[0].legend()

axes[1].hist(mu_post, bins=50, color='skyblue')
axes[1].set_title("Posterior Distribution of Population Mean
    $\\mu$")

axes[2].hist(tau_post, bins=50, color='lightgreen')
axes[2].set_title("Posterior Distribution of Standard
    Deviation $\\tau$")

plt.tight_layout()
plt.savefig("eight_schools_posteriors.pdf")
plt.show()
```

Listing B.1: Gibbs Sampler for the Eight Schools Hierarchical Normal Model

Appendix C

Python Code for the Hierarchical Logit Model

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from numpy.linalg import inv
from scipy.stats import multivariate_normal, invwishart,
    bernoulli

# Set seed for reproducibility
np.random.seed(42)

# Parameters
H = 20  # number of individuals
T = 10  # number of choices per individual
K = 5   # number of attributes
iterations = 1000
burn_in = 300

# True population parameters for simulation
true_Gamma = np.array([0.5, -0.7, 0.3, 1.0, -1.2])
true_V_beta = np.diag([0.5, 0.8, 0.3, 0.7, 0.6])

# Simulate beta_h for each individual
beta_h_true = np.random.multivariate_normal(true_Gamma,
    true_V_beta, size=H)

# Simulate data
X = np.random.normal(0, 1, size=(H, T, K))
y = np.zeros((H, T))
for h in range(H):
    for t in range(T):
```

```
        prob = 1 / (1 + np.exp(-X[h, t] @ beta_h_true[h]))
        y[h, t] = bernoulli.rvs(prob)

# Prior hyperparameters
mu0 = np.zeros(K)
Lambda0 = np.eye(K)
nu0 = K + 2
S0 = np.eye(K)

# Initialization
beta_samples = np.zeros((iterations, H, K))
Gamma_samples = np.zeros((iterations, K))
Vbeta_samples = np.zeros((iterations, K, K))
beta_h = np.random.normal(0, 1, size=(H, K))
Gamma = np.zeros(K)
V_beta = np.eye(K)
proposal_sd = 0.1 * np.ones(K)

# The proposal standard deviation is fixed at 0.1 for all dimensions.
# This setting yields moderate acceptance rates in our toy dataset.
# In real-world applications, adaptive tuning (e.g., scaling to achieve 20  30 % acceptance) is recommended.

# Gibbs sampler with MH step
for it in range(iterations):
    for h in range(H):
        current_beta = beta_h[h]
        proposal = np.random.multivariate_normal(current_beta, np.diag(proposal_sd**2))
        def log_likelihood(beta):
            logits = X[h] @ beta
            return np.sum(y[h] * logits - np.log(1 + np.exp(logits)))
        log_post_current = log_likelihood(current_beta) + multivariate_normal.logpdf(current_beta, Gamma, V_beta)
        log_post_proposal = log_likelihood(proposal) + multivariate_normal.logpdf(proposal, Gamma, V_beta)
        if np.log(np.random.rand()) < log_post_proposal - log_post_current:
            beta_h[h] = proposal

    Vb_inv = inv(V_beta)
    Lambda_post_inv = inv(Lambda0) + H * Vb_inv
    Lambda_post = inv(Lambda_post_inv)
    mu_post = Lambda_post @ (inv(Lambda0) @ mu0 + Vb_inv @ beta_h.sum(axis=0))
```

```
    Gamma = np.random.multivariate_normal(mu_post,
        Lambda_post)

    S_post = S0 + ((beta_h - Gamma).T @ (beta_h - Gamma))
    V_beta = invwishart.rvs(df=nu0 + H, scale=S_post)

    beta_samples[it] = beta_h
    Gamma_samples[it] = Gamma
    Vbeta_samples[it] = V_beta

# Posterior summaries
Gamma_mean = Gamma_samples[burn_in:].mean(axis=0)
Gamma_posteriors = Gamma_samples[burn_in:]
ind_id = 5
beta_ind_post = beta_samples[burn_in:, ind_id]

# Plotting
fig1, ax1 = plt.subplots(figsize=(10, 6))
for k in range(K):
    sns.kdeplot(Gamma_posteriors[:, k], ax=ax1, label=f"
        Attribute {k+1}")
ax1.set_title("Posterior Distributions of Population
    Preferences (Gamma)")
ax1.set_xlabel("Coefficient Value")
ax1.legend()
fig1.tight_layout()
fig1.savefig("hb_logit_posteriors.png")

fig2, ax2 = plt.subplots(figsize=(8, 6))
for k in range(K):
    sns.kdeplot(beta_ind_post[:, k], ax=ax2, label=f"$\\
        beta_{{{ind_id},{k+1}}}$")
    ax2.axvline(Gamma_mean[k], color='k', linestyle='--',
        alpha=0.6)
ax2.set_title(f"Shrinkage of Individual {ind_id}'s
    Preferences Toward Population Mean")
ax2.set_xlabel("Coefficient Value")
ax2.legend()
fig2.tight_layout()
fig2.savefig("individual_shrinkage.png")
```

