Senior Thesis in Mathematics

Bridging Latent Dirichlet Allocation and Network Community Detection

An Exploration for Latent Structure Discovery

Author:
Jiarui Hubery Hu

Advisor:
Prof. Johanna S. Hardin

Submitted to Pomona College in Partial Fulfillment
of the Degree of Bachelor of Arts

May 18, 2025

Abstract

This thesis presents a study of latent structure discovery, connecting topic modeling and community detection. We trace the evolution of topic modeling from early heuristic methods such as Latent Semantic Indexing (LSI) and Probabilistic Latent Semantic Indexing (pLSI) to more robust Bayesian framework of Latent Dirichlet Allocation (LDA). By deriving the theoretical foundations of LDA, we provide its precise mathematical formulation.

Furthermore, this work presents an innovative perspective inspired by Gerlach et al. (2018) by mapping word–document associations onto a bipartite network, thereby bridging the gap between topic modeling and community detection. Leveraging established network analysis techniques, such as the hierarchical Stochastic Block Model (hSBM), we uncover latent group structures that parallel those discovered through probabilistic methods. We reveal an intriguing equivalence between the discrete linear algebra techniques used in network clustering and the continuous Bayesian framework underlying LDA. In addition, we implement and evaluate both LDA and hSBM models using multiple performance metrics and benchmarking.

Contents

Chapter 1

Introduction and Historical Review

1.1 Introduction

Vast amounts of textual data—ranging from scientific literature to social media streams—demand robust methods for uncovering hidden patterns and thematic structures. A central challenge in this domain is to transform raw, unstructured text into interpretable insights that can inform decision-making and foster deeper understanding. One of the seminal breakthroughs in this area is *Latent Dirichlet Allocation* (LDA) (Blei et al., 2003), a generative probabilistic model that has revolutionized topic modeling by representing a document as a mixture of latent topics, with each topic defined as a distribution over words.

At its core, LDA elegantly marries the principles of Bayesian statistics with the practical demands of large-scale text analysis. For instance, when analyzing a corpus of research articles, LDA can automatically reveal clusters of related topics—such as "machine learning," "data mining," and "statistical inference"—each characterized by distinctive vocabularies. These latent topics are not merely abstract constructs; they align closely with human intuition about thematic content and provide a basis for organizing and summarizing vast collections of documents.

While LDA has been widely adopted in natural language processing and data analysis, its significance extends beyond its immediate application. The model's reliance on a well-defined probabilistic structure facilitates the use of

rigorous inference techniques, such as Variational Bayes and Gibbs Sampling, to iteratively estimate hidden variables.

In contrast to many models that function as opaque black boxes, the Bayesian framework underlying LDA offers transparency and interpretability. Rather than treating models as monolithic entities, LDA emphasizes a modular understanding—where each component (e.g., the Dirichlet priors) has a clear statistical interpretation. This perspective is crucial when one seeks to balance the power of data-driven modeling with the need for theoretical insight and rigorous validation.

Building on these ideas, this thesis explores both the mathematical foundations and practical implementations of LDA. We first trace the predecessors of LDA like LSI (Deerwester et al., 1990) and pLSI (Hofmann, 1999). Then, we derive the model's formulation and examine its numerical algorithms. In addition, by mapping word–document associations onto a bipartite network, we extend the discussion to include modern community detection methods, such as the hierarchical Stochastic Block Model (hSBM) (Holland et al., 1983). The bridging between LDA and hSBM (Gerlach et al., 2018) enriches our understanding of topic models and offers new perspectives on traditional text analysis and network science.

1.2 Historical Review

The discovery and modeling of latent structures in both textual and network data have been central themes in data science over the past several decades. Early efforts in text analysis relied on algebraic and heuristic techniques, while community detection in networks was initially based on quality functions. Over time, statistical text analysis transitioned to probabilistic, generative models that provide a more principled framework for inference. In this section, we trace the evolution of these ideas—from **Latent Semantic Indexing (LSI)** and **Probabilistic Latent Semantic Indexing (pLSI)** to **Latent Dirichlet Allocation (LDA)**—and then connect them with developments in network community detection culminating in the **hierarchical Stochastic Block Model (hSBM)**.

1.2.1 Early Heuristic Approaches

Latent Semantic Indexing (LSI) was one of the first methods proposed for extracting latent semantic information from a corpus. As introduced by Deerwester et al. (1990), LSI applies singular value decomposition to the document-term matrix, reducing dimensionality and uncovering latent concepts that capture synonymy and polysemy. Despite its effectiveness in information retrieval, LSI lacks a probabilistic foundation, limiting its interpretability as a generative model and its ability to generalize to new data.

Parallel to text analysis, note that similarly in network science, early methods for community detection relied on optimizing quality functions such as heuristics proposed by M. E. J. Newman and Girvan (2004). Although modularity maximization became a popular heuristic for uncovering community structure, it suffers from issues like the resolution limit, where smaller communities may remain undetected in large networks.

1.2.2 Probabilistic Generative Models

The introduction of **Probabilistic Latent Semantic Indexing (pLSI)** by Hofmann (1999) marked a significant advance by framing topic discovery as a probabilistic process. In pLSI, each word occurrence in a document is assumed to be generated by first selecting a latent topic and then sampling a word from the corresponding topic distribution. While this model provided a clear generative story, it treated the topic mixture for each document as a set of free parameters, which led to an explosion in the number of parameters and hindered the model's ability to handle unseen documents.

To overcome these limitations, **Latent Dirichlet Allocation (LDA)** was proposed by Blei et al. (2003). LDA introduces Dirichlet priors over document–topic and topic–word distributions, thereby placing pLSI within a fully Bayesian framework. This development not only curtails overfitting by reducing the effective number of parameters but also provides a natural mechanism for inferring topics in new documents. The hierarchical Bayesian nature of LDA has since established it as a foundational model in text mining and topic modeling.

1.2.3 Extensions and Applications of LDA

Since its inception, LDA has spurred a host of extensions aimed at addressing its inherent limitations and broadening its applicability. For example, Wang and Grimson (2007) proposed Spatial LDA (SLDA), which incorporates spatial context—critical for image analysis—into the standard LDA framework. To tackle the challenges of processing massive datasets, Hoffman et al. (2010) developed an online variational Bayes algorithm (Online LDA) that leverages stochastic optimization for scalability.

Other significant advances include spectral algorithms for LDA. Anandkumar et al. (2012) introduced a spectral decomposition method based on third-order moments, guaranteeing consistent parameter recovery and offering computational efficiency. In parallel, statistical testing frameworks proposed by Lewis and Grossetti (2022) have been used to determine optimal LDA configurations, while Gutiérrez et al. (2024) revisit classical statistical techniques to enhance the robustness and interpretability of LDA, especially in conjunction with large language models. Surveys by Jelodar et al. (2018) further underscore LDA's extensive applications, ranging from software engineering to political science.

1.2.4 Community Detection and Unified Perspectives

In parallel to developments in topic modeling, the probabilistic formulation of community detection via the **Stochastic Block Model (SBM)** emerged as a counterpart to pLSI. Originating with Holland et al. (1983) and refined by Nowicki and Snijders (2001), the SBM models networks by assuming that nodes belong to latent communities and that the probability of an edge depends solely on these community assignments.

Building on this framework, **Hierarchical SBMs (hSBM)** were developed by Peixoto (2014) to reveal multilevel community structures. hSBM not only overcomes the resolution limitations of modularity but also automatically infers the number and hierarchy of communities through a Bayesian model selection process. A pivotal insight by Gerlach et al. (2018) is that when a text corpus is represented as a bipartite network (with documents and words as nodes), the problem of topic modeling becomes equivalent to that of community detection. Under this view, LDA is seen as a specific instance of a more general framework, and hSBM can be interpreted as a natural extension of LDA. This unified perspective bridges two historically separate

fields, enabling models that capture both the latent thematic structure of texts and the underlying community structure of networks.

1.2.5 Remarks

The evolution from LSI and modularity-based methods to advanced generative models such as LDA and hSBM illustrates a broader trend toward probabilistic and hierarchical approaches in data analysis. LDA revolutionized topic modeling by providing a robust Bayesian framework, while hSBM extends these ideas to uncover multiscale structure in networks and text simultaneously. The interplay between these models not only enhances our understanding of latent structures in complex data but also opens up new avenues for research that leverage the strengths of both topic modeling and community detection.

Chapter 2

Latent Semantic Indexing and Probabilistic Latent Semantic Indexing

In this chapter, we briefly study Latent Semantic Indexing (LSI) and Probabilistic Latent Semantic Indexing (pLSI) for a better understanding of the origin of Latent Dirichlet Allocation (LDA). This chapter presents the concise mathematical foundations for these methods, emphasizing the formulation and derivations, and briefly discuss their applications in information retrieval.

2.1 Latent Semantic Indexing

LSI is a linear algebraic technique for extracting latent semantic structure from a corpus of text documents proposed by Deerwester et al. (1990). Although LSI lacks a probabilistic interpretation, it has proven to be effective for information retrieval tasks by uncovering the main patterns in term usage (Ding, 2005).

2.1.1 Mathematical Derivation of LSI

Let D be the number of documents and V be the size of the vocabulary. We construct a term-document matrix

$$\mathbf{X} = [X_{ij}] \in \mathbb{R}^{V \times D},$$

where each entry X_{ij} represents a weighting (e.g., term frequency, number of times word i is in document j, or TF–IDF [1]) of term i in document j.

LSI employs Singular Value Decomposition (SVD) to factorize $\mathbf{X}$:

$$\mathbf{X} = \mathbf{U}\,\mathbf{\Sigma}\,\mathbf{V}^{\top}, \tag{2.1}$$

where:

- $\mathbf{U} \in \mathbb{R}^{V \times r}$ is the matrix of left singular vectors,
- $\mathbf{\Sigma} \in \mathbb{R}^{r \times r}$ is the diagonal matrix of singular values, and
- $\mathbf{V} \in \mathbb{R}^{D \times r}$ is the matrix of right singular vectors,

with $r = \operatorname{rank}(\mathbf{X})$.

The SVD expresses $\mathbf{X}$ as a sum of rank-1 matrices:

$$\mathbf{X} = \sum_{i=1}^{r} \sigma_i\, \mathbf{u_i}\, \mathbf{v_i}^{\top},$$

where $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \geq 0$.

Low-Rank Approximation and Latent Semantic Space

To reveal the most salient semantic structures, LSI truncates the SVD to the top K singular values, where $K < r$. Define:

- $\mathbf{U}_K \in \mathbb{R}^{V \times K}$: first K columns of $\mathbf{U}$,
- $\mathbf{\Sigma}_K \in \mathbb{R}^{K \times K}$: diagonal matrix with the top K singular values, and
- $\mathbf{V}_K \in \mathbb{R}^{D \times K}$: first K columns of $\mathbf{V}$.

[1] Term Frequency–Inverse Document Frequency, calculated as term frequency in a document × inverse document frequency across the corpus.

Then the rank-K approximation is:

$$\mathbf{X}^{(K)} = \mathbf{U}_K \, \mathbf{\Sigma}_K \, \mathbf{V}_K^\top = \sum_{i=1}^{K} \sigma_i \, u_i \, v_i^\top. \tag{2.2}$$

This approximation minimizes the Frobenius norm $\|\mathbf{X} - \mathbf{X}^{(K)}\|_F$ over all rank-K matrices (by the Eckart–Young theorem).

In the latent semantic space spanned by the columns of $\mathbf{U}_K$, each document is represented by the vector:

$$\mathbf{y}_j = \mathbf{\Sigma}_K \, (v_{1j}, v_{2j}, \ldots, v_{Kj})^\top,$$

and each term is represented by the corresponding row of $\mathbf{U}_K \, \mathbf{\Sigma}_K$. This representation enables us to analyze the latent connections in the corpus. For example, the inner product or cosine similarity between document vectors in this latent space is then used to assess document similarity.

2.1.2 Explanation for LSI

In essence, LSI factorizes the term-document matrix $\mathbf{X}$ into three components: $\mathbf{U}$, which can be intuitively viewed as capturing patterns in term usage; $\mathbf{\Sigma}$, which holds singular values that indicate the "importance of these patterns"; and $\mathbf{V}^\top$, which encodes how documents relate to these latent patterns. By truncating the SVD to the top K singular values, we filter out less significant information (often regarded as noise) and retain the most prominent semantic features.

This low-rank approximation maps high-dimensional term vectors into a K-dimensional latent semantic space where each axis represents an abstract concept or topic. For example, documents covering subjects like "biology, "chemistry," or "physics" may use different vocabularies but are mapped closely together if they share similar underlying themes. This process helps overcome the challenges of synonymy and polysemy, making it easier to compare and retrieve documents based on their underlying semantic content rather than their specific word choices.

Once documents and terms are represented in this compact latent space, similarity measures (such as cosine similarity) become more effective at capturing the true relationships between documents, enhancing tasks like document clustering and information retrieval. A simple demonstration can be

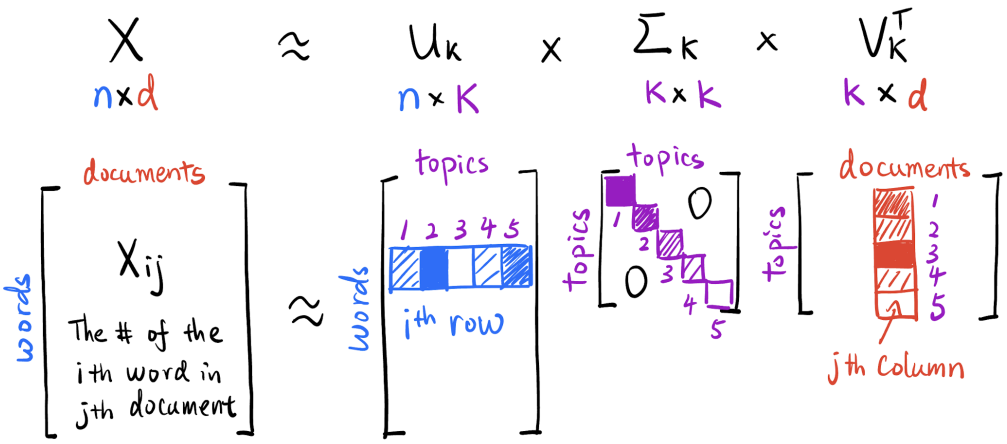

Figure 2.1: Low-rank approximation via SVD in LSI

found in Figure 2.1. In this demonstration, $K = 5$, and darker shades indicate larger entry values. Leveraging LSI, we can observe, for example, the following aspects of the corpus:

- Topic 1 is very dominant in the corpus.
- The i^{th} word is strongly associated with Topic 2, but weakly with Topic 3.
- The j^{th} word is strongly associated with Topic 3, but not with Topic 5.

2.2 Probabilistic Latent Semantic Indexing

pLSI, proposed by Hofmann (1999), provides a probabilistic framework for latent semantic analysis by modeling each word occurrence in a document as generated by a latent topic. Unlike LSI, pLSI defines a generative model which allows for statistical inference and model estimation via techniques such as the Expectation-Maximization (EM) algorithm.

In pLSI, each word occurrence in a document is assumed to be generated by an unobserved latent topic. Every document is viewed as a mixture of

these latent topics, and each topic is described by a probability distribution over words. Thus, the probability of a word appearing in a document is expressed as a weighted sum over the topics present in that document.

2.2.1 Model Formulation

Let D be the set of documents and $\mathbf{V}$ the vocabulary. For each document $d \in \{1, \ldots, D\}$ and each word $w \in \mathbf{V}$, pLSI introduces a latent topic variable $k \in \{1, \ldots, K\}$.

Define:

- $\theta_{d,k} = P(k \mid d)$: the probability that topic k is present in document d, with the sum $\sum_{k=1}^{K} \theta_{d,k} = 1$,
- $\phi_{k,w} = P(w \mid k)$: the probability of word w under topic k, with the sum $\sum_{w \in \mathbf{V}} \phi_{k,w} = 1$.

Under the model, the probability of observing a word w in document d is:

$$P(w \mid d) = \sum_{k=1}^{K} P(w \mid k) P(k \mid d) = \sum_{k=1}^{K} \theta_{d,k}\, \phi_{k,w}\,. \tag{2.3}$$

Now, we can write the joint probability of word w in document d, denoted $P(d, w)$, as

$$P(d, w) \;=\; P(d)\, P(w \mid d).$$

Here, $P(d)$ can be viewed as either a known or an estimated document prior. Substituting $P(w \mid d)$ from Equation (2.3) gives:

$$P(d, w) \;=\; P(d) \sum_{k=1}^{K} \theta_{d,k}\, \phi_{k,w}.$$

Under this model, each observed pair (d, w) is assumed to arise from an unobserved (latent) topic k. Specifically, one first selects a document d with probability $P(d)$, then chooses a topic k conditioned on d via $P(k \mid d)$, and finally draws a word w according to $P(w \mid k)$. Because the latent variable k is not observed, we sum over it, which yields the mixture-form expression for the conditional probability of a word given a document:

$$P(w \mid d) \;=\; \sum_{k=1}^{K} P(w \mid k)\, P(k \mid d) \;=\; \sum_{k=1}^{K} \theta_{d,k}\, \phi_{k,w}.$$

From the chain rule of probability, we can write the joint distribution as

$$P(d, w) \;=\; P(d)\, P(w \mid d).$$

Substituting $P(w \mid d)$ from above, we obtain

$$P(d, w) \;=\; P(d) \sum_{k=1}^{K} \theta_{d,k}\, \phi_{k,w}.$$

In the terminology used by Hofmann (1999) (especially his Equations (1) and (2)), this model is a *re-parameterized* version of the underlying generative process, sometimes referred to as the "aspect model." The key assumptions are that w is conditionally independent of d given k, and that k is a latent class variable which we marginalize out. Thus, although only the pairs (d, w) are observed, the model posits an interpretable intermediate representation in terms of latent topics k.

Next, let $n(d, w)$ be the number of times word w appears in document d. Since a document is viewed as a bag of words, a document is essentially the frequencies of all words that it has. Thus, the likelihood of the entire corpus (all documents) is

$$\prod_{d \in D} \prod_{w \in \mathbf{V}} \Big(P(d, w) \Big)^{n(d,w)}$$

assuming independence. Therefore, the log likelihood of the entire corpus is

$$\mathcal{L} \;=\; \sum_{d \in D} \sum_{w \in \mathbf{V}} n(d, w)\, \log \Big(P(d, w) \Big).$$

Substituting $P(d, w)$, we have

$$\mathcal{L} = \sum_{d \in D} \sum_{w \in \mathbf{V}} n(d, w)\, \log \left(P(d) \sum_{k=1}^{K} \theta_{d,k}\, \phi_{k,w} \right). \tag{2.4}$$

In many cases, the document prior $P(d)$ is taken as "known" (e.g., proportional to the document length).

2.2.2 Parameter Estimation via Expectation-Maximization

Direct maximization of the likelihood in (2.4) is difficult due to the latent topic variable and the inherent complexity of the joint distribution. Instead,

the EM algorithm is employed to find maximum likelihood estimates of the parameters $\theta_{d,k}$ and $\phi_{k,w}$.

In practice, we do not know the best values of θ and ϕ at the outset, so we must choose some initial values. A common approach is to initialize them randomly or use a small amount of random noise around a uniform distribution. For instance:

$$\theta_{d,k}^{(0)} \sim \text{Uniform}(0,1), \quad \phi_{k,w}^{(0)} \sim \text{Uniform}(0,1),$$

followed by normalization to ensure each distribution sums to 1. Alternatively, one can initialize using heuristics:

- *Uniform Initialization*: Set $\theta_{d,k}^{(0)} = 1/K$ for all k and $\phi_{k,w}^{(0)} = 1/|\mathbf{V}|$ for all w.
- *Term-Frequency Initialization*: Start $\phi_{k,w}^{(0)}$ in proportion to the relative frequencies of words for a randomly selected subset of documents associated with topic k.

E-Step

At iteration t, we compute the posterior probability of the latent topic k for each word occurrence (d, w) using the current estimates $\theta^{(t)}$ and $\phi^{(t)}$:

$$P^{(t)}(k \mid d, w) = \frac{\theta_{d,k}^{(t)} \, \phi_{k,w}^{(t)}}{\sum_{k'=1}^{K} \theta_{d,k'}^{(t)} \, \phi_{k',w}^{(t)}} \,. \tag{2.5}$$

After computing $P^{(t)}(k \mid d, w)$ for each (d, w), these posterior values act as the fractional counts indicating how often each topic k is responsible for each observed word w in document d.

M-Step

In the M-step, we update the parameter estimates using the expected counts from the E-step. These updates yield the new estimates $\phi^{(t+1)}$ and $\theta^{(t+1)}$:

$$\phi_{k,w}^{(t+1)} = \frac{\sum_{d \in D} n(d, w) \, P^{(t)}(k \mid d, w)}{\sum_{d \in D} \sum_{w' \in \mathbf{V}} n(d, w') \, P^{(t)}(k \mid d, w')}, \tag{2.6}$$

$$\theta_{d,k}^{(t+1)} = \frac{\sum_{w \in \mathbf{V}} n(d, w) \, P^{(t)}(k \mid d, w)}{\sum_{w \in \mathbf{V}} n(d, w)}. \tag{2.7}$$

Equation (2.6) updates the topic–word distribution $\phi_{k,\cdot}$ by normalizing the expected count of word w given topic k, while Equation (2.7) updates the document–topic distribution $\theta_{d,\cdot}$ by normalizing the expected topic counts in document d.

These E- and M-steps are iterated until convergence, ensuring that the log-likelihood does not decrease at each iteration. Regardless of the specific initialization, the EM algorithm typically converges to a local maximum of the likelihood. However, different initializations may lead to different local optima, so it is often advisable to run multiple initializations and pick the best solution (e.g., the one with the highest log-likelihood). We will explore in Chapter 3 the derivation of the E-step and M-step for pLSI.

Chapter 3

Derivation of the EM Algorithm and Its Application to pLSI

In Probabilistic Latent Semantic Indexing from the previous chapter, we observe a document–word pair (d, w), but the topic (latent variable k) that generated this pair is not observed. More generally speaking, in many estimation problems, part of the data is missing or latent. The EM (Expectation–Maximization) algorithm (Dempster et al., 1977) is a general method to obtain maximum likelihood estimates in such incomplete-data settings.

In this chapter, we first follow the proof of the EM algorithm as presented in Ng (2022)[1], and then show how to apply these principles to pLSI. Special care is taken to indicate where the observed counts in pLSI arise.

[1]Note that the proof does not contain a specific application to pLSI

3.1 Proof of the EM Algorithm

3.1.1 Setup and Notation

Let $x^{(i)} \in \mathcal{X}$ denote the observed part of the i-th data point[2], and $z^{(i)} \in \mathcal{Z}$ denote its unobserved latent component. For simplicity, assume that the complete-data likelihood factorizes as:

$$p(x^{(i)}, z^{(i)}; \theta),$$

for model parameters θ. The observed-data likelihood (also called the marginal or incomplete-data likelihood) is then given by marginalizing out the latent variable:

$$p(x^{(i)}; \theta) = \sum_{z^{(i)}} p(x^{(i)}, z^{(i)}; \theta),$$

and the total log-likelihood for m independent examples is:

$$\ell(\theta) = \sum_{i=1}^{m} \log p(x^{(i)}; \theta).$$

However, maximizing $\ell(\theta)$ directly is often computationally difficult due to the presence of the inner summation over latent variables inside the log.

3.1.2 Lower Bounding the Log-Likelihood via Jensen's Inequality

To address this issue, we use Jensen's inequality, which states that for a concave function f, and any probability distribution Q, the following holds:

$$f\left(\sum_{z} Q(z) \cdot \frac{p(z)}{Q(z)}\right) \geq \sum_{z} Q(z) \cdot f\left(\frac{p(z)}{Q(z)}\right).$$

[2]We use superscript i to index individual data points, such as $x^{(i)}$ and $z^{(i)}$, in the derivation of the general EM algorithm (Section 3.1). In contrast, we use superscript t to denote EM iteration steps when applying the algorithm to pLSI (Chapter 3). This distinction avoids confusion between the identity of data instances and the evolution of parameter estimates across iterations.

Let us introduce, for each example $x^{(i)}$, a distribution $Q_i(z^{(i)})$ over the latent variable $z^{(i)}$. Then:

$$\begin{aligned}\log p(x^{(i)};\theta) &= \log \sum_{z^{(i)}} p(x^{(i)}, z^{(i)};\theta) \\ &= \log \sum_{z^{(i)}} Q_i(z^{(i)}) \cdot \frac{p(x^{(i)}, z^{(i)};\theta)}{Q_i(z^{(i)})} \\ &\geq \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)};\theta)}{Q_i(z^{(i)})},\end{aligned}$$

where the inequality comes from Jensen's inequality applied to the concave log function.

Summing this inequality over all $i = 1, \ldots, m$, we obtain the following lower bound on the log-likelihood:

$$\ell(\theta) \geq \sum_{i=1}^{m} \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)};\theta)}{Q_i(z^{(i)})} := J(Q, \theta). \tag{3.1}$$

3.1.3 The E-Step: Tightening the Bound

The right-hand side of Eq. (3.1) depends on both Q and θ. For the lower bound to be useful, we want it to be tight at the current parameter value $\theta^{(t)}$. This can be achieved by choosing:

$$Q_i(z^{(i)}) := p(z^{(i)} \mid x^{(i)};\theta^{(t)}).$$

Indeed, with this choice:

$$Q_i(z^{(i)}) = \frac{p(x^{(i)}, z^{(i)};\theta^{(t)})}{\sum_{z'} p(x^{(i)}, z';\theta^{(t)})} = \frac{p(x^{(i)}, z^{(i)};\theta^{(t)})}{p(x^{(i)};\theta^{(t)})},$$

so the ratio inside the log becomes:

$$\log \frac{p(x^{(i)}, z^{(i)};\theta^{(t)})}{Q_i(z^{(i)})} = \log p(x^{(i)};\theta^{(t)}),$$

which is constant over $z^{(i)}$, and thus the Jensen inequality holds with equality. Therefore:

$$\ell(\theta^{(t)}) = J(Q, \theta^{(t)}).$$

This step is called the **E-step**, where we compute the posterior distribution over the latent variables given the current parameter estimate.

3.1.4 The M-Step: Maximizing the Bound

In the **M-step**, we fix the distributions $Q_i(z^{(i)}) = p(z^{(i)} \mid x^{(i)}; \theta^{(t)})$ computed in the E-step and maximize the lower bound $J(Q, \theta)$ with respect to θ. That is,

$$\theta^{(t+1)} := \arg\max_{\theta} J(Q, \theta) = \arg\max_{\theta} \sum_{i=1}^{m} \sum_{z^{(i)}} Q_i(z^{(i)}) \log p(x^{(i)}, z^{(i)}; \theta).$$

Note that since Q_i is fixed, the second term inside the log-ratio in Eq. (3.1) becomes independent of θ and drops out during the optimization.

3.1.5 Monotonic Increase in Log-Likelihood

Let us now show that EM is guaranteed not to decrease the observed-data log-likelihood. Suppose that $\theta^{(t)}$ is the current parameter value and $\theta^{(t+1)}$ is the result of applying one iteration of EM.

Then, using the tightness of the bound at $\theta^{(t)}$ and the maximization step:

$$\begin{aligned} \ell(\theta^{(t+1)}) &\geq J(Q, \theta^{(t+1)}) && \text{(lower bound)} \\ &\geq J(Q, \theta^{(t)}) && \text{(M-step maximization)} \\ &= \ell(\theta^{(t)}), && \text{(tightness at } \theta^{(t)}) \end{aligned}$$

Hence:

$$\ell(\theta^{(t+1)}) \geq \ell(\theta^{(t)}),$$

so the log-likelihood is guaranteed to increase (or stay the same) at every step.

3.1.6 Summary of the EM Algorithm

We now summarize the general EM procedure:

- **E-step:** Compute

$$Q_i(z^{(i)}) := p(z^{(i)} \mid x^{(i)}; \theta^{(t)}).$$

- **M-step:** Update

$$\theta^{(t+1)} := \arg\max_{\theta} \sum_{i=1}^{m} \sum_{z^{(i)}} Q_i(z^{(i)}) \log p(x^{(i)}, z^{(i)}; \theta).$$

Repeat these steps until convergence.

The EM algorithm can also be interpreted as coordinate ascent on the auxiliary function $J(Q, \theta)$. The E-step maximizes J with respect to Q (holding θ fixed), and the M-step maximizes with respect to θ (holding Q fixed). This guarantees convergence to a local optimum or saddle point of the likelihood.

3.2 Application to pLSI

Having established the general EM framework, we now apply it to a specific probabilistic model: *Probabilistic Latent Semantic Indexing* (pLSI) (Hofmann, 1999). This model is widely used in topic modeling for documents and builds on the idea that each observed document–word pair is generated by an unobserved topic variable.

3.2.1 The pLSI Generative Model Revisit

Let us review the model from the previous chapter. Let D be the number of documents and V the size of the vocabulary. The model assumes that each word $w \in \{1, \ldots, V\}$ in a document $d \in \{1, \ldots, D\}$ is generated via the following generative process:

1. Select a latent topic $k \in \{1, \ldots, K\}$ according to a multinomial distribution $P(k)$.
2. Given k, generate a document d according to $P(d \mid k)$.
3. Given k, generate a word w according to $P(w \mid k)$.

The joint distribution over (d, w, k) is therefore:

$$P(d, w, k) = P(k) \cdot P(d \mid k) \cdot P(w \mid k),$$

and the marginal (observed) distribution over (d, w) is:

$$P(d, w) = \sum_{k=1}^{K} P(k) \cdot P(d \mid k) \cdot P(w \mid k).$$

Suppose we are given a corpus represented as a term-document matrix with entries $n(d, w)$, which denote the number of times word w appears in document d. Our goal is to estimate the parameters $\theta = \{P(k), P(d \mid k), P(w \mid k)\}$ to maximize the likelihood of the observed counts.

3.2.2 Log-Likelihood of the Observed Data

The total log-likelihood of the corpus is:

$$\ell(\theta) = \sum_{d=1}^{D} \sum_{w=1}^{V} n(d, w) \cdot \log P(d, w),$$

where:

$$P(d, w) = \sum_{k=1}^{K} P(k) \cdot P(d \mid k) \cdot P(w \mid k).$$

Again, we face the same challenge: the log contains a sum over latent variables, making direct maximization intractable. We therefore apply the EM algorithm.

3.2.3 E-Step for pLSI

In the E-step, we compute the posterior probability of the latent topic variable k given each observed document–word pair (d, w), using the current parameter estimates $\theta^{(t)}$. This is also called the *responsibility* of topic k for the pair (d, w):

$$r_{d,w}^{(t)}(k) := p(k \mid d, w; \theta^{(t)}) = \frac{P^{(t)}(k) \cdot P^{(t)}(d \mid k) \cdot P^{(t)}(w \mid k)}{\sum_{k'=1}^{K} P^{(t)}(k') \cdot P^{(t)}(d \mid k') \cdot P^{(t)}(w \mid k')}. \quad (3.2)$$

This assigns soft cluster memberships of topics to each document–word occurrence.

3.2.4 M-Step for pLSI

In the M-step, we maximize the expected complete-data log-likelihood with respect to the model parameters, using the responsibilities $r_{d,w}^{(t)}(k)$ computed in the E-step.

The complete-data log-likelihood for a single pair (d, w) is:

$$\sum_{k=1}^{K} r_{d,w}^{(t)}(k) \cdot \log\left[P(k) \cdot P(d \mid k) \cdot P(w \mid k)\right],$$

weighted by the count $n(d, w)$. Thus, the full expected log-likelihood is:

$$Q(\theta, \theta^{(t)}) = \sum_{d=1}^{D} \sum_{w=1}^{V} n(d, w) \sum_{k=1}^{K} r_{d,w}^{(t)}(k) \cdot \log\left[P(k) \cdot P(d \mid k) \cdot P(w \mid k)\right].$$

Since the terms for $P(k)$, $P(d \mid k)$, and $P(w \mid k)$ are separable, we optimize them independently under their respective normalization constraints.

Update for $P(w \mid k)$: We maximize:

$$\sum_{d,w} n(d, w) \cdot r_{d,w}^{(t)}(k) \cdot \log P(w \mid k), \quad \text{subject to} \quad \sum_{w} P(w \mid k) = 1.$$

This is a standard constrained optimization problem, and using Lagrange multipliers, we obtain:

$$P(w \mid k) = \frac{\sum_d n(d, w) \cdot r_{d,w}^{(t)}(k)}{\sum_{d,w'} n(d, w') \cdot r_{d,w'}^{(t)}(k)}.$$

Update for $P(d \mid k)$:

$$P(d \mid k) = \frac{\sum_w n(d, w) \cdot r_{d,w}^{(t)}(k)}{\sum_{d',w} n(d', w) \cdot r_{d',w}^{(t)}(k)}.$$

Update for $P(k)$:

$$P(k) = \frac{\sum_{d,w} n(d, w) \cdot r_{d,w}^{(t)}(k)}{\sum_{d,w} n(d, w)}.$$

Each of these updates has an intuitive interpretation: the numerator counts the expected number of times topic k is responsible for an event (e.g., generating word w), and the denominator normalizes these expected counts into valid probability distributions.