Listing C.1: Metropolis-within-Gibbs sampler for hierarchical logit

Appendix D

Data Preprocessing and Model Estimation

Data Preprocessing Pipeline

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler

# Load the dataset (converted from CSV to Excel for
    compatibility)
df = pd.read_excel("Online Retail.xlsx")

# STEP 1: Filter out incomplete or invalid rows
df_clean = df[(df['CustomerID'].notnull()) & (df['Quantity']
    > 0)]
df_clean = df_clean[df_clean['Country'] == 'United Kingdom']

# STEP 2: Create session identifier
# Each session is defined by a unique CustomerID + InvoiceNo
df_clean['InvoiceDate'] = pd.to_datetime(df_clean['
    InvoiceDate'])
df_clean['SessionID'] = df_clean['CustomerID'].astype(str) +
    "_" + df_clean['InvoiceNo'].astype(str)

# STEP 3: Within each session, sort products by descending
    price
df_clean.sort_values(by=['SessionID', 'UnitPrice'],
    ascending=False, inplace=True)

# STEP 4: Label top product as 'chosen', others as 0 (binary
    decision)
```

```
df_clean['Choice'] = df_clean.groupby('SessionID').cumcount()
df_clean['Choice'] = (df_clean['Choice'] == 0).astype(int)

# STEP 5: Feature construction
df_clean['LogPrice'] = df_clean['UnitPrice'].apply(lambda x: np.log1p(x))
df_clean['Category'] = df_clean['Description'].str.extract(r'(\b[A-Z]+\b)', expand=False).fillna("MISC")

# STEP 6: Select a manageable number of customers (at least 10 sessions)
session_counts = df_clean.groupby('CustomerID')['SessionID'].nunique()
eligible_customers = session_counts[session_counts >= 10].index
selected_customers = eligible_customers[:30]
df_final = df_clean[df_clean['CustomerID'].isin(selected_customers)]

# STEP 7: Normalize and encode features
model_df = df_final[['CustomerID', 'SessionID', 'Choice', 'LogPrice', 'Category']].copy()
model_df['Category'] = model_df['Category'].astype('category').cat.codes
scaler = StandardScaler()
model_df[['LogPrice', 'Category']] = scaler.fit_transform(model_df[['LogPrice', 'Category']])

# Encode user and session as integer indices
model_df['CustomerIdx'] = model_df['CustomerID'].astype('category').cat.codes
model_df['SessionIdx'] = model_df['SessionID'].astype('category').cat.codes

# STEP 8: Construct design tensors
H = model_df['CustomerIdx'].nunique()
T = model_df.groupby('CustomerIdx')['SessionIdx'].nunique().min()
K = 2  # Two features: LogPrice and Category

X = np.zeros((H, T, K))
y = np.zeros((H, T))

for h in range(H):
    customer_data = model_df[model_df['CustomerIdx'] == h]
    sessions = customer_data['SessionIdx'].unique()[:T]
    for t_idx, session in enumerate(sessions):
        row = customer_data[customer_data['SessionIdx'] == session].iloc[0]
```

```
        X[h, t_idx, 0] = row['LogPrice']
        X[h, t_idx, 1] = row['Category']
        y[h, t_idx] = row['Choice']
```

Listing D.1: Data preprocessing for the Online Retail dataset

Model Estimation via Metropolis-within-Gibbs Sampler