3.2.5 Summary of the EM Algorithm for pLSI

To summarize, the EM algorithm for pLSI proceeds as follows:

- **E-step:** For each (d, w), compute $r_{d,w}^{(t)}(k)$ using Eq. (3.2).
- **M-step:** Update the parameters using the weighted counts:

$$P(w \mid k) = \frac{\sum_d n(d,w) \cdot r_{d,w}^{(t)}(k)}{\sum_{d,w'} n(d,w') \cdot r_{d,w'}^{(t)}(k)},$$

$$P(d \mid k) = \frac{\sum_w n(d,w) \cdot r_{d,w}^{(t)}(k)}{\sum_{d',w} n(d',w) \cdot r_{d',w}^{(t)}(k)},$$

$$P(k) = \frac{\sum_{d,w} n(d,w) \cdot r_{d,w}^{(t)}(k)}{\sum_{d,w} n(d,w)}.$$

Because pLSI treats documents as observed rather than drawn from a document-generating distribution, it lacks a generative model for new documents. Later extensions such as Latent Dirichlet Allocation (LDA) resolve this limitation by introducing document-level priors.

3.3 Understanding Lagrange Multipliers in the EM Derivation

In the M-step of the EM algorithm for pLSI, we maximize the expected complete-data log-likelihood with respect to the model parameters. Because these parameters are probability distributions, they must satisfy normalization constraints. To perform such constrained optimizations, we use the method of *Lagrange multipliers*, a classical technique in multivariate calculus.

3.3.1 Mathematical Background

Suppose we want to maximize a scalar-valued function $F(\mathbf{x})$, subject to the constraint:

$$g(\mathbf{x}) = c,$$

where $\mathbf{x} \in \mathbb{R}^n$. In the unconstrained case, we would set the gradient $\nabla F(\mathbf{x}) = 0$ to locate an optimum. But when constraints are present, we need to enforce that any optimum must lie on the feasible set $\{\mathbf{x} : g(\mathbf{x}) = c\}$.

To solve this, we introduce a new variable $\lambda \in \mathbb{R}$ (the *Lagrange multiplier*) and define the *Lagrangian*:

$$\mathcal{L}(\mathbf{x}, \lambda) = F(\mathbf{x}) + \lambda \cdot (c - g(\mathbf{x})).$$

The necessary conditions for optimality become:

$$\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \lambda) = 0, \quad \text{and} \quad g(\mathbf{x}) = c.$$

These equations jointly enforce both optimality and feasibility. See Boyd and Vandenberghe (2004) for an in-depth treatment.

3.3.2 Application to Updating $P(w \mid k)$ in pLSI

Let us now apply this technique to the update of $P(w \mid k)$ during the M-step of EM for pLSI. We are maximizing the expected log-likelihood term associated with $P(w \mid k)$, which, for a fixed topic k, is:

$$Q_k(P(w \mid k)) = \sum_{d,w} n(d, w) \cdot r_{d,w}(k) \cdot \log P(w \mid k),$$

subject to the normalization constraint:

$$\sum_{w} P(w \mid k) = 1.$$

Step 1: Form the Lagrangian. We introduce a Lagrange multiplier λ and define:

$$\mathcal{L}\big(P(w \mid k), \lambda\big) = \sum_{d,w} n(d, w) \cdot r_{d,w}(k) \cdot \log P(w \mid k) + \lambda \cdot \left(1 - \sum_{w} P(w \mid k)\right).$$

Step 2: Take derivatives. We differentiate with respect to $P(w \mid k)$, treating each w as a separate variable:

$$\frac{\partial \mathcal{L}}{\partial P(w \mid k)} = \sum_{d} \frac{n(d, w) \cdot r_{d,w}(k)}{P(w \mid k)} - \lambda.$$

Setting the derivative to zero, we get:

$$P(w \mid k) = \frac{\sum_d n(d, w) \cdot r_{d,w}(k)}{\lambda}.$$

Step 3: Enforce the constraint. Using $\sum_w P(w \mid k) = 1$, we solve for λ:

$$\sum_w P(w \mid k) = \frac{1}{\lambda} \sum_{w,d} n(d,w) \cdot r_{d,w}(k) = 1 \quad \Longrightarrow \quad \lambda = \sum_{d,w} n(d,w) \cdot r_{d,w}(k).$$

Step 4: Final update rule. Substituting back, we obtain:

$$P(w \mid k) = \frac{\sum_d n(d,w) \cdot r_{d,w}(k)}{\sum_{d',w'} n(d',w') \cdot r_{d',w'}(k)}.$$

This update has a natural probabilistic interpretation: the numerator represents the expected count of word w being associated with topic k, while the denominator is the total expected count of all words associated with topic k. The method of Lagrange multipliers ensures that this update maximizes the objective while maintaining valid probability constraints. Identical steps can be used to derive the update rules for $P(d \mid k)$ and $P(k)$.

Chapter 4

Derivation of Latent Dirichlet Allocation

Notations

Below, we introduce the key notations used in this derivation of Latent Dirichlet Allocation (LDA):

- $\mathbf{V}$: the vocabulary, i.e., the set of all possible words; denote $|\mathbf{V}|$ as its size.
- $\mathbf{Z} = \{1, 2, \ldots, K\}$: the set (or index set) of topics, where $K \in \mathbb{N}$ is the total number of topics.
- $D \in \mathbb{N}$: the total number of documents; we index documents by $d \in \{1, 2, \ldots, D\}$.
- $N_d \in \mathbb{N}$: the number of words in document d (if every document has the same length, one may denote this simply by N).
- $\theta_d = \langle \theta_{d,1}, \theta_{d,2}, \ldots, \theta_{d,K} \rangle$ [1]: the topic-proportion vector for document d; each $\theta_{d,k}$ represents the probability of topic k in document d, so that $\sum_{k=1}^{K} \theta_{d,k} = 1$ (i.e., θ_d lies in the $(K-1)$-simplex).
- $\phi_k = \langle \phi_{k,1}, \phi_{k,2}, \ldots, \phi_{k,|\mathbf{V}|} \rangle$: the word distribution for topic k; here, $\sum_{v=1}^{|\mathbf{V}|} \phi_{k,v} = 1$.

[1] Note that the notation $\langle \cdot \rangle$ is used here instead of $\{ \cdot \}$ for θ_d and ϕ_k to indicate that these objects are treated as ordered arrays (i.e., matrices) rather than as sets. Consequently, the order of rows and columns remains fixed during computations.

- $z_{d,n} \in \{1, 2, \ldots, K\}$: the latent topic assignment for the n-th word in document d.
- $w_{d,n} \in \mathbf{V}$: the observed word in position n of document d.
- $Z = \{z_{d,n}\}$, $W = \{w_{d,n}\}$, $\Theta = \{\theta_d\}$, $\Phi = \{\phi_k\}$.
  Note: these collections record multiplicities. Here, $\Theta = \{\theta_d\}$, $\Phi = \{\phi_k\}$ can be viewed as matrices since they are collections of vectors. Although $Z = \{z_{d,n}\}$, $W = \{w_{d,n}\}$ can also be viewed as matrices, it might be easier to just think them as collections where their internal order is not important.
- $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_K)$: hyperparameters for the Dirichlet prior on the document-topic distributions θ_d.
- $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_{|\mathbf{V}|})$: hyperparameters for the Dirichlet prior on the topic-word distributions ϕ_k.

Generative Process

The data generative process on which Latent Dirichlet Allocation is built consists of the following steps:

1. **For each topic** $k \in \{1, \ldots, K\}$:
   (a) Draw the topic-word distribution:
   $$\phi_k \sim \text{Dir}(\boldsymbol{\beta}).$$
2. **For each document** $d \in \{1, \ldots, D\}$:
   (a) Draw the document-specific topic proportions:
   $$\theta_d \sim \text{Dir}(\boldsymbol{\alpha}).$$
   (b) **For each word** $n \in \{1, \ldots, N_d\}$:
      i. Draw a single topic assignment:
      $$z_{d,n} \sim \text{Categorical}(\theta_d)^2.$$
      ii. Draw a single word:
      $$w_{d,n} \sim \text{Categorical}(\phi_{z_{d,n}}).$$

[2]The categorical distribution generalizes the Bernoulli distribution to more than two possible outcomes but still one value will be drawn.

4.1 The Joint Distribution

The following derivation (as well as the derivation for the Gibbs sampler) is adapted from Mukherjee (2003) with some modifications and updates. In this section, we derive the joint probability of the observed words and latent topic assignments and then show how we can integrate out the parameters to obtain this joint likelihood. This derivation is foundational for later inference procedures (e.g., Gibbs sampling) and for understanding the relationship between the joint distribution and the posterior distributions.

Recall that:

- $W = \{w_{d,n} : d = 1, \ldots, D;\ n = 1, \ldots, N_d\}$ is the collection of observed words.
- $Z = \{z_{d,n} : d = 1, \ldots, D;\ n = 1, \ldots, N_d\}$ is the collection of latent topic assignments.
- $\Theta = \{\theta_d : d = 1, \ldots, D\}$ is the set of document–topic distributions.
- $\Phi = \{\phi_k : k = 1, \ldots, K\}$ is the set of topic–word distributions.

Given the hyperparameters $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K)$ and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_{|\mathbf{V}|}$, the joint distribution over W and Z is expressed as

$$P(W, Z; \alpha, \beta) = P(W \mid Z)\, P(Z) \equiv I_1 \times I_2. \tag{4.1}$$

Here, we define

$$I_1 = P(W \mid Z) = \int P(W \mid Z, \Phi)\, P(\Phi)\, d\Phi, \tag{4.2}$$

and

$$I_2 = P(Z) = \int P(Z \mid \Theta)\, P(\Theta)\, d\Theta. \tag{4.3}$$

This factorization clearly separates the contribution from the topic–word distributions Φ and the document–topic proportions Θ.

In the above factorization, we assume that:

1. The documents are conditionally independent given their respective topic distributions θ_d. This allows us to factorize $P(Z \mid \Theta)$ over documents.

2. The words within each document are independent given θ_d (the bag-of-words assumption) and similarly, the words are conditionally independent given the topic–word distributions ϕ_k.

These independence assumptions are standard in topic models and are crucial for tractable inference.

4.1.1 Derivation of $I_2 = P(Z)$

We begin by marginalizing out the document–topic distributions Θ. For each document d, the prior over its topic distribution is given by a Dirichlet:

$$P(\theta_d \mid \alpha) = \frac{1}{B(\alpha)} \prod_{k=1}^{K} \theta_{d,k}^{\alpha_k - 1}, \tag{4.4}$$

with the normalization constant

$$B(\alpha) = \frac{\prod_{k=1}^{K} \Gamma(\alpha_k)}{\Gamma\left(\sum_{k=1}^{K} \alpha_k\right)}. \tag{4.5}$$

Given θ_d, each word in document d is assigned a topic according to a categorical distribution:

$$P(z_{d,n} \mid \theta_d) = \theta_{d,z_{d,n}}. \tag{4.6}$$

Thus, for document d with N_d words, the likelihood of its topic assignments is

$$P(Z_d \mid \theta_d) = \prod_{n=1}^{N_d} \theta_{d,z_{d,n}}. \tag{4.7}$$

It is convenient to aggregate these assignments by defining the count

$$C(d,k) = \sum_{n=1}^{N_d} \mathbf{1}\{z_{d,n} = k\}, \tag{4.8}$$

which enables us to write

$$P(Z_d \mid \theta_d) = \prod_{k=1}^{K} \theta_{d,k}^{C(d,k)}. \tag{4.9}$$

Here we use the independence of words in a document (given θ_d) to express the joint probability as a product over word positions. Assuming independence across documents, the joint likelihood for all documents is

$$P(Z \mid \Theta) = \prod_{d=1}^{D} \prod_{k=1}^{K} \theta_{d,k}^{C(d,k)}. \tag{4.10}$$

Marginalizing over Θ, we obtain

$$I_2 = P(Z) = \prod_{d=1}^{D} \left[\frac{1}{B(\alpha)} \int_{\Delta_{K-1}} \prod_{k=1}^{K} \theta_{d,k}^{\alpha_k - 1 + C(d,k)} \, d\theta_d \right]. \tag{4.11}$$

By the Dirichlet integral identity,

$$\int_{\Delta_{K-1}} \prod_{k=1}^{K} \theta_{d,k}^{\alpha_k - 1 + C(d,k)} \, d\theta_d = B(\alpha + C_d), \tag{4.12}$$

with $\alpha + C_d = (\alpha_1 + C(d,1), \ldots, \alpha_K + C(d,K))$. Therefore, we have

$$I_2 = \prod_{d=1}^{D} \frac{B(\alpha + C_d)}{B(\alpha)}. \tag{4.13}$$

By integrating out the document–topic distributions, we capture the uncertainty about the true topic proportions in each document while relying on the observed topic assignments. This process reduces the number of parameters and yields a more compact representation of the model evidence.

4.1.2 Derivation of $I_1 = P(W \mid Z)$

Next, we marginalize out the topic–word distributions Φ. For each topic k, the prior is given by:

$$P(\phi_k \mid \beta) = \frac{1}{B(\beta)} \prod_{v=1}^{|\mathbf{V}|} \phi_{k,v}^{\beta_v - 1}, \tag{4.14}$$

with the normalization constant

$$B(\beta) = \frac{\prod_{v=1}^{|\mathbf{V}|} \Gamma(\beta_v)}{\Gamma\left(\sum_{v=1}^{|\mathbf{V}|} \beta_v\right)}. \tag{4.15}$$

Given ϕ_k, the probability of generating a word $w_{d,n}$ when $z_{d,n} = k$ is

$$P(w_{d,n} \mid z_{d,n} = k, \Phi) = \phi_{k,w_{d,n}}. \tag{4.16}$$

Defining the count

$$C(k, v) = \sum_{d=1}^{D} \sum_{n=1}^{N_d} \mathbf{1}\{w_{d,n} = v \text{ and } z_{d,n} = k\}, \tag{4.17}$$

we can express the likelihood of the observed words given Z and Φ as

$$P(W \mid Z, \Phi) = \prod_{k=1}^{K} \prod_{v=1}^{|\mathbf{V}|} \phi_{k,v}^{C(k,v)}. \tag{4.18}$$

Here, we assume that word occurrences are conditionally independent given the topic assignment and the topic–word distribution, which lets us factor the likelihood as a product over topics and vocabulary entries. Marginalizing over Φ yields

$$I_1 = P(W \mid Z) = \prod_{k=1}^{K} \left[\frac{1}{B(\beta)} \int_{\Delta_{|\mathbf{V}|-1}} \prod_{v=1}^{|\mathbf{V}|} \phi_{k,v}^{C(k,v)+\beta_v - 1} \, d\phi_k \right]. \tag{4.19}$$

By the Dirichlet integral identity,

$$\int_{\Delta_{|\mathbf{V}|-1}} \prod_{v=1}^{|\mathbf{V}|} \phi_{k,v}^{C(k,v)+\beta_v - 1} \, d\phi_k = B(\beta + C_k), \tag{4.20}$$

where $\beta + C_k = (\beta_1 + C(k, 1), \ldots, \beta_{|\mathbf{V}|} + C(k, |\mathbf{V}|))$. Thus, we have

$$I_1 = \prod_{k=1}^{K} \frac{B(\beta + C_k)}{B(\beta)}. \tag{4.21}$$

This step integrates out the topic–word distributions, accounting for uncertainty in the word probabilities for each topic while conditioning on the observed word counts. It also helps in penalizing overly complex topic models by incorporating the prior information through β.

4.1.3 Full Joint Distribution

Substituting equations (4.13) and (4.21) into equation (4.1) yields the full joint distribution:

$$P(W, Z; \alpha, \beta) = \left[\prod_{d=1}^{D} \frac{B(\alpha + C_d)}{B(\alpha)}\right] \left[\prod_{k=1}^{K} \frac{B(\beta + C_k)}{B(\beta)}\right]. \tag{4.22}$$

This full joint distribution captures the entire generative process of the corpus by combining the uncertainty from both the document–topic and topic–word distributions. This representation is essential for posterior inference and model comparison.

4.2 The Posterior on θ and ϕ

The goal of deriving the posterior distributions is to update our beliefs about the latent parameters (θ_d and ϕ_k) after observing the data W and Z. The posterior connects to the joint distribution via Bayes' rule and forms the basis for inference algorithms such as Gibbs sampling.

4.2.1 Posterior Distribution for θ_d

For a given document d, the prior is

$$\theta_d \sim \mathrm{Dir}(\alpha) \quad \text{with} \quad P(\theta_d) = \frac{1}{B(\alpha)} \prod_{k=1}^{K} \theta_{d,k}^{\alpha_k - 1}. \tag{4.23}$$

Given the topic assignments $Z_d = \{z_{d,n} : n = 1, \ldots, N_d\}$, we have

$$P(Z_d \mid \theta_d) = \prod_{k=1}^{K} \theta_{d,k}^{C(d,k)}. \tag{4.24}$$

By Bayes' rule, the posterior is proportional to

$$P(\theta_d \mid Z_d) \propto P(Z_d \mid \theta_d)\, P(\theta_d) \propto \prod_{k=1}^{K} \theta_{d,k}^{C(d,k)+\alpha_k - 1}. \tag{4.25}$$

Thus, the posterior distribution is

$$\theta_d \mid Z_d \sim \text{Dir}(\alpha + C_d). \tag{4.26}$$

Deriving the posterior for θ_d allows us to combine our prior beliefs (encoded by α) with the observed[3] topic counts $C(d, k)$. This results in a smoothed estimate that reflects both the data and our prior assumptions. Note that in this context, the term "observed topic counts" does not imply that topic assignments are directly measured from the data. Instead, it refers to the counts obtained from the inferred (or imputed) latent topic assignments—via inference methods such as Gibbs sampling—based on the observed words. These counts are treated as if they were observed for the purpose of updating our posterior.

4.2.2 Posterior Distribution for ϕ_k

For a given topic k, the prior is

$$\phi_k \sim \text{Dir}(\beta) \quad \text{with} \quad P(\phi_k) = \frac{1}{B(\beta)} \prod_{v=1}^{|\mathbf{V}|} \phi_{k,v}^{\beta_v - 1}. \tag{4.27}$$

Let W_k denote the set of all words assigned to topic k (i.e., $W_k = \{w_{d,n} : z_{d,n} = k\}$). The likelihood for the words in W_k is

$$P(W_k \mid \phi_k) = \prod_{v=1}^{|\mathbf{V}|} \phi_{k,v}^{C(k,v)}. \tag{4.28}$$

Applying Bayes' rule, we have

$$P(\phi_k \mid W_k) \propto P(W_k \mid \phi_k)\, P(\phi_k) \propto \prod_{v=1}^{|\mathbf{V}|} \phi_{k,v}^{C(k,v)+\beta_v - 1}. \tag{4.29}$$

Thus, the posterior for ϕ_k is

$$\phi_k \mid W_k \sim \text{Dir}(\beta + C_k). \tag{4.30}$$

[3]Deriving the posterior for θ_d allows us to combine our prior beliefs (encoded by α) with the observed topic counts $C(d, k)$, which results in a smoothed estimate that reflects both the data and our prior assumptions.

The posterior for ϕ_k reflects how the observed word counts for each topic update our initial beliefs (given by β) regarding the probability distribution over the vocabulary. It is essential for estimating the most likely word distributions under each topic.

4.2.3 Expected Values of the Posterior Distributions

For a Dirichlet-distributed random variable $X \sim \mathrm{Dir}(\alpha_1, \ldots, \alpha_K)$, the expected value of its i-th component is

$$\mathbb{E}[X_i] = \frac{\alpha_i}{\sum_{j=1}^{K} \alpha_j}. \tag{4.31}$$

Thus, for the document–topic distribution,

$$\mathbb{E}[\theta_{d,k}] = \frac{\alpha_k + C(d,k)}{\sum_{j=1}^{K} \Big(\alpha_j + C(d,j)\Big)}. \tag{4.32}$$

And for the topic–word distribution,

$$\mathbb{E}[\phi_{k,v}] = \frac{\beta_v + C(k,v)}{\sum_{u=1}^{|\mathbf{V}|} \Big(\beta_u + C(k,u)\Big)}. \tag{4.33}$$

These expected values provide smoothed estimates of the first moment of the latent distributions. They balance the contribution of the observed data (through the counts $C(d,k)$ and $C(k,v)$) with the prior information (α and β), yielding robust estimates that are less sensitive to data sparsity.

Chapter 5

Implementation of Latent Dirichlet Allocation

5.1 Gibbs Sampling: A Markov Chain Monte Carlo Approach

Markov Chain Monte Carlo (MCMC) methods are a class of algorithms for generating samples from complex probability distributions when direct sampling is impractical. The overarching goal of MCMC is to construct a Markov chain whose stationary distribution is the target distribution $\pi(\mathbf{x})$. By simulating the chain for a sufficiently large number of steps, one obtains a sequence of correlated samples $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots$, which (under mild conditions) converge to having $\pi(\mathbf{x})$ as their marginal distribution.

Detailed Balance and Transition Probability. A convenient way to ensure $\pi(\mathbf{x})$ is invariant under the Markov chain is via the *detailed balance* condition, also known as *reversibility*. If $P(\mathbf{x} \to \mathbf{x}')$ denotes the probability of transitioning from state $\mathbf{x}$ to $\mathbf{x}'$ in a single Markov chain step, detailed balance requires:

$$\pi(\mathbf{x})\, P(\mathbf{x} \to \mathbf{x}') \;=\; \pi(\mathbf{x}')\, P(\mathbf{x}' \to \mathbf{x}). \tag{5.1}$$

When (5.1) is satisfied for all $\mathbf{x}, \mathbf{x}'$, it implies that $\pi(\mathbf{x})$ is a stationary distribution of the chain. Famous MCMC algorithms such as the Metropolis algorithm (Metropolis et al., 1953) and Metropolis–Hastings (Hastings, 1970) are designed to satisfy (5.1) through acceptance-rejection steps.

The chief advantage of MCMC is its asymptotic correctness: given enough samples, it can approximate the true posterior as closely as desired (up to Monte Carlo error). This yields high accuracy and the ability to quantify uncertainty (e.g., via posterior samples of θ or Z). However, MCMC can be computationally expensive. Each iteration of Gibbs sampling for LDA requires processing every word token to update its topic, and many iterations may be needed for the chain to sufficiently mix. For large document collections, this becomes slow and memory-intensive. Moreover, diagnosing convergence is non-trivial—one may need to run multiple chains or use statistical diagnostics to ensure the sampler has converged. In practice, choices like the number of burn-in iterations and thinning (intervals between recorded samples) are important hyperparameters for MCMC. Poorly tuned sampling can lead to biased results or high autocorrelation in the samples.

5.1.1 Gibbs Sampling

Gibbs sampling (Casella & George, 1992; Gelfand & Smith, 1990; Geman & Geman, 1984) is a specialized MCMC algorithm that updates only one component (or subset of components) of the state vector at a time. Let $\mathbf{x} = (x_1, x_2, \ldots, x_D)$ be a D-dimensional state. Gibbs sampling proceeds by iteratively cycling through the coordinates $x_1, x_2, \ldots, x_D$. Suppose we want to update the j-th coordinate while keeping all others fixed at their current values $\mathbf{x}_{-j}$. The new value of x_j is drawn from the conditional distribution:

$$x_j' \;\sim\; \pi\big(x_j \,\big|\, \mathbf{x}_{-j}\big),$$

where $\pi(x_j \mid \mathbf{x}_{-j})$ is the probability of x_j given all other coordinates.

In terms of the transition probability notation $P(\mathbf{x} \to \mathbf{x}')$, we have:

$$P\big(\mathbf{x} \to \mathbf{x}'\big) \;=\; \begin{cases} \pi\big(x_j' \,\big|\, \mathbf{x}_{-j}\big), & \text{if } \mathbf{x}' \text{ differs from } \mathbf{x} \text{ only in the } j\text{-th component,} \\ 0, & \text{otherwise.} \end{cases}$$

The chain constructed in this manner can be shown to satisfy detailed balance with respect to $\pi(\mathbf{x})$, and therefore $\pi(\mathbf{x})$ is stationary under the Gibbs updates. We illustrate this more explicitly below.

5.1.2 Detailed Derivation of the Gibbs Sampler's Detailed Balance

The fact that the Gibbs sampler satisfies the detailed balance condition is a well-established result in the Markov Chain Monte Carlo (MCMC) literature. The following derivation expands upon the proof presented in the lecture notes by Maneesh Sahani (2015) to enhance clarity and accessibility.

Consider two states $\mathbf{x}$ and $\mathbf{x}'$ that differ only in the j-th coordinate:

$$\mathbf{x}_{-j} = \mathbf{x}'_{-j}, \quad x_j \neq x'_j.$$

By definition of the Gibbs sampler,

$$P(\mathbf{x} \to \mathbf{x}') \;=\; \pi\big(x'_j \,\big|\, \mathbf{x}_{-j}\big), \quad P(\mathbf{x}' \to \mathbf{x}) \;=\; \pi\big(x_j \,\big|\, \mathbf{x}'_{-j}\big).$$

Since $\mathbf{x}'_{-j} = \mathbf{x}_{-j}$, we have

$$P(\mathbf{x}' \to \mathbf{x}) \;=\; \pi\big(x_j \,\big|\, \mathbf{x}_{-j}\big).$$

To verify detailed balance, we need:

$$\pi(\mathbf{x})\, P(\mathbf{x} \to \mathbf{x}') \;=\; \pi(\mathbf{x}')\, P(\mathbf{x}' \to \mathbf{x}).$$

Observe that

$$\pi\big(x_j \,\big|\, \mathbf{x}_{-j}\big) \;=\; \frac{\pi\big(x_j, \mathbf{x}_{-j}\big)}{\pi(\mathbf{x}_{-j})}, \quad \pi\big(x'_j, \mathbf{x}_{-j}\big) \;=\; \pi(\mathbf{x}'), \quad \pi\big(x_j, \mathbf{x}_{-j}\big) \;=\; \pi(\mathbf{x}).$$

Hence,

$$\pi(\mathbf{x})\, P(\mathbf{x} \to \mathbf{x}') \;=\; \pi(\mathbf{x})\, \pi\big(x'_j \,\big|\, \mathbf{x}_{-j}\big) \;=$$

$$\pi(\mathbf{x})\, \frac{\pi\big(x'_j, \mathbf{x}_{-j}\big)}{\pi(\mathbf{x}_{-j})} \;=\; \frac{\pi\big(x_j, \mathbf{x}_{-j}\big)\, \pi\big(x'_j, \mathbf{x}_{-j}\big)}{\pi(\mathbf{x}_{-j})}.$$

Similarly,

$$\pi(\mathbf{x}')\, P(\mathbf{x}' \to \mathbf{x}) \;=\; \pi(\mathbf{x}')\, \pi\big(x_j \,\big|\, \mathbf{x}_{-j}\big) \;=$$

$$\pi\big(x'_j, \mathbf{x}_{-j}\big)\, \frac{\pi\big(x_j, \mathbf{x}_{-j}\big)}{\pi(\mathbf{x}_{-j})} \;=\; \frac{\pi\big(x'_j, \mathbf{x}_{-j}\big)\, \pi\big(x_j, \mathbf{x}_{-j}\big)}{\pi(\mathbf{x}_{-j})}.$$

Because multiplication is commutative,

$$\pi\big(x_j, \mathbf{x}_{-j}\big)\, \pi\big(x'_j, \mathbf{x}_{-j}\big) \;=\; \pi\big(x'_j, \mathbf{x}_{-j}\big)\, \pi\big(x_j, \mathbf{x}_{-j}\big),$$

so the two expressions are indeed equal:

$$\pi(\mathbf{x})\, P(\mathbf{x} \to \mathbf{x}') \;=\; \pi(\mathbf{x}')\, P(\mathbf{x}' \to \mathbf{x}).$$

Thus, the Gibbs sampler satisfies detailed balance, ensuring $\pi(\mathbf{x})$ is stationary. Under typical conditions (e.g., irreducibility and aperiodicity), the Markov chain will converge to $\pi(\mathbf{x})$ from almost any starting state.

5.1.3 Gibbs Sampling Conditional Derivation

Gibbs sampling is an iterative Markov Chain Monte Carlo (MCMC) algorithm used in LDA to approximate the posterior distribution over the latent topic assignments. In this derivation (adapted from Mukherjee (2003) with modifications), we derive the conditional probability for assigning topic k' to the ith word (at position j in document d [1]), given the topic assignments of all other words and the observed words. At each iteration, the counts $C(d, k)$ for documents and $C(k, v)$ for topics are updated, and these counts determine the posterior distributions

$$\theta_d \sim \mathrm{Dir}(\boldsymbol{\alpha} + C_d) \quad \text{and} \quad \phi_k \sim \mathrm{Dir}(\boldsymbol{\beta} + C_k).$$

Over successive iterations, the Gibbs sampler converges to a set of topic assignments that (approximately) maximize the likelihood of the data.

Let

$$Z = \{z_{d,n} : d = 1, \ldots, D;\ n = 1, \ldots, N_d\}$$

be the collection of latent topic assignments for all words in the corpus, where N_d is the number of words in document d. For convenience, we can also represent these assignments as a single-indexed vector:

$$Z = \{z_1, z_2, \ldots, z_N\},$$

with $N = \sum_{d=1}^{D} N_d$. We denote by Z_{-i} the set of all topic assignments except for the ith word (i.e., the word at position j in document d). Similarly, let

$$W = \{w_1, w_2, \ldots, w_N\}$$

[1] For clarity and consistency, we denote the document containing the ith word as d rather than d'. If using a prime notation to indicate a specific document is desired, one must explain that it differentiates the document in question from others. Here, we choose the simpler notation d throughout.

be the set of all observed words, and let W_{-i} denote the words excluding w_i.

We now wish to derive the conditional probability

$$P(z_i = k' \mid Z_{-i}, W). \tag{5.2}$$

Using the definition of conditional probability, we have

$$P(z_i = k' \mid Z_{-i}, W) = \frac{P(z_i = k', Z_{-i}, W)}{P(Z_{-i}, W)}. \tag{5.3}$$

Since the full assignment Z can be written as the union $\{z_i, Z_{-i}\}$, this relation becomes

$$P(z_i = k' \mid Z_{-i}, W) = \frac{P(Z, W)}{P(Z_{-i}, W)}. \tag{5.4}$$

In the above, the equality holds exactly. However, for the purpose of the Gibbs sampling update, we are interested only in the terms that depend on the candidate topic k'. Since the denominator $P(Z_{-i}, W)$ does not depend on k', we can write

$$P(z_i = k' \mid Z_{-i}, W) \propto \frac{P(Z, W)}{P(Z_{-i}, W)}. \tag{5.5}$$

Before proceeding, note that the joint distribution for the LDA model (as derived previously) is given by in 4.22

$$P(W, Z; \alpha, \beta) = \left(\frac{1}{B(\alpha)} \prod_{d=1}^{D} B(\alpha + C_d) \right) \left(\frac{1}{B(\beta)} \prod_{k=1}^{K} B(\beta + C_k) \right), \tag{5.6}$$

where:

- $C_d = [C(d, 1), C(d, 2), \ldots, C(d, K)]$ is the count vector for document d (i.e., the number of words assigned to each topic),
- $C_k = [C(k, 1), C(k, 2), \ldots, C(k, |V|)]$ is the count vector for topic k (i.e., the counts of each word in topic k), and $|V|$ is the vocabulary size.