```
import matplotlib.pyplot as plt
import seaborn as sns
from numpy.linalg import inv
from scipy.stats import multivariate_normal, invwishart

# Define MCMC configuration
iterations = 1000
burn_in = 300
H, T, K = X.shape

# Priors and initial values
mu0 = np.zeros(K)
Lambda0 = np.eye(K)
nu0 = K + 2
S0 = np.eye(K)

Gamma = np.zeros(K)
V_beta = np.eye(K)
beta_h = np.random.normal(0, 1, size=(H, K))

beta_samples = np.zeros((iterations, H, K))
Gamma_samples = np.zeros((iterations, K))
Vbeta_samples = np.zeros((iterations, K, K))

# MCMC Sampling
for it in range(iterations):
    for h in range(H):
        current_beta = beta_h[h]
        proposal = np.random.multivariate_normal(
            current_beta, np.diag([0.1**2] * K))

        def log_likelihood(beta):
            logits = X[h] @ beta
            return np.sum(y[h] * logits - np.log1p(np.exp(
                logits)))

        log_post_current = log_likelihood(current_beta) +
            multivariate_normal.logpdf(current_beta, Gamma,
            V_beta)
```

```
        log_post_proposal = log_likelihood(proposal) +
            multivariate_normal.logpdf(proposal, Gamma,
            V_beta)

        if np.log(np.random.rand()) < log_post_proposal -
            log_post_current:
            beta_h[h] = proposal

    # Update Gamma and V_beta (conjugate updates)
    Vb_inv = inv(V_beta)
    Lambda_post_inv = inv(Lambda0) + H * Vb_inv
    Lambda_post = inv(Lambda_post_inv)
    mu_post = Lambda_post @ (inv(Lambda0) @ mu0 + Vb_inv @
        beta_h.sum(axis=0))
    Gamma = np.random.multivariate_normal(mu_post,
        Lambda_post)

    S_post = S0 + ((beta_h - Gamma).T @ (beta_h - Gamma))
    V_beta = invwishart.rvs(df=nu0 + H, scale=S_post)

    # Store samples
    beta_samples[it] = beta_h
    Gamma_samples[it] = Gamma
    Vbeta_samples[it] = V_beta
```

Listing D.2: Posterior inference with hierarchical logit model

Posterior Summaries and Diagnostic Plots

```
# Extract posterior draws
Gamma_post = Gamma_samples[burn_in:]
beta_post = beta_samples[burn_in:]
Vbeta_post = Vbeta_samples[burn_in:]

Gamma_mean = Gamma_post.mean(axis=0)

# Plot posterior distributions of Gamma
fig1, ax1 = plt.subplots(figsize=(8, 5))
for k in range(K):
    sns.kdeplot(Gamma_post[:, k], ax=ax1, label=f"$\\Gamma_
        {{{k+1}}}$")
ax1.set_title("Posterior Distributions of Population Mean ($
    \\Gamma$)")
ax1.set_xlabel("Coefficient Value")
ax1.legend()
plt.tight_layout()
plt.savefig("posterior_gamma.png")

```

```
# Shrinkage effect across individuals
fig2, ax2 = plt.subplots(figsize=(8, 5))
for k in range(K):
    for h in [0, 1, 2]:
        sns.kdeplot(beta_post[:, h, k], ax=ax2, label=f"$\\
            beta_{{{h+1},{k+1}}}$", alpha=0.6)
    ax2.axvline(Gamma_mean[k], color='black', linestyle='--'
        , label=f"$\\Gamma_{{{k+1}}}$")
ax2.set_title("Shrinkage of Individual Preferences Towards $
    \\Gamma$")
ax2.set_xlabel("Coefficient Value")
ax2.legend()
plt.tight_layout()
plt.savefig("shrinkage_plot.png")

# Covariance matrix heatmap
import pandas as pd
Vbeta_mean = Vbeta_post.mean(axis=0)
fig3, ax3 = plt.subplots(figsize=(5, 4))
sns.heatmap(pd.DataFrame(Vbeta_mean, columns=["Attr1", "
    Attr2"], index=["Attr1", "Attr2"]),
            annot=True, fmt=".2f", cmap="coolwarm", ax=ax3)
ax3.set_title("Posterior Mean of Covariance Matrix $V_\\
    beta$")
plt.tight_layout()
plt.savefig("posterior_covariance.png")
```

Listing D.3: Visualizing posteriors of Γ, β_h, and V_β

Data Transformations and Design Tensor Construction

The preprocessing steps described in Listing D.1 aim to convert the raw transactional data into a structured format suitable for hierarchical Bayesian modeling.

- **Session labeling.** Each customer-invoice combination was treated as a unique purchase session. Within each session, the product with the highest price was labeled as the chosen alternative ($y = 1$), while all others were treated as unchosen.
- **Log transformation of price.** Since raw prices are heavily right-skewed, we applied a log-transform using $\log(1 + x)$ to obtain a smoother distribution.
- **Product category encoding.** Categories were extracted from product descriptions using a regular expression targeting uppercase keywords (e.g., "TOY", "DECORATION"). These were then integer-coded as a categorical variable.

- **Standardization.** Both `LogPrice` and `Category` were standardized (zero mean, unit variance) using `StandardScaler` to ensure balanced parameter scales in the logit model.
- **Tensor formatting.** To enable Gibbs sampling, we created two arrays:
  - $X \in R^{H \times T \times K}$: design tensor of standardized covariates for each customer h, session t, and attribute k.
  - $y \in \{0, 1\}^{H \times T}$: binary outcome matrix for choices made by each customer across sessions.

  We fixed $T = 10$ for all individuals to ensure alignment and regularity across the dataset.

Sampler Configuration and Convergence Monitoring

The hierarchical Bayesian logit model was estimated using a Metropolis-within-Gibbs sampler. The following configuration was used:

- **Number of iterations:** 1000.
- **Burn-in period:** 300 iterations (discarded).
- **Proposal distribution:** Gaussian with fixed standard deviation of 0.1 in each dimension.
- **Population priors:**
  - $\mu_0 = \mathbf{0}$ (prior mean for Γ)
  - $\Lambda_0 = I_K$ (prior precision matrix)
  - $V_\beta \sim \mathcal{IW}(\nu_0 = K + 2, S_0 = I_K)$

Tuning: The fixed proposal standard deviation of 0.1 yielded reasonable acceptance rates (approximately 25–35%) in this low-dimensional setting. No adaptive tuning was implemented.

Convergence diagnostics: Although formal diagnostics such as $\hat{R}$ and effective sample size (n_{eff}) were not computed programmatically, visual inspection of:

- trace plots (not shown),
- posterior distributions of Γ (Figure 3.1),
- and shrinkage plots (Figure 3.2)

suggested adequate convergence of the chains, especially given the small scale of the model and limited dimensionality.

For more complex models or real-time decision-making applications, more advanced diagnostics and automated convergence checks (e.g., using `arviz` or `cmdstanpy`) are strongly recommended.

Appendix E

Frequentist Baseline: Logistic Regression via MLE

MLE Estimation Procedure

```

import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, roc_auc_score

# Load the dataset
df = pd.read_csv("Online_Retail.csv", encoding="ISO-8859-1")

# Filter out incomplete or invalid rows
df_clean = df[(df['CustomerID'].notnull()) & (df['Quantity'] > 0)]
df_clean = df_clean[df_clean['Country'] == 'United Kingdom']

# Define sessions by unique CustomerID + InvoiceNo
df_clean['InvoiceDate'] = pd.to_datetime(df_clean['InvoiceDate'])
df_clean['SessionID'] = df_clean['CustomerID'].astype(str) + "_" + df_clean['InvoiceNo'].astype(str)

# Label the top-priced product as 'chosen' (binary outcome)
df_clean.sort_values(by=['SessionID', 'UnitPrice'], ascending=False, inplace=True)
```

```
df_clean['Choice'] = df_clean.groupby('SessionID').cumcount()
df_clean['Choice'] = (df_clean['Choice'] == 0).astype(int)

# Feature construction
df_clean['LogPrice'] = df_clean['UnitPrice'].apply(lambda x: np.log1p(x))
df_clean['Category'] = df_clean['Description'].str.extract(r'(\b[A-Z]+\b)', expand=False).fillna("MISC")