Removing the ith word w_i (which appears in document d and has vocabulary index v') leads to updated counts denoted with a subscript $-i$. Hence, Equation (5.5) can be rewritten as

$$P(z_i = k' \mid Z_{-i}, W) \propto \frac{P(W, Z; \alpha, \beta)}{P(W_{-i}, Z_{-i}; \alpha, \beta)}. \tag{5.7}$$

Since the removal of w_i affects only the counts associated with document d and topic k', the ratio in Equation (5.7) factors into two independent components:

$$\frac{P(W,Z;\alpha,\beta)}{P(W_{-i},Z_{-i};\alpha,\beta)} = \underbrace{\frac{\prod_{d=1}^{D} B(\alpha + C_d)}{\prod_{d=1}^{D} B(\alpha + [C_d]_{-i})}}_{\text{Document part}} \times \underbrace{\frac{\prod_{k=1}^{K} B(\beta + C_k)}{\prod_{k=1}^{K} B(\beta + [C_k]_{-i})}}_{\text{Topic–word part}}. \tag{5.8}$$

Since only document d (the one containing w_i) is affected by the removal, the document part simplifies to

$$\frac{\prod_{d=1}^{D} B(\alpha + C_d)}{\prod_{d=1}^{D} B(\alpha + [C_d]_{-i})} = \frac{B(\alpha + C_d)}{B(\alpha + [C_d]_{-i})}. \tag{5.9}$$

Similarly, for the topic–word counts, only topic k' (the topic assigned to w_i) is affected, so that

$$\frac{\prod_{k=1}^{K} B(\beta + C_k)}{\prod_{k=1}^{K} B(\beta + [C_k]_{-i})} = \frac{B(\beta + C_{k=k'})}{B(\beta + [C_{k=k'}]_{-i})}. \tag{5.10}$$

Recall that the normalization constant for the Dirichlet distribution is defined as

$$B(\gamma) = \frac{\prod_{i=1}^{n} \Gamma(\gamma_i)}{\Gamma\left(\sum_{i=1}^{n} \gamma_i\right)}. \tag{5.11}$$

Document Part: Focus on the document part first. For document d, the count for topic k is updated upon the removal of w_i as follows:

$$C(d,k) = \begin{cases} [C(d,k)]_{-i} + 1, & \text{if } k = k', \\ [C(d,k)]_{-i}, & \text{if } k \neq k'. \end{cases} \tag{5.12}$$

Using the property $\Gamma(x+1) = x\,\Gamma(x)$, one can show that

$$\frac{B(\alpha + C_d)}{B(\alpha + [C_d]_{-i})} = \frac{\alpha + [C(d,k')]_{-i}}{\sum_{k=1}^{K}\left(\alpha + [C(d,k)]_{-i}\right)}. \tag{5.13}$$

This equation quantifies how the updated document count influences the probability of assigning topic k' to w_i.

Topic–Word Part: Next, consider the topic–word counts. For topic k' and for the vocabulary index v' corresponding to w_i, we have

$$\frac{B(\beta + C_{k=k'})}{B(\beta + [C_{k=k'}]_{-i})} = \frac{\beta + [C(k', v')]_{-i}}{\sum_{v=1}^{V} \Big(\beta + [C(k', v)]_{-i}\Big)}. \tag{5.14}$$

This expression shows how the removal of w_i modifies the topic–word probability associated with topic k'.

Final Gibbs Update: Combining the document component from Equation (5.13) and the topic–word component from Equation (5.14) into Equation (5.8), we obtain the full Gibbs sampling update:

$$P(z_i = k' \mid Z_{-i}, W) \propto \left[\frac{\alpha + [C(d, k')]_{-i}}{\sum_{k=1}^{K} \Big(\alpha + [C(d, k)]_{-i}\Big)}\right] \times \left[\frac{\beta + [C(k', v')]_{-i}}{\sum_{v=1}^{V} \Big(\beta + [C(k', v)]_{-i}\Big)}\right]. \tag{5.15}$$

Equation (5.15) provides the conditional probability that the ith word (at position j in document d) is assigned to topic k', taking into account the updated counts from the rest of the corpus. The first term (in square brackets) adjusts for the document-specific topic allocation, while the second term adjusts for the topic-specific word distribution. Together, these factors ensure that the Gibbs sampler correctly approximates the posterior distribution over topic assignments.

5.1.4 Gibbs Sampling Algorithm

The overall Gibbs sampling procedure for LDA is summarized in Algorithm 1.

5.2 Alternative Approximate Inference Techniques for LDA

5.2.1 Variational Inference (VI)

Variational inference offers a deterministic alternative to MCMC by casting posterior estimation as an optimization problem. The idea, often called *mean-field variational Bayes*, is to posit a family of simpler distributions $q(\theta, Z)$ with free parameters, and then find the member of this family that

Algorithm 1 Gibbs Sampling for Latent Dirichlet Allocation (LDA)

Input: Corpus of D documents, where document d contains N_d words. Vocabulary $\mathbf{V}$ and number of topics K. Hyperparameters α (for document–topic distributions) and β. Total number of iterations N_{iter}.

Initialize:

$d = 1$ **to** D $n = 1$ **to** N_d Randomly assign topic $z_{d,n} \in \{1, 2, \ldots, K\}$. Update counts:

$$C(d, z_{d,n}) \leftarrow C(d, z_{d,n}) + 1, \quad C(z_{d,n}, w_{d,n}) \leftarrow C(z_{d,n}, w_{d,n}) + 1.$$

$t = 1$ **to** N_{iter} $d = 1$ **to** D $n = 1$ **to** N_d Let $w = w_{d,n}$ and current topic $z = z_{d,n}$. **Remove** the current assignment:

$$C(d, z) \leftarrow C(d, z) - 1, \quad C(z, w) \leftarrow C(z, w) - 1.$$

For each topic $k = 1, \ldots, K$, compute:

$$p_k \propto \frac{\alpha + C(d, k)_{-i}}{\sum_{k'=1}^{K} \Big(\alpha + C(d, k')_{-i}\Big)} \times \frac{\beta + C(k, w)_{-i}}{\sum_{v \in \mathbf{V}} \Big(\beta + C(k, v)_{-i}\Big)},$$

where the subscript $-i$ indicates that the counts exclude the current word $w_{d,n}$. Sample a new topic $z_{d,n}$ from the categorical distribution defined by $\{p_1, \ldots, p_K\}$. **Update** counts with the new assignment:

$$C(d, z_{d,n}) \leftarrow C(d, z_{d,n}) + 1, \quad C(z_{d,n}, w) \leftarrow C(z_{d,n}, w) + 1.$$

Output: Final topic assignments $Z = \{z_{d,n}\}$. Estimate

$$\theta_d \sim \text{Dir}(\alpha + C_d) \quad \text{and} \quad \phi_k \sim \text{Dir}(\beta + C_k),$$

where $C_d = \{C(d, 1), \ldots, C(d, K)\}$ and $C_k = \{C(k, v) : v \in \mathbf{V}\}$.

is closest to the true posterior $p(\theta, Z \mid W)$, typically measured by the Kullback–Leibler divergence (Jordan et al., 1999). In the original LDA paper, Blei et al. (2003) introduced a mean-field variational algorithm for LDA. This method uses an approximating distribution that factorizes over documents and words, governed by variational parameters γ (for each document's topic proportions θ) and ϕ (for each word's topic assignment probabilities). The inference proceeds by iteratively updating these parameters: each update re-estimates the probabilities for the topic assignment for each word in a document given the current estimate of θ, and then updates γ based on the new counts. These steps are repeated until convergence, and γ ultimately yields an approximate posterior Dirichlet for θ_d.

Variational methods are typically much faster than MCMC for large datasets. Each iteration of VI often involves computing expectations or counts rather than drawing thousands of samples, and it usually converges in fewer iterations (since it is doing coordinate ascent on an objective called the Evidence Lower Bound, ELBO) (Bishop, 2006). Additionally, VI is deterministic (given a fixed initialization), which makes it easier to reproduce results and debug. The trade-off is that variational inference is an approximation that does not guarantee asymptotic exactness—it often underestimates uncertainty by fitting a simpler distribution that may not capture the full posterior correlations. In LDA, mean-field VI tends to be slightly less accurate than Gibbs sampling in terms of held-out likelihood or perplexity (Griffiths & Steyvers, 2004; Neal, 1993), especially if the variational family is too restrictive or if it converges to a local optimum of the ELBO. Nonetheless, in situations where speed and scalability are crucial—such as analyzing millions of documents—VI provides a practical solution when MCMC would be infeasible.

5.2.2 Online and Streaming Inference

Standard variational Bayes is a batch algorithm, meaning it uses the entire corpus in each iteration. This becomes problematic when the dataset is extremely large or when data arrive continuously. *Online variational inference* addresses this by updating the model incrementally using mini-batches of documents. Hoffman et al. (2010) pioneered this approach for LDA, introducing an online variational Bayes algorithm that leverages stochastic optimization. The algorithm processes a subset of documents at a time, computes noisy estimates of the variational updates, and then refines the global

parameters (e.g., the topic–word distributions) using a learning rate. Over many mini-batch updates, the model converges to a similar solution as batch VI would, but with dramatically lower memory usage and computational cost per iteration. The main advantage of online inference is its scalability to massive or streaming datasets; it can handle orders of magnitude more data than batch methods by updating the model continuously, rather than requiring multiple passes over a static corpus (Hoffman et al., 2010).

5.2.3 Comparisons and Practical Considerations

Each class of inference method offers a different balance of computational efficiency, accuracy, and scalability. MCMC methods, as described in Neal (1993) and Robert and Casella (2004), tend to yield the most accurate posterior approximations and provide a rich characterization of uncertainty, but are computationally intensive and challenging to scale. In contrast, variational inference sacrifices some accuracy for speed, providing fast, deterministic estimates that work well on large datasets, although it may underestimate uncertainty. Online variational inference further improves scalability, making it feasible to train LDA on web-scale data, at the cost of introducing stochastic noise and requiring careful tuning of additional hyperparameters (Hoffman et al., 2010). In practical applications, the choice of inference method depends on the problem setting. For small corpora or when high accuracy is paramount, MCMC methods may be preferable despite their computational burden. For larger corpora or time-sensitive applications, variational Bayes is attractive for its efficiency, with online variants being the method of choice for streaming or very large datasets.

Chapter 6

From Traditional Topic Models to Graph-Based Latent Structure Models

6.1 Motivation

As we have explored in previous sections, traditional topic modeling techniques—such as Latent Semantic Indexing (LSI), probabilistic Latent Semantic Indexing (pLSI), and Latent Dirichlet Allocation (LDA)—have played a central role in uncovering latent semantic structures in text corpora. However, each approach exhibits noteworthy limitations:

- **LSI:** While LSI leverages Singular Value Decomposition to extract dominant patterns in a document–term matrix (Deerwester et al., 1990), it lacks a probabilistic interpretation and does not provide a principled mechanism to assess uncertainty or model the generative process.
- **pLSI:** Probabilistic extensions such as pLSI introduce a generative model for document–word associations (Hofmann, 1999). Nevertheless, pLSI is prone to overfitting, and its number of parameters grows linearly with the number of documents, making it less scalable to large datasets.
- **LDA:** LDA addresses some drawbacks of pLSI by incorporating Dirichlet priors, which lead to more robust parameter estimation and improved generalization (Blei et al., 2003). Yet, LDA requires the user to

fix the number of topics in advance and can be overly rigid due to the Dirichlet prior, potentially limiting its flexibility in capturing the full complexity of textual data.

These limitations motivate the search for new approaches that more naturally capture latent structures—without sacrificing interpretability or scalability. An alternative perspective is offered by *graph-based modeling*. Graphs are ubiquitous in representing relational data, and the token–document relation can be elegantly modeled as a bipartite network. In such a representation:

1. **Representation:** The document–word count matrix is interpreted as a weighted bipartite graph, where one set of nodes corresponds to documents and the other to words. An edge's weight reflects the frequency of a word's occurrence in a document.

2. **Latent Structure:** Community detection in graphs, which seeks to uncover densely connected groups of nodes, is conceptually analogous to discovering latent topics in text. Here, communities (or blocks) in the graph indicate groups of nodes (documents or words) that share similar properties.

This graph-based view naturally leads to probabilistic generative models for networks. A particularly promising candidate is the *Stochastic Block Model* (SBM) and its hierarchical extensions. Not only do SBMs provide a flexible framework for modeling community structure, but they also enable automatic inference of the number of communities and hierarchical relationships among them. Inspired by the insightful work of Gerlach et al. (2018), who demonstrated an equivalence between pLSI and SBM formulations, the second part of this thesis explores how ideas from network clustering can bridge the gap between traditional topic models and community detection algorithms.

In summary, by reinterpreting the document–token relationship as a graph and applying probabilistic generative models such as the SBM and hierarchical SBM (hSBM), we aim to overcome the limitations of LSI, pLSI, and LDA. This approach not only leverages powerful techniques from network science but also provides a more flexible and nuanced model of latent structure in textual data.

6.2 An Introduction of the Stochastic Block Model

Graphs (or networks) are natural representations for systems of interacting entities. A graph is defined as

$$G = (V, E),$$

where V is the set of nodes (or vertices) and E is the set of edges connecting pairs of nodes. One common and powerful way to represent a graph is via its *adjacency matrix* $A \in \{0,1\}^{n\times n}$, where

$$A_{ij} = \begin{cases} 1, & \text{if there is an edge between nodes } i \text{ and } j\,, \\ 0, & \text{otherwise}\,. \end{cases}$$

For undirected graphs, A is symmetric (i.e., $A_{ij} = A_{ji}$), and typically, $A_{ii} = 0$ to exclude self-loops. This matrix representation enables one to leverage linear algebra and probability theory to study network properties. In cases where multiple edges exist between two nodes, the entry corresponding to that pair can be interpreted as an integer representing the number of edges.

A fundamental problem in network science is *community detection* —identifying groups of nodes that are more densely connected internally than with the remainder of the network. When nodes are rearranged according to community memberships, the adjacency matrix tends to exhibit a *block structure* with dense diagonal blocks (corresponding to within-community connections) and sparser off-diagonal blocks (corresponding to between-community connections) (Fortunato, 2010; M. E. J. Newman, 2006; M. E. J. Newman & Girvan, 2004).

6.2.1 The Stochastic Block Model (SBM)

The Stochastic Block Model provides a principled, probabilistic framework for community detection. Introduced by Holland Holland et al. (1983) and later formalized by Nowicki and Snijders (Nowicki & Snijders, 2001), the SBM assumes that the set of nodes is partitioned into K communities. Let $g_i \in \{1, 2, \ldots, K\}$ denote the community assignment of node i. Under the SBM, edges are generated independently according to

$$P(A_{ij} = 1 \mid g_i = r,\, g_j = s) = \omega_{rs}\,, \tag{6.1}$$

where $\Omega = [\omega_{rs}]_{1 \le r,s \le K}$ is a $K \times K$ matrix of connection probabilities. For undirected graphs, $\omega_{rs} = \omega_{sr}$, and typically, ω_{rr} (the within-community connection probability) is higher than ω_{rs} for $r \neq s$; this structure captures the intuition that nodes within the same community are more likely to be connected.

An alternative formulation for weighted or count data on edges is to assume that

$$A_{ij} \sim \text{Poisson}(\lambda_{g_i,g_j}),$$

where λ_{g_i,g_j} is the expected number (or weight) of edges between nodes in communities g_i and g_j. This formulation reduces to the Bernoulli model in the limit of sparse graphs.

A significant advantage of the SBM is that it is a *generative model*. Given parameters Ω and the community assignments $\{g_i\}$, one can generate synthetic networks that exhibit realistic community structure. Moreover, by fitting an SBM to an observed adjacency matrix A, one can infer the latent community structure, either by maximum-likelihood estimation or by adopting a Bayesian approach (e.g., via Gibbs sampling as in Nowicki and Snijders (2001)).

Recent extensions of the basic SBM include *mixed-membership* models—where each node can belong to multiple communities—and *hierarchical* SBMs, which allow for multiscale or nested community structure. Notably, Peixoto (2014) developed a hierarchical SBM that automatically determines the number of communities and their nested organization based on the data. In this thesis, the SBM and its extensions provide a complementary perspective to topic models like LDA, especially when documents and words are represented as nodes in a bipartite network.

6.3 Example: Bipartite Representation of Documents and Words

Consider a simple corpus with three documents, d_1, d_2, and d_3, and a vocabulary of four words, w_1, w_2, w_3, and w_4. The document–word count matrix

$X \in \mathbb{R}^{4\times 3}$ is given by

$$X = \begin{pmatrix} 2 & 0 & 1 \\ 0 & 3 & 1 \\ 1 & 0 & 2 \\ 0 & 1 & 0 \end{pmatrix}.$$

Here, each entry X_{ij} represents the number of times word w_i appears in document d_j; in fact, this is equivalent to the term-document matrix common in latent semantic indexing (LSI).

This matrix can be naturally interpreted as a *bipartite graph* $G = (V_W \cup V_D, E)$, where:

- $V_W = \{w_1, w_2, w_3, w_4\}$ is the set of word nodes.
- $V_D = \{d_1, d_2, d_3\}$ is the set of document nodes.
- An edge $e = (w_i, d_j) \in E$ exists if $X_{ij} > 0$, with the edge weight equal to X_{ij}.

Figure 6.1 illustrates this bipartite graph representation. Such bipartite formulations are well established in the literature; for example, Barber (2007) discusses modularity in bipartite networks, and M. Newman (2010) provides a comprehensive review of network representations and community detection.

This bipartite graph provides a natural framework for linking topic models and network models. When documents and words are represented as nodes, the document–word count matrix X can be interpreted as the weighted adjacency matrix of a bipartite network.

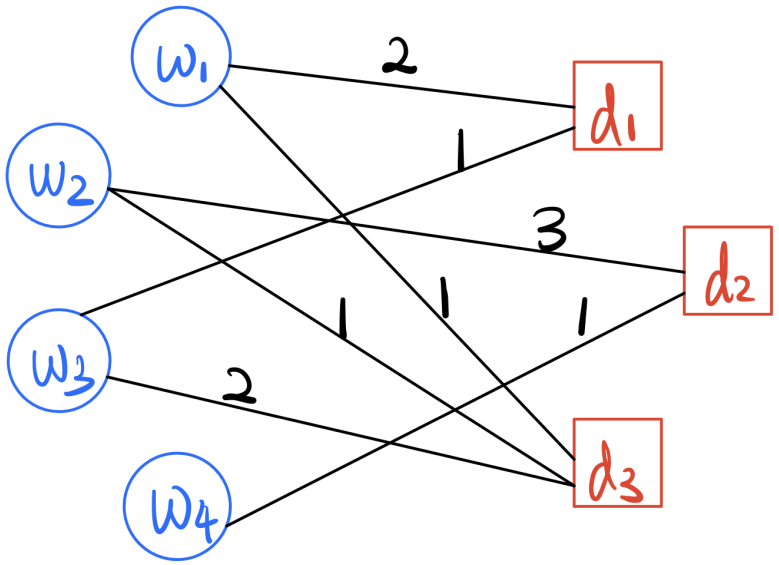

Figure 6.1: A toy example of a bipartite graph representation of documents and words. The left set of nodes represents words and the right set represents documents. An edge from word w_i to document d_j is drawn if the count $X_{ij} > 0$, with the edge weight equal to the count of words in the document.

Chapter 7

Relating SBM and pLSI

In this chapter, we establish a formal equivalence between probabilistic latent semantic indexing (pLSI) and a Poisson-based stochastic block model (SBM) on a bipartite document–word network. While inspired by Gerlach et al. (2018), our derivation is self-contained and proceeds in three stages. First, we recast pLSI's multinomial bag-of-words likelihood as independent Poisson counts, exposing clean factorization (Section 7.1). Next, we introduce document- and word-specific rate parameters and reparameterize the topic–word interactions to reveal a symmetric community-membership interpretation for both node types (Section 7.2). Finally, we show that under this mapping, the Poisson rates of pLSI coincide exactly with the SBM's edge-weight parameters, yielding identical joint likelihoods for the observed counts (Section 7.3). Figure 7.1 illustrates the core idea: each document–word occurrence becomes a weighted edge whose rate is determined by shared topic (community) affiliations.

7.1 From Bag-of-Words to a Poisson Likelihood in pLSI

The pLSI model introduced in Chapter 2 represents each document $d \in \{1, \dots, D\}$ as a mixture over K latent topics. Under the classical *bag-of-words* assumption, each token in document d is generated by first sampling a topic

$$k \sim \text{Multinomial}\big(\theta_d\big),$$

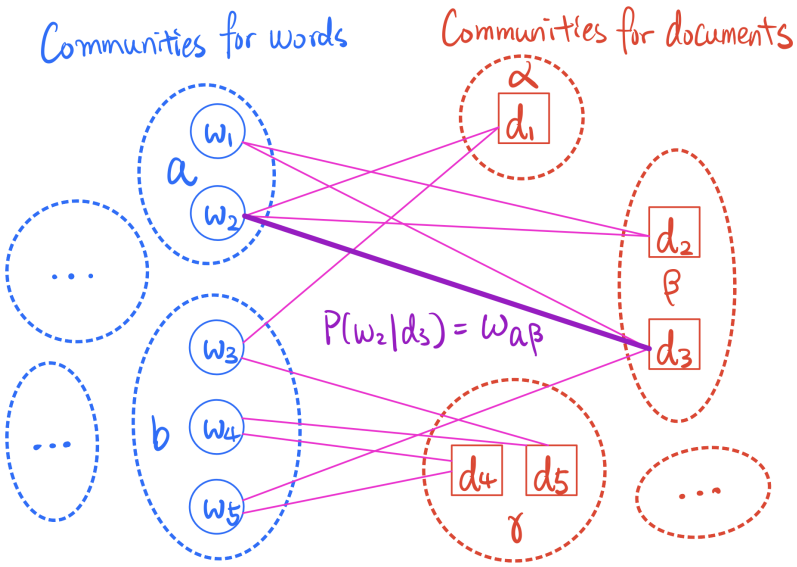

Figure 7.1: Bipartite document–word graph with topic-based communities.

and then sampling a word

$$w \;\sim\; \text{Multinomial}\big(\phi_k\big).$$

Equivalently, the marginal probability of observing word w in document d is

$$P(w \mid d) \;=\; \sum_{k=1}^{K} \theta_{d,k}\, \phi_{k,w},$$

where $\theta_{d,k} = P(k \mid d)$ and $\phi_{k,w} = P(w \mid k)$.

Let $n(d, w)$ denote the observed count of word $w \in \{1, \dots, V\}$ in document d, and write

$$N_d \;=\; \sum_{w=1}^{V} n(d, w)$$

for the total number of tokens in d. Under the bag-of-words multinomial model, the likelihood of the entire corpus is

$$P\big(\{n(d,w)\} \mid \{\theta_d\}, \{\phi_k\}\big) = \prod_{d=1}^{D} \left(\frac{N_d!}{\prod_{w=1}^{V} n(d,w)!} \right) \prod_{w=1}^{V} \big[P(w \mid d)\big]^{n(d,w)}.$$

To build a bridge to stochastic block models, we now recast this multinomial formulation into an equivalent *Poisson* representation. First, treat the document length N_d as random:

$$N_d \;\sim\; (h_d),$$

so that

$$P(N_d = n) = \frac{h_d^n \, e^{-h_d}}{n!},$$

with $h_d > 0$ the expected length of document d. Conditional on N_d, we then generate each of the N_d tokens independently according to the same topic–word mixture.

A well-known result shows that marginalizing out the multinomial token assignments yields an equivalent factorization in terms of independent Poisson counts:

$$n(d,w) \;\sim\; \big(\ell_{d,w}\big), \quad \ell_{d,w} = h_d \sum_{k=1}^{K} \theta_{d,k}\, \phi_{k,w}.$$

Hence the joint distribution over $\{n(d,w)\}_{w=1}^{V}$ is

$$P\big(n(d,1), \ldots, n(d,V)\big) = \prod_{w=1}^{V} \frac{\ell_{d,w}^{\,n(d,w)}}{n(d,w)!}\, e^{-\ell_{d,w}},$$

which—upon summing over all documents—recovers exactly the same likelihood as the multinomial bag-of-words model (up to the multinomial normalization constants).

Why the Poisson viewpoint matters.

- The Poisson form dispenses with combinatorial coefficients and yields independent factors $n(d,w) \sim (\ell_{d,w})$.
- The rate parameter $\ell_{d,w} = h_d \sum_k \theta_{d,k}\phi_{k,w}$ naturally interprets as the *expected edge weight* between document node d and word node w in a bipartite network.
- This representation paves the way to reinterpret pLSI as a Poisson-based stochastic block model in Section 7.2.

7.2 Transition from pLSI to a Stochastic Block Model Perspective

Armed with the Poisson representation of pLSI (Section 7.1), we now reinterpret document–word counts as edge-weights in a bipartite network and show how pLSI coincides with a Poisson-based stochastic block model (SBM).

Expected Poisson Rates as Bipartite Edge Weights

Under the Poisson formulation, each count

$$n(d,w) \;\sim\; \big(\ell_{d,w}\big), \quad \ell_{d,w} = h_d \sum_{k=1}^{K} \theta_{d,k}\, \phi_{k,w},$$

captures the expected number of times word w appears in document d. We interpret each document d and word w as nodes in a bipartite graph, and $n(d,w)$ as the (integer) weight of the edge (d,w). The Poisson rate $\ell_{d,w}$ thus plays exactly the role of the SBM's expected edge weight between these two node sets.

Symmetry via Topic Reparameterization

To make the parallel with SBM membership more transparent, introduce a word-specific scale $h_w > 0$ and normalized topic-word probabilities

$$\phi'_{k,w} \;=\; \frac{\phi_{k,w}}{h_w}, \qquad h_w \;=\; \sum_{k=1}^{K} \phi_{k,w}.$$

Then

$$\phi_{k,w}\, \theta_{d,k} = h_w \,\big(\theta_{d,k}\, \phi'_{k,w}\big),$$

and

$$\ell_{d,w} = h_d\, h_w \sum_{k=1}^{K} \theta_{d,k}\, \phi'_{k,w}.$$

This factorization exhibits a perfect symmetry: each node d has an intrinsic activity h_d, each node w has activity h_w, and their interaction is mediated by the inner product of membership vectors $\theta_d = (\theta_{d,1}, \ldots, \theta_{d,K})$ and $\theta_w = (\phi'_{1,w}, \ldots, \phi'_{K,w})$.

Mapping to the Poisson SBM

A Poisson SBM on a bipartite graph with document nodes $\{1, \dots, D\}$ and word nodes $\{1, \dots, V\}$ generates each edge-weight $A_{d,w}$ independently as

$$A_{d,w} \;\sim\; \big(\lambda_{d,w}\big), \quad \lambda_{d,w} = h_d\, h_w \sum_{k=1}^{K} \theta_{d,k}\, \theta_{w,k},$$

where $\theta_{d,k}$ and $\theta_{w,k}$ are the (nonnegative) membership weights of document d and word w in community (topic) k, respectively, satisfying $\sum_k \theta_{d,k} = 1$ and $\sum_k \theta_{w,k} = 1$.

By identifying

$$\theta_{d,k} \;\equiv\; \theta_{d,k}, \quad \theta_{w,k} \;\equiv\; \phi'_{k,w}, \quad \lambda_{d,w} \;\equiv\; \ell_{d,w},$$

we see that the Poisson-SBM likelihood exactly matches the pLSI Poisson likelihood:

$$P\big(A_{d,w} = n(d,w)\big) = \frac{\lambda_{d,w}^{n(d,w)}}{n(d,w)!} e^{-\lambda_{d,w}} \;=\; \frac{\ell_{d,w}^{n(d,w)}}{n(d,w)!} e^{-\ell_{d,w}}.$$

Key insight. *Under a Poisson generative view, pLSI is equivalent to a Poisson-SBM on a bipartite document–word network, with topic assignments playing the role of community memberships.* This equivalence opens the door to applying community-detection tools (e.g., hierarchical SBMs, likelihood-based inference) to topic-modeling problems, and conversely to enriching SBM methodology with ideas from latent-variable language models.

7.3 Proof of Equivalence between pLSI and the Poisson SBM

In order to demonstrate that the Poissonized pLSI model introduced in Section 7.1 and a Poisson stochastic block model (SBM) on the same bipartite document–word graph produce identical joint likelihoods for the observed counts $n(d, w)$, we begin by recalling the Poisson form of pLSI. From Section 7.1, each count is modeled as

$$n(d,w) \;\sim\; \big(\ell_{d,w}\big), \qquad \ell_{d,w} = h_d \sum_{k=1}^{K} \theta_{d,k}\, \phi_{k,w},$$

where h_d is the expected length of document d, $\theta_{d,k}$ its mixture weight on topic k, and $\phi_{k,w}$ the probability of word w under topic k. Because Poisson variables factorize independently, the joint likelihood under pLSI can be written as

$$L_{\text{pLSI}} = \prod_{d=1}^{D} \prod_{w=1}^{V} \frac{\ell_{d,w}^{n(d,w)}}{n(d,w)!}\, e^{-\ell_{d,w}}.$$

To reveal a symmetric "community-membership" interpretation that treats documents and words on equal footing, we reparameterize the word-side probabilities. Define the word activity

$$h_w = \sum_{k=1}^{K} \phi_{k,w},$$

which captures the overall prevalence of word w across topics, and introduce normalized topic–word memberships

$$\phi'_{k,w} = \frac{\phi_{k,w}}{h_w}, \quad \sum_{k=1}^{K} \phi'_{k,w} = 1.$$

Substituting these into the rate gives

$$\ell_{d,w} = h_d\, h_w \sum_{k=1}^{K} \theta_{d,k}\, \phi'_{k,w}.$$

We may then view $\theta_d = (\theta_{d,1}, \ldots, \theta_{d,K})$ and $\theta_w = (\phi'_{1,w}, \ldots, \phi'_{K,w})$ as membership vectors in the same K-dimensional simplex, one for each document and one for each word. In this form, $\ell_{d,w}$ appears exactly as the inner product $\theta_d^\top \theta_w$ scaled by the product of node activities $h_d h_w$.

We now consider a Poisson SBM on the bipartite graph of documents and words. In this model, each edge weight $A_{d,w}$ is generated independently as

$$A_{d,w} \sim \big(\lambda_{d,w}\big), \qquad \lambda_{d,w} = h_d\, h_w \sum_{k=1}^{K} \theta_{d,k}\, \theta_{w,k},$$

where θ_d and θ_w are membership vectors in K communities and $h_d h_w$ governs overall activity. The joint likelihood under the SBM factorizes as

$$L_{\text{SBM}} = \prod_{d=1}^{D} \prod_{w=1}^{V} \frac{\lambda_{d,w}^{A_{d,w}}}{A_{d,w}!}\, e^{-\lambda_{d,w}}.$$

Finally, the equivalence follows immediately by identifying the two sets of parameters. By setting

$$\lambda_{d,w} = \ell_{d,w}, \qquad \theta_{w,k} = \phi'_{k,w},$$

we obtain

$$\lambda_{d,w} = h_d\, h_w \sum_{k=1}^{K} \theta_{d,k}\, \theta_{w,k} = h_d\, h_w \sum_{k=1}^{K} \theta_{d,k}\, \phi'_{k,w} = \ell_{d,w}.$$

Under this mapping, each factor in L_{SBM} coincides exactly with the corresponding factor in L_{pLSI}, and hence

$$L_{\mathrm{SBM}} = L_{\mathrm{pLSI}}.$$

This completes the proof that, under the Poisson generative construction and the above reparameterization, pLSI and the Poisson SBM on a bipartite document–word network assign the same joint probability to every observed count $n(d, w)$, establishing their formal equivalence.

Conclusion. *Under the Poisson generative construction and the above reparameterization, pLSI and the Poisson SBM on a bipartite document–word network are mathematically equivalent: they assign the same joint probability to all observed counts* $\{n(d, w)\}$. This equivalence justifies cross-application of inference algorithms and theoretical results between topic models and community-detection frameworks.

Chapter 8

Metrics for LDA and hSBM: MDL and Coherence

In the previous chapter, we explored how topic modeling and community detection can be unified through a network-based interpretation of probabilistic models. With this foundation in place, we now turn to a practical question: how can we compare the quality of different models?

Evaluating models like Latent Dirichlet Allocation (LDA) and hierarchical Stochastic Block Models (hSBM) requires metrics that capture both statistical performance and human interpretability. Some models may fit the data well but be overly complex; others may offer simpler explanations but fail to capture important patterns. To address this trade-off, we study two types of evaluation criteria:

- **Minimum Description Length (MDL)**: an information-theoretic framework that rewards models which explain the data well while remaining concise (Barron & Cover, 1991; Grünwald, 2007; Rissanen, 1978).
- **Topic Coherence**: a set of interpretability measures that assess how semantically consistent the top words of each topic are.

This chapter begins with the MDL principle, which offers a mathematically grounded way to compare models by counting how many "bits" are needed to describe both the data and the model. We then move on to coherence metrics, which provide more human-centric evaluations based on semantic associations among topic words.

8.1 Minimum Description Length

The Minimum Description Length (MDL) principle provides a powerful approach to model selection grounded in information theory. At its heart lies a simple question: if we wanted to transmit both the data and the model over a digital channel, how many bits would we need?

Rather than focusing solely on how well a model fits the data, MDL balances two competing objectives: (1) explaining the data as accurately as possible, and (2) keeping the model itself as simple as possible. This embodies a formal version of Occam's Razor—favoring simpler models when they are sufficient (Grünwald, 2007; Rissanen, 1978).