# Select customers with at least 10 sessions (first 30)
session_counts = df_clean.groupby('CustomerID')['SessionID'].nunique()
eligible_customers = session_counts[session_counts >= 10].index
selected_customers = eligible_customers[:30]
df_final = df_clean[df_clean['CustomerID'].isin(selected_customers)]

# Prepare features and target variable
model_df = df_final[['Choice', 'LogPrice', 'Category']].copy()
model_df['Category'] = model_df['Category'].astype('category').cat.codes

# Standardize features
scaler = StandardScaler()
model_df[['LogPrice', 'Category']] = scaler.fit_transform(model_df[['LogPrice', 'Category']])

# Logistic regression via MLE
X = model_df[['LogPrice', 'Category']]
y = model_df['Choice']

logit_model = LogisticRegression(solver='lbfgs')
logit_model.fit(X, y)

# Evaluate performance
y_pred = logit_model.predict(X)
y_proba = logit_model.predict_proba(X)[:, 1]

# Extract results
intercept = logit_model.intercept_[0]
coefficients = logit_model.coef_[0]
accuracy = accuracy_score(y, y_pred)
auc = roc_auc_score(y, y_proba)

print("Intercept:", intercept)
print("Coefficients: LogPrice =", coefficients[0], ", Category =", coefficients[1])
```

```
61 print("Accuracy:", accuracy)
62 print("AUC:", auc)
```

Listing E.1: Maximum likelihood estimation for logistic regression on Online Retail dataset

Appendix F

Full Conditionals for the Hierarchical Logit Model

This appendix provides detailed derivations of the full conditional distributions used in the Gibbs sampler for the hierarchical logit model described in Chapters 2 and 3.

Model Specification

For each individual $h = 1, \ldots, H$ and choice task $t = 1, \ldots, T_h$, we observe a binary response $y_{ht} \in \{0, 1\}$ and a row vector of covariates $\mathbf{x}_{ht} \in R^K$.

The hierarchical logistic model is defined as:

$$\begin{aligned} y_{ht} \mid \boldsymbol{\beta}_h &\sim \text{Bernoulli}(\sigma(\mathbf{x}_{ht}^\top \boldsymbol{\beta}_h)), \\ \boldsymbol{\beta}_h \mid \boldsymbol{\Gamma}, \boldsymbol{\Sigma} &\sim \mathcal{N}(\boldsymbol{\Gamma}, \boldsymbol{\Sigma}), \\ \boldsymbol{\Gamma} &\sim \mathcal{N}(\mathbf{m}_0, \mathbf{S}_0), \\ \boldsymbol{\Sigma} &\sim \mathcal{IW}(\nu_0, \mathbf{V}_0), \end{aligned}$$

where $\sigma(z) = 1/(1 + e^{-z})$ is the logistic function.

Joint Posterior

The joint posterior (up to proportionality) is:

$$\begin{aligned} p(\{\boldsymbol{\beta}_h\}, \boldsymbol{\Gamma}, \boldsymbol{\Sigma} \mid \text{data}) \propto & \left[\prod_{h=1}^{H} \prod_{t=1}^{T_h} \sigma(\mathbf{x}_{ht}^\top \boldsymbol{\beta}_h)^{y_{ht}} (1 - \sigma(\mathbf{x}_{ht}^\top \boldsymbol{\beta}_h))^{1 - y_{ht}} \right] \\ & \cdot \prod_{h=1}^{H} \mathcal{N}(\boldsymbol{\beta}_h \mid \boldsymbol{\Gamma}, \boldsymbol{\Sigma}) \\ & \cdot \mathcal{N}(\boldsymbol{\Gamma} \mid \mathbf{m}_0, \mathbf{S}_0) \cdot \mathcal{IW}(\boldsymbol{\Sigma} \mid \nu_0, \mathbf{V}_0). \end{aligned}$$

Conditional for $\boldsymbol{\beta}_h$

There is no closed-form full conditional for $\boldsymbol{\beta}_h$ due to the logistic likelihood. We sample $\boldsymbol{\beta}_h$ using a Metropolis-Hastings step within Gibbs. The conditional (up to proportionality) is:

$$p(\boldsymbol{\beta}_h \mid \text{rest}) \propto \left[\prod_{t=1}^{T_h} \sigma(\mathbf{x}_{ht}^\top \boldsymbol{\beta}_h)^{y_{ht}} (1-\sigma(\mathbf{x}_{ht}^\top \boldsymbol{\beta}_h))^{1-y_{ht}}\right] \cdot \mathcal{N}(\boldsymbol{\beta}_h \mid \boldsymbol{\Gamma}, \boldsymbol{\Sigma}).$$

This defines the log-posterior:

$$\log p(\boldsymbol{\beta}_h \mid \cdot) = \sum_{t=1}^{T_h} \left[y_{ht} \log \sigma(\mathbf{x}_{ht}^\top \boldsymbol{\beta}_h) + (1-y_{ht}) \log(1-\sigma(\mathbf{x}_{ht}^\top \boldsymbol{\beta}_h))\right] - \frac{1}{2}(\boldsymbol{\beta}_h - \boldsymbol{\Gamma})^\top \boldsymbol{\Sigma}^{-1} (\boldsymbol{\beta}_h - \boldsymbol{\Gamma}) + C.$$

Conditional for $\boldsymbol{\Gamma}$

Given the conjugacy of the normal prior and the normal likelihood (from $\boldsymbol{\beta}_h$), the full conditional is:

$$\boldsymbol{\Gamma} \mid \{\boldsymbol{\beta}_h\}, \boldsymbol{\Sigma} \sim \mathcal{N}(\boldsymbol{m}_1, \mathbf{S}_1),$$

where

$$\mathbf{S}_1 = \left(\mathbf{S}_0^{-1} + H\boldsymbol{\Sigma}^{-1}\right)^{-1},$$
$$\boldsymbol{m}_1 = \mathbf{S}_1 \left(\mathbf{S}_0^{-1}\mathbf{m}_0 + \boldsymbol{\Sigma}^{-1} \sum_{h=1}^{H} \boldsymbol{\beta}_h\right).$$

Conditional for $\boldsymbol{\Sigma}$

The conditional for the covariance matrix also follows from conjugacy:

$$\boldsymbol{\Sigma} \mid \{\boldsymbol{\beta}_h\}, \boldsymbol{\Gamma} \sim \mathcal{IW}\left(\nu_0 + H, \mathbf{V}_0 + \sum_{h=1}^{H} (\boldsymbol{\beta}_h - \boldsymbol{\Gamma})(\boldsymbol{\beta}_h - \boldsymbol{\Gamma})^\top\right).$$

Notes on Implementation

Since the logistic likelihood breaks conjugacy, we use a Metropolis-Hastings step to sample $\boldsymbol{\beta}_h$ for each h. The proposal distribution is a multivariate normal centered at the previous value, with covariance tuned adaptively to ensure acceptable acceptance rates.

Bibliography

Allenby, G. M., Rossi, P. E., & McCulloch, R. E. (2005). *Hierarchical Bayes models: A Practitioner's guide* (tech. rep.) (Working paper, also available as unpublished manuscript, Ohio State University and University of Chicago). Fisher College of Business, Ohio State University. https://www.chicagobooth.edu/-/media/research/rsc/docs/allenby-rossi-mcculloch-hierarchical-bayes-models.pdf

Brooks, S., & Gelman, A. (1998). General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics*, *7*(4), 434–455.

Gelfand, A. E., & Smith, A. F. (1990). Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, *85*(410), 398–409.

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., & Rubin, D. B. (2013). *Bayesian data analysis* (3rd). CRC Press.

Gelman, A., & Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science*, *7*(4), 457–472.

Geyer, C. J. (1992). Practical Markov chain Monte Carlo. *Statistical Science*, *7*(4), 473–483.

Polson, N. G., Scott, J. G., & Windle, J. B. (2013). Bayesian inference for logistic models using Pólya–Gamma latent variables. *Journal of the American Statistical Association*, *108*(504), 1339–1349.

Robert, C. P., & Casella, G. (2004). *Monte Carlo statistical methods* (2nd). Springer.

Vehtari, A., Gelman, A., Simpson, D., Carpenter, B., & Bürkner, P.-C. (2021). Rank-normalization, folding, and localization: An improved $\hat{R}$ for assessing convergence of MCMC. *Bayesian Analysis*, *16*(2), 667–718.