8.1.1 The MDL Framework

Let $\mathcal{M}$ represent a model (e.g., "LDA with $K = 10$ topics"), and let $\mathbf{D}$ denote the observed dataset. Then the total description length S is defined as:

$$S(\mathcal{M}; \mathbf{D}) = \underbrace{-\ln P(\mathbf{D} \mid \mathcal{M})}_{\text{data encoding}} + \underbrace{-\ln P(\mathcal{M})}_{\text{model encoding}}. \tag{8.1}$$

Each term has a natural interpretation:

- The first term, $-\ln P(\mathbf{D} \mid \mathcal{M})$, measures how well the model explains the data. This is also known as the *negative log-likelihood*, and it reflects how many bits are needed to encode the data assuming the model is correct.
- The second term, $-\ln P(\mathcal{M})$, reflects the complexity of the model itself—essentially, the cost of describing the model, including its structure and parameters. [1]

In practice, more complex models can reduce the data encoding term by fitting the data more closely. However, this comes at the cost of a longer

[1] For example, consider the model "LDA with $K = 10$ topics." In the MDL framework, the term $-\ln P(\mathcal{M})$ quantifies how many bits are needed to describe the model itself. This includes encoding the number of topics ($K = 10$), the topic–word distributions, the document–topic distributions, and the hyperparameters (e.g., α, β). More complex models—such as those with more topics or greater freedom in parameter values—require more bits to describe, and hence incur a larger model encoding cost.

model description. Simpler models, on the other hand, are cheaper to describe but may leave patterns in the data unexplained. The MDL principle balances these forces by selecting the model that minimizes the total cost S.

MDL also has strong connections to Bayesian statistics. If we interpret $-\ln P(\mathcal{M})$ as the negative log-prior and $-\ln P(\mathbf{D} \mid \mathcal{M})$ as the negative log-likelihood, then minimizing MDL corresponds to finding the maximum a posteriori (MAP) estimate of the model (Barron & Cover, 1991).

8.1.2 A Simple Example: Coin Tosses

To build intuition, consider a toy problem: modeling a sequence of three coin tosses:

$$D = \{H, H, T\}.$$

Suppose we model the probability of heads as p, so the likelihood of observing this sequence is:

$$P(D \mid p) = p^2(1-p).$$

The data encoding cost is the negative log-likelihood:

$$S_{\text{data}} = -\ln P(D \mid p) = -(2\ln p + \ln(1-p)).$$

Now suppose we assume a uniform prior over p, i.e., all values in $[0, 1]$ are equally likely. Then the model encoding cost is:

$$S_{\text{model}} = -\ln P(p) = 0.$$

Thus, the total description length is:

$$S(p) = S_{\text{data}} + S_{\text{model}} = -(2\ln p + \ln(1-p)).$$

Minimizing $S(p)$ yields $p = 2/3$, which is also the maximum likelihood estimate. This example shows how MDL naturally integrates model fit and complexity, even in simple settings.

8.1.3 Applying MDL to Topic Models

We now extend the MDL principle to more complex models used in topic modeling: Latent Dirichlet Allocation (LDA) and hierarchical Stochastic Block Models (hSBM). Both models aim to uncover hidden structure in a document corpus—LDA by discovering latent topics, and hSBM by detecting communities in a document–word network. In both cases, the goal is to find a balance between how well the model explains the data and how costly it is to describe the model itself.

Description Length in LDA

Latent Dirichlet Allocation (Blei et al., 2003) assumes that each document is a mixture of topics and that each topic is a distribution over words. The key components of the model are:

- The document–topic distributions θ_d for each document d,
- The topic–word distributions ϕ_k for each topic k,
- The latent topic assignments $z_{d,n}$ for each word token,
- The Dirichlet hyperparameters α and β.

The total description length of LDA can be written as:

$$S_{\text{LDA}} = -\ln P(\mathbf{w}, \mathbf{z} \mid \theta, \phi) - \ln P(\theta \mid \alpha) - \ln P(\phi \mid \beta) - \ln P(\alpha, \beta),$$

where each term reflects the cost of encoding either the observed data, latent variables, parameters, or hyperparameters. Since exact computation of these terms is intractable, practical implementations use approximations—such as variational inference or Gibbs sampling—to estimate the likelihood and marginalize out latent variables (Wallach et al., 2009).

Description Length in hSBM

The hierarchical Stochastic Block Model (Peixoto, 2013, 2014) takes a different perspective: it treats the corpus as a bipartite graph where documents and words are nodes, and edges represent word frequencies. The goal is to

group these nodes into hierarchical communities that explain the observed connections.

The MDL expression for hSBM is:

$$S_{\text{hSBM}} = -\ln P(A \mid \mathcal{T}, \Lambda) - \ln P(\Lambda \mid \mathcal{T}) - \ln P(\mathcal{T}),$$

where A is the observed document–word adjacency matrix, $\mathcal{T}$ is the hierarchical partition tree, and Λ represents the edge parameters (e.g., Poisson rates). Unlike LDA, the hSBM model admits an exact MDL formulation, which makes it highly suitable for principled model comparison (Peixoto, 2013).

Comparing LDA and hSBM

By computing S_{LDA} and S_{hSBM} on the same corpus, we can make an apples-to-apples comparison: which model provides a more efficient explanation of the data? A smaller description length means the model is both better at capturing structure and simpler to describe. This unified MDL framework offers a principled way to compare topic models and network-based models on equal footing. In summary, the MDL principle helps us look beyond raw likelihood or classification accuracy. It forces us to ask: which model tells the best story with the fewest words?

8.2 Coherence Measures

While the MDL principle helps us compare topic models from a statistical or compression-oriented perspective, it does not always align with human interpretability. In practical applications, we often care not only about how well a model fits the data, but also about how meaningful and coherent its topics appear to human readers.

Topic coherence is a family of evaluation metrics that attempt to quantify this interpretability. These metrics examine how semantically consistent the most probable words in a topic are—under the assumption that coherent topics will consist of words that frequently co-occur or appear in similar contexts. For example, a topic with the words `doctor`, `hospital`, `nurse`, `patient`, and `treatment` is more coherent than one with `doctor`, `planet`, `banana`, `algorithm`, and `happiness`.

Early evaluations of topic quality used *perplexity*, the exponentiated negative predictive log-likelihood on held-out data (Blei et al., 2003). However, studies have shown that perplexity correlates poorly with human judgments of topic interpretability (Chang et al., 2009). As a result, coherence measures have become standard tools for evaluating topics in an interpretable way.

This section presents three main types of coherence measures: (1) direct co-occurrence-based measures, (2) context-vector-based measures, and (3) contextualized measures using large language models.

8.2.1 Direct Coherence Measures

Direct coherence metrics quantify how often pairs of words from a topic occur together in a corpus. These methods typically consider the top N words $W = \{w_1, w_2, \ldots, w_N\}$ for each topic and average some function of their pairwise co-occurrence statistics.

UMass Coherence

Mimno et al. (2011) proposed the UMass coherence metric, which measures conditional log-probabilities based on word-document co-occurrence counts in the *training corpus*. For a topic's top words W, the coherence score is:

$$C_{\text{UMass}}(W) = \frac{2}{N(N-1)} \sum_{i=2}^{N} \sum_{j=1}^{i-1} \log \frac{D(w_i, w_j) + \varepsilon}{D(w_j)}, \tag{8.2}$$

where $D(w_i, w_j)$ is the number of documents containing both words w_i and w_j, $D(w_j)$ is the number of documents containing w_j, and ε is a small smoothing constant (e.g., 10^{-12}) to avoid taking the logarithm of zero.

This metric approximates the conditional probability $P(w_i \mid w_j)$, and rewards word pairs that frequently appear together. Larger (less negative) scores indicate more coherent topics. Because it uses only the training corpus, UMass coherence is computationally efficient but may favor frequent or generic words.

PMI-Based Coherence (UCI Coherence)

An alternative approach is pointwise mutual information (PMI), introduced in this context by D. Newman et al. (2010). Unlike UMass coherence,

PMI-based coherence typically relies on an external reference corpus such as Wikipedia. Co-occurrence counts are collected using a sliding window over the corpus (rather than whole documents), and probabilities are estimated as:

$$P(w_i) \approx \frac{\#\text{ windows containing } w_i}{\#\text{ total windows}},$$
$$P(w_i, w_j) \approx \frac{\#\text{ windows containing both } w_i, w_j}{\#\text{ total windows}}.$$

The PMI for a pair of words is:

$$\text{PMI}(w_i, w_j) = \log \frac{P(w_i, w_j) + \varepsilon}{P(w_i) \cdot P(w_j)}, \tag{8.3}$$

and the coherence score is the average PMI over all word pairs:

$$C_{\text{PMI}}(W) = \frac{2}{N(N-1)} \sum_{i=2}^{N} \sum_{j=1}^{i-1} \text{PMI}(w_i, w_j). \tag{8.4}$$

To improve comparability across topics, one often uses the normalized variant:

$$\text{NPMI}(w_i, w_j) = \frac{\text{PMI}(w_i, w_j)}{-\log(P(w_i, w_j) + \varepsilon)},$$

which ranges between -1 and 1. Higher NPMI values indicate stronger associations. Because these PMI measures use large, diverse corpora, they often align well with human evaluations of topic quality (Lau et al., 2014).

8.2.2 Indirect Coherence Measures

Indirect coherence metrics go beyond raw co-occurrence and instead compare words based on their overall distributional context. These methods build *context vectors* for each word, representing how often it co-occurs with many other words across the corpus. Coherence is then measured by averaging pairwise similarities (often cosine similarity) between these context vectors.

Aletras and Stevenson (2013) and Röder et al. (2015) pioneered this approach. For a topic $W = \{w_1, \ldots, w_N\}$, let $\mathbf{v}_i$ be the context vector for word w_i. Then the context-vector coherence is:

$$C_v(W) = \frac{2}{N(N-1)} \sum_{i=2}^{N} \sum_{j=1}^{i-1} \cos(\mathbf{v}_i, \mathbf{v}_j). \tag{8.5}$$

The cosine similarity captures how similarly two words are used across contexts, even if they do not co-occur directly. This method can identify deeper semantic links and often performs well in human evaluations, especially when trained on large corpora (Röder et al., 2015).

8.2.3 Contextualized Coherence via Language Models

Recent advances in large language models (LLMs) have introduced new coherence metrics that go beyond frequency counts and vector spaces. Instead, these approaches use masked language models (MLMs), such as BERT or LLaMA 2, to measure how well one topic word helps predict another in a realistic sentence context.

Grootendorst (2022) and Touvron et al. (2023) demonstrate that MLM-based coherence metrics often correlate more strongly with human judgments. One such measure is:

$$C_{\text{MLM}}(w_i, w_j) = \frac{1}{2} \left(\log P_{\text{MLM}}(w_i \mid w_j) + \log P_{\text{MLM}}(w_j \mid w_i) \right), \tag{8.6}$$

where the probabilities are obtained by feeding masked sentences into a language model. The final coherence score is the average over all pairs (w_i, w_j) in the topic.

These contextualized approaches open new doors for evaluating coherence in domain-specific settings or languages with limited training data, making them an exciting area of ongoing research.

Implementation Note. Coherence is typically computed on a per-topic basis. To assess the overall quality of a model, one often reports the *mean* and *standard deviation* of per-topic coherence scores across all topics.

Chapter 9

Empirical Evaluation and Results

This chapter presents a comprehensive empirical comparison between Latent Dirichlet Allocation (LDA) and the hierarchical Stochastic Block Model (hSBM) for unsupervised latent structure discovery. Building on the theoretical foundations and algorithmic implementations discussed in previous chapters, we evaluate both models on a benchmark corpus using key metrics that reflect predictive accuracy, semantic interpretability, and model parsimony. These metrics include *topic coherence* (C_v), *log perplexity*, *minimum description length* (MDL), and *training time*. There will be more details regarding these specific metrics in 9.1.

To ensure fairness and reproducibility, both models are trained on identically preprocessed versions of the 20 Newsgroups dataset, and their outputs are evaluated using shared vocabularies and consistent coherence estimation procedures. We analyze performance across two regimes: a fine-grained experimental sweep with smaller training sizes and more data points, and a coarse-grained benchmark with larger training sizes but fewer checkpoints. These dual perspectives offer insight into the scalability, robustness, and effectiveness of each model under varying data conditions.

The following sections present our evaluation methodology, describe each metric in detail, and analyze trends through a series of comparative plots. Our findings demonstrate that while LDA offers superior perplexity and faster training time, hSBM consistently excels in coherence and compression, highlighting a trade-off between generative likelihood and structural parsimony in unsupervised topic modeling.

9.1 Experimental Setup and Metric Definitions

9.1.1 Dataset and Preprocessing

We evaluate both models on the widely-used `20 Newsgroups` dataset, which contains approximately 20,000 documents partitioned across 20 categories. For our purposes, we focus on the training subset and remove headers, footers, and quoted replies to reduce noise. All documents are lowercased, tokenized using `gensim`'s `simple_preprocess` utility, and filtered to exclude stopwords and low-frequency terms. This ensures a clean and consistent vocabulary for both models.

To investigate scalability and sensitivity to corpus size, we conduct two types of experimental sweeps:

- **Small-scale benchmark:** sample sizes range from 500 to 5000 documents in increments of 500, yielding fine-grained performance curves.
- **Large-scale benchmark:** sample sizes range from 1000 to 10000 documents in increments of 1000, offering coarse insights into asymptotic trends.

9.1.2 Evaluation Metrics

We compare LDA and hSBM across four core metrics, each reflecting a distinct modeling priority:

Topic Coherence (C_v): Topic coherence measures the degree of semantic similarity among high-probability words within each topic. We use the C_v metric from `gensim`'s `CoherenceModel`, which combines a sliding window, normalized pointwise mutual information (NPMI), and cosine similarity. For fairness, we evaluate coherence using the same preprocessed documents and dictionary for both LDA and hSBM. Coherence results are plotted in Figures 9.5 and 9.6.

Log Perplexity: Perplexity is a standard measure of language model performance, reflecting how well a model predicts unseen data. For LDA, perplexity is calculated using `gensim`'s built-in approximation. For hSBM, we

implement a custom approximation based on inferred document-topic and topic-word distributions:

$$\log p(w|d) \approx \log \left(\sum_{t=1}^{B} P(t|d) \cdot P(w|t) \right),$$

where B is the number of inferred topics. Both models' log perplexities are shown in Figures 9.3 and 9.4.

Minimum Description Length (MDL): MDL serves as a model selection criterion grounded in information theory. For LDA, we compute:

$$\text{MDL}_{\text{LDA}} = -\text{log-likelihood} + \frac{1}{2}(K(V-1) + D(K-1)) \cdot \log N,$$

where K is the number of topics, V is the vocabulary size, D is the number of documents, and N is the total number of tokens. For hSBM, the MDL is directly obtained from the entropy of the inferred block state in `graph-tool`. MDL results are visualized in Figures 9.1 and 9.2.

Training Time: We record wall-clock training time for both models using Python's `time.time()` function. LDA is trained using 20 full passes over the corpus, and hSBM is initialized once per setting. Timing comparisons appear in Figures 9.7 and 9.8.

9.2 Results and Visual Analysis

9.2.1 Minimum Description Length (MDL)

Figures 9.1 and 9.2 show that hSBM consistently achieves a lower MDL than LDA across all dataset sizes. In the fine-grained regime, LDA's MDL increases steeply and almost linearly with the number of documents. In contrast, hSBM exhibits a more gradual, sublinear increase in description length.

This outcome aligns with theoretical expectations. The MDL of hSBM directly reflects the model's ability to compress network structure via hierarchical partitioning. Since hSBM infers both the number of groups and their hierarchical organization, it avoids overparameterization. LDA, in contrast,

requires a fixed number of topics and does not adapt its complexity to the data. As a result, its model complexity grows rapidly with document count, leading to higher encoding cost.

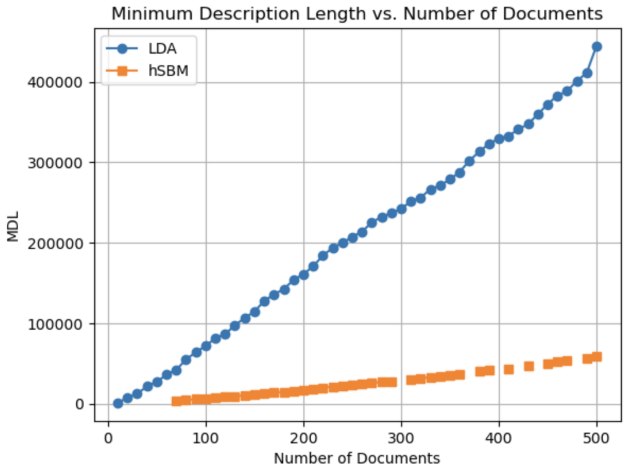

Figure 9.1: Minimum Description Length vs. Number of Documents (Small-scale)

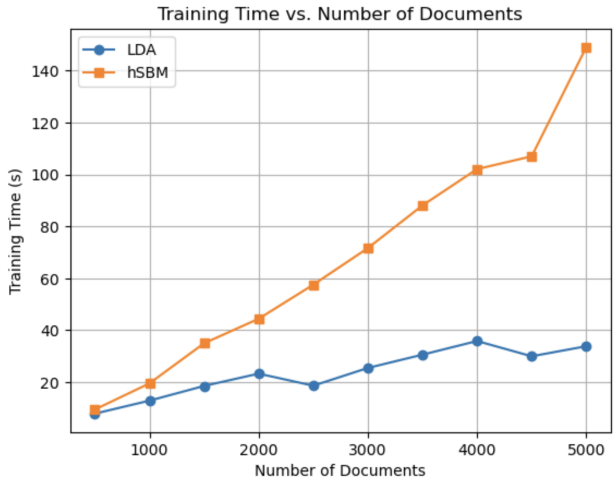

Figure 9.2: Minimum Description Length vs. Number of Documents (Large-scale)

9.2.2 Log Perplexity

Figures 9.3 and 9.4 show that LDA achieves significantly lower log perplexity compared to hSBM. Perplexity values for LDA remain stable and low, suggesting its predictive distributions fit the observed data well. hSBM, on the other hand, maintains higher log perplexity, though the curve is relatively flat.

This discrepancy is expected given that LDA is optimized explicitly for log-likelihood over held-out tokens. Its training objective directly minimizes perplexity. Conversely, hSBM prioritizes structural compression over token-level prediction. The higher perplexity thus reflects a trade-off: hSBM sacrifices some predictive accuracy to gain more parsimonious representations.

Despite this, the stability of hSBM's perplexity curve—despite increasing corpus size—suggests strong generalization and robustness. It captures core semantic groupings without overfitting to token-level co-occurrences.

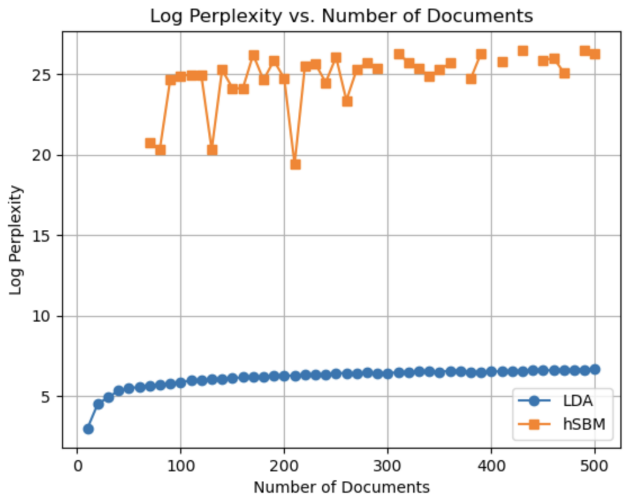

Figure 9.3: Log Perplexity vs. Number of Documents (Small-scale)

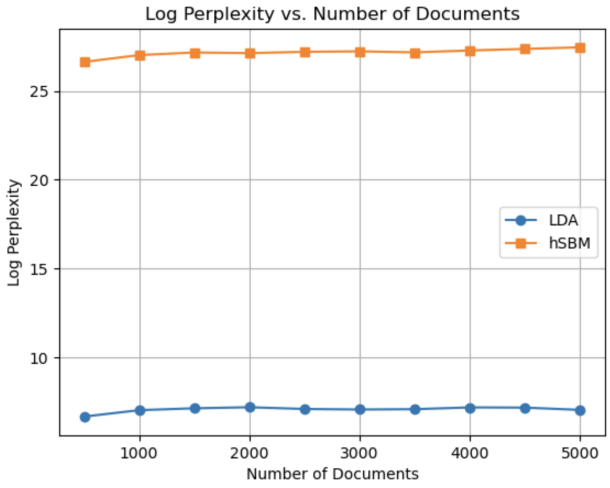

Figure 9.4: Log Perplexity vs. Number of Documents (Large-scale)

9.2.3 Topic Coherence (C_v)

Topic coherence results, shown in Figures 9.5 and 9.6, highlight hSBM's advantage in interpretability. For most of the training sizes, hSBM achieves higher or comparable coherence than LDA. Particularly in the range of 1500 to 4000 documents, hSBM's coherence peaks and outperforms LDA by a notable margin.

Coherence measures how top topic words co-occur within documents. The stronger performance of hSBM suggests that its inferred word groups are more semantically consistent. This stems from its treatment of words and documents as nodes in a bipartite graph, where communities emerge from mutual interactions, not just co-occurrence counts.

Interestingly, hSBM's coherence slightly declines at the largest dataset sizes. This may be due to vocabulary dilution or the model's increasing reliance on structural regularities over local semantic clusters. Meanwhile, LDA's coherence improves slowly with more data, narrowing the gap in the large-scale benchmark.

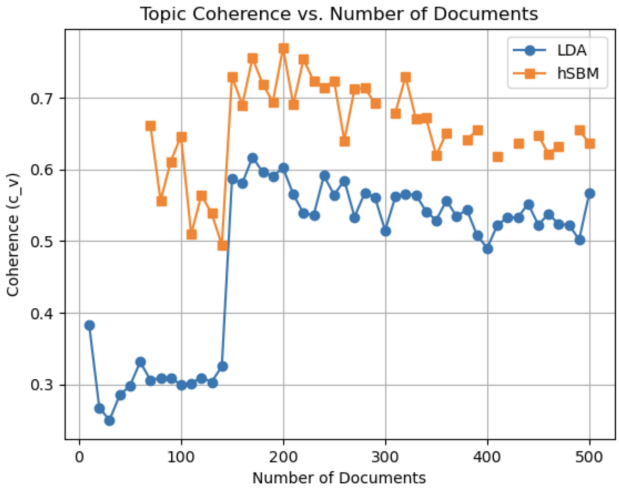

Figure 9.5: Topic Coherence vs. Number of Documents (Small-scale)

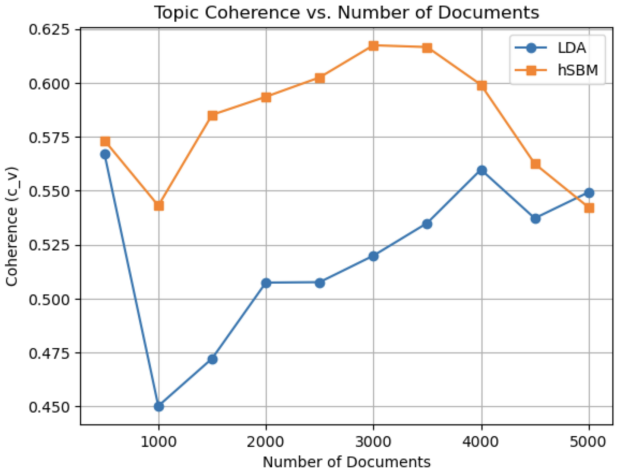

Figure 9.6: Topic Coherence vs. Number of Documents (Large-scale)

9.2.4 Training Time

As expected, Figures 9.7 and 9.8 show that LDA trains significantly faster than hSBM. Even with 5000 documents, LDA completes in under 40 seconds, while hSBM exceeds 140 seconds on the same data.

This difference arises from model complexity. LDA's variational updates and sparse Dirichlet priors enable efficient optimization. hSBM, by contrast, performs nested MCMC sampling over potential partitions in a bipartite graph—an inherently costlier process. The nearly linear increase in hSBM's training time suggests that its scalability could be improved with parallelization or GPU acceleration.

Nevertheless, hSBM remains tractable for mid-sized datasets and delivers strong performance on interpretability and compression.

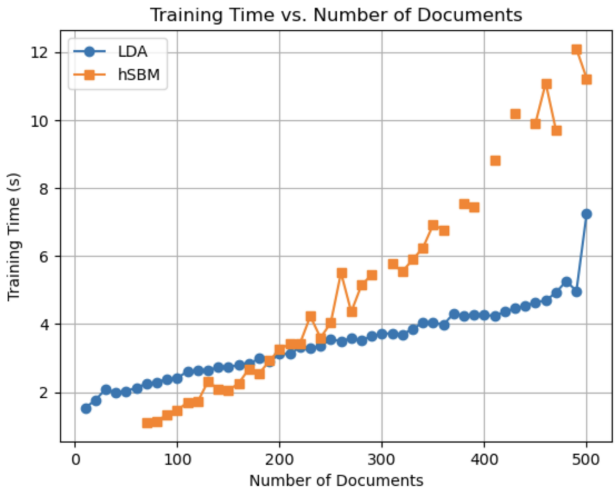

Figure 9.7: Training Time vs. Number of Documents (Small-scale)

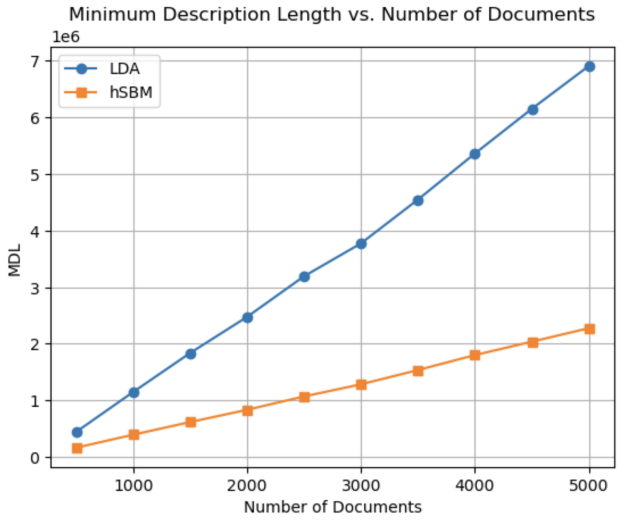

Figure 9.8: Training Time vs. Number of Documents (Large-scale)

9.3 Discussion and Summary

The experimental results demonstrate clear trade-offs between LDA and hSBM in the context of unsupervised topic modeling. LDA excels in predictive performance and computational efficiency, achieving lower log perplexity and faster runtimes across all dataset sizes. However, this comes at the cost of model complexity and less coherent topics at small and medium scales.

hSBM, on the other hand, consistently achieves lower Minimum Description Length (MDL), indicating superior model parsimony. It also produces more semantically coherent topic groupings in most regimes, particularly in mid-sized corpora. These strengths stem from its hierarchical and nonparametric nature, which adapts to the data without requiring a fixed number of topics.

While hSBM's perplexity is higher, this reflects its structural focus rather than a modeling flaw. Its stable performance across scales, combined with stronger coherence, makes it a compelling alternative to LDA when interpretability and compression are prioritized.

Acknowledgment. The hierarchical Stochastic Block Model (hSBM) implementation used in this study is adapted from the open-source library available at https://github.com/martingerlach/hSBM_Topicmodel. All ex-

perimental procedures, benchmarking code, and comparative analysis were independently designed and implemented by the author.

Acknowledgements

Thank you for your guidance and feedback, Prof. Hardin! I've learned so much throughout this process, and this thesis wouldn't be what it is without your support.

I would also like to express my gratitude to the other professors and mentors from both the Mathematics and Computer Science departments who have helped me along the way. My research experience with Prof. Chen has been especially formative in preparing me to become a better researcher.

Lastly, I want to thank my family—my mom, dad, and sister—and my friends for their constant love and unwavering support.

Additionally, I'd like to acknowledge the role of AI tools, especially ChatGPT, for helping me catch mistakes, refine my writing, and occasionally brainstorm ideas. Without them, much more time would have been spent on trivial details. They are also good mentor that patiently explain to me any concept that I don't understand.

Bibliography

Aletras, N., & Stevenson, M. (2013). Evaluating topic coherence using distributional semantics. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 1386–1391.

Anandkumar, A., Foster, D. P., Hsu, D. J., Kakade, S. M., & Liu, Y.-k. (2012). A spectral algorithm for latent dirichlet allocation. *Advances in Neural Information Processing Systems*, *25*.

Barber, M. J. (2007). Modularity and community detection in bipartite networks. *Physical Review E*, *76*(6), 066102.

Barron, A., & Cover, T. M. (1991). Minimum complexity density estimation. *IEEE Transactions on Information Theory*, *37*(4), 1034–1054.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, *3*(Jan), 993–1022.

Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Casella, G., & George, E. I. (1992). Explaining the gibbs sampler. *The American Statistician*, *46*(3), 167–174.

Chang, J., Boyd-Graber, J., Wang, C., Gerrish, S., & Blei, D. M. (2009). Reading tea leaves: How humans interpret topic models. *Advances in Neural Information Processing Systems*, 288–296.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, *41*(6), 391–407.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, *39*(1), 1–38.

Ding, C. (2005). A probabilistic model for latent semantic indexing. *Journal of the American Society for Information Science and Technology*, *56*(6), 597–608. https://doi.org/10.1002/asi.20148

Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, *486*(3-5), 75–174.

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

Geman, S., & Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *6*(6), 721–741.

Gerlach, M., Peixoto, T. P., & Altmann, E. G. (2018). A network approach to topic models. *Science Advances*, *4*(7), eaaq1360.

Griffiths, T. L., & Steyvers, M. (2004). Finding scientific topics. *Proceedings of the National Academy of Sciences*, *101*(suppl 1), 5228–5235. https://doi.org/10.1073/pnas.0307752101

Grootendorst, M. (2022). Bertopic: Neural topic modeling with class-based tf-idf and bertopic representations [Accessed: 2025-04-18].

Grünwald, P. D. (2007). *The minimum description length principle*. MIT Press.

Gutiérrez, I., Gómez, D., Castro, J., Bimber, B., & Labarre, J. (2024). Beyond large language models: Rediscovering the role of classical statistics in modern data science. *2024 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 1–8. https://doi.org/10.1109/FUZZ-IEEE60900.2024.10611766

Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, *57*(1), 97–109.

Hoffman, M., Bach, F., & Blei, D. (2010). Online learning for latent dirichlet allocation. *Advances in Neural Information Processing Systems (NIPS)*, *23*, 856–864.

Hofmann, T. (1999). Probabilistic latent semantic indexing. *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 50–57.

Holland, P. W., Laskey, K. B., & Leinhardt, S. (1983). Stochastic blockmodels: First steps. *Social Networks*, *5*(2), 109–137.

Jelodar, H., Wang, Y., Yuan, C., Feng, X., Jiang, X., Li, Y., & Zhao, L. (2018). Latent dirichlet allocation (lda) and topic modeling: Models, applications, a survey. https://arxiv.org/abs/1711.04305

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., & Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine Learning*, *37*(2), 183–233.

Lau, J. H., Newman, D., & Baldwin, T. (2014). Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, 530–539.

Lewis, C. M., & Grossetti, F. (2022). A statistical approach for optimal topic model identification. *Journal of Machine Learning Research*, *23*(58), 1–20. http://jmlr.org/papers/v23/19-297.html

Maneesh Sahani. (2015). Sampling methods (gatsby ml1 2015) [Lecture 12 handout]. https://www.gatsby.ucl.ac.uk/teaching/courses/ml1-2015/lect12-handout.pdf

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, *21*(6), 1087–1092.

Mimno, D., Wallach, H. M., Talley, E., Leenders, M., & McCallum, A. (2011). Optimizing semantic coherence in topic models. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Mukherjee, A. (2003). Gibbs sampler derivation for latent dirichlet allocation [Lecture Notes].

Neal, R. M. (1993). Probabilistic inference using markov chain monte carlo methods. *Technical Report CRG-TR-93-1, University of Toronto*.

Newman, D., Lau, J. H., Grieser, K., & Baldwin, T. (2010). Automatic evaluation of topic coherence. *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 100–108.

Newman, M. (2010). *Networks: An introduction*. Oxford University Press.

Newman, M. E. J. (2006). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, *103*(23), 8577–8582.

Newman, M. E. J., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E*, *69*(2), 026113.

Ng, A. (2022). Cs229 lecture notes: The em algorithm [Available from the CS229 course website].

Nowicki, K., & Snijders, T. A. B. (2001). Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, *96*(455), 1077–1087.

Peixoto, T. P. (2013). Parsimonious module inference in large networks. *Physical Review Letters*, *110*(14), 148701.

Peixoto, T. P. (2014). Hierarchical block structures and high-resolution model selection in large networks. *Physical Review X*, *4*(1), 011047.

Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, *14*(5), 465–471.

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

Röder, M., Both, A., & Hinneburg, A. (2015). Exploring the space of topic coherence measures. *Proceedings of the 8th ACM International Conference on Web Search and Data Mining*, 399–408.

Touvron, H., et al. (2023). Llama 2: Open foundation and fine-tuned chat models.

Wallach, H. M., Mimno, D., & McCallum, A. (2009). Rethinking LDA: Why priors matter. *UAI*.

Wang, X., & Grimson, E. (2007). Spatial latent dirichlet allocation. In J. Platt, D. Koller, Y. Singer, & S. Roweis (Eds.), *Advances in neural information processing systems* (Vol. 20). Curran Associates, Inc.