Sampling in Computer Vision and Bayesian Nonparametric Mixtures

by

Jason Chang

B.S., Electrical Eng., University of Illinois at Urbana-Champaign, 2007
S.M., Electrical Eng. and Comp. Sci., Massachusetts Institute of Technology, 2009

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Electrical Engineering and Computer Science
at the Massachusetts Institute of Technology

June 2014

Signature of Author: _____

Department of Electrical Engineering and Computer Science
May 21, 2014

Certified by: _____

John W. Fisher III
Senior Research Scientist of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by: _____

Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair of the Committee on Graduate Students

Sampling in Computer Vision and Bayesian Nonparametric Mixtures

by Jason Chang

Submitted to the Department of Electrical Engineering
and Computer Science on May 21, 2014
in Partial Fulfillment of the Requirements for the Degree
of Doctor of Philosophy in Electrical Engineering and Computer Science

Abstract

The field of computer vision focuses on understanding and reasoning about the visual world. Due to the complexity of this problem, researchers often focus on one specific component of this large task, such as segmentation or recognition. This modularized approach necessitates the combination of each separate component, which Bayesian formulations handle in a mathematically consistent framework. Unfortunately, probabilistic formulations are often difficult in computer vision due to the complexity and large dimensionality of data. In this thesis, we demonstrate how efficient Markov chain Monte Carlo (MCMC) sampling techniques can address a subset of these problems.

In the first half of this thesis, we consider the problem of inference in discrete Markov random fields (MRFs) that often occur in segmentation and tracking. We develop the Permutation-based Gibbs-Inspired Metropolis-Hasting (PGIMH) sampling algorithm and show its applicability to a variety of formulations (including curve-length penalties and topology priors). In particle filtering, PGIMH precludes the need to update particle weights or use of sequential importance resampling. Empirical results demonstrate that PGIMH is approximately 10^4 times faster than previous shape sampling approaches and that it improves results in segmentation, boundary detection, and object tracking.

In the second half of this thesis, we focus on inference in the Dirichlet process mixture model (DPMM), which is often slow and cumbersome due to the infinite number of mixture components. We develop a parallel algorithm that samples from the posterior distribution of a DPMM without requiring finite model approximations. This method, called DP Sub-Clusters, essentially fits a two-component mixture model to each regular cluster. These "sub-clusters" are then used to propose splits and merges, resulting in the efficient exploration of the sample space. We show how the developed framework extends to other mixture models, such as the hierarchical Dirichlet process, often used in document analysis. Additionally, we develop the spatially-varying Dirichlet process Gaussian mixture model (SV-DPGMM), which achieves state-of-the-art results in intrinsic image decomposition by leveraging the DP Sub-Cluster algorithm.

By addressing these problems, we demonstrate the applicability of MCMC methods to computer vision, and highlight the importance of designing fast sampling algorithms.

Thesis Supervisor: John W. Fisher III
Title: Senior Research Scientist of Electrical Engineering and Computer Science

Acknowledgments

This thesis would not have been possible without the support of many people at MIT and beyond. First and foremost, I would like to start by thanking my thesis committee, John Fisher, Alan Willsky, and Antonio Torralba. Their comments on the writing and research directions have undoubtedly made it form into one cohesive story.

I am very grateful for John, who has been a great friend and mentor for the past seven years at MIT. John has given me the freedom to pursue any research interest that I have had without ever having to worry about funding. When I look back at my time at MIT, it is abundantly clear that John has shaped both my thought process and research interests. I cannot even begin to imagine what my experience would have been like if it was not for John's guidance.

I would also like to personally thank Alan, who has essentially been a second advisor to me. I met with Alan to discuss research on a weekly basis in the SSG grouplets, and I was always amazed at how much he knew about every subject that came up, whether research-related or otherwise. Grouplets with Alan have helped me become a better presenter and have exposed me to a wealth of other interesting research topics. I will always remember Alan's comments that were prefaced by his "synapses firing," since they were guaranteed to be something very insightful.

This thesis would also not have been possible without the help of collaborators. Donglai Wei and I worked tirelessly on the temporal superpixels work, solving a seemingly endless flow of problems that kept arising. Unfortunately, our optimization scheme was too much of a departure from the sampling goals of this thesis to include. Randi Cabezas helped a lot with the intrinsic image decomposition and even created an entire dataset that we never ended up using. I would also like to acknowledge all the grouplet members I have had over the years. In particular, Matt Johnson, Dahua Lin, Ying Liu, James Saunderson, and Kush Varshney were recurring grouplet participants. Their combined expertise in computer vision, machine learning, probabilistic modeling, and optimization is simply astounding.

The other SLI group members, Christopher Dean, Oren Freifeld, Zoran Dzunic, Bonny Jain, Hosseein Mobahi, Giorgos Papachristoudis, Guy Rosman, Michael Siracusa, Julian Straub, and Sue Zheng have also been very helpful in reading over early drafts of research papers and sitting through many presentations. The SLI group has grown from the four members when I started to the current large, twelve member group, and I'm glad to see that the culture remains.

I've had the pleasure of sharing an office with a fun group of friends: Randi Cabezas, Michal Depa, Andrew Mastin, Julian Straub, Archana Venkataraman, and Sue Zheng. We've had many dart fights and random discussions over the past years, and I could

always count on one of them being in the office regardless of the time or day.

Lastly, I would like to thank my family and friends. In particular, my parents and sister have provided the emotional support and love to get through my Ph.D. journey. They have taught me to value what is important in life and that anything is possible when you set your mind on a goal. Thank you for always being there for me.

Different aspects of this thesis were partially supported by the Office of Naval Research Multidisciplinary Research Initiative program, award N000141110688, the Army Research Office Multidisciplinary Research Initiative program award W911NF-11-1-0391, the Defense Advanced Research Projects Agency, award FA8650-11-1-7154, and Shell via the MIT Energy Initiative.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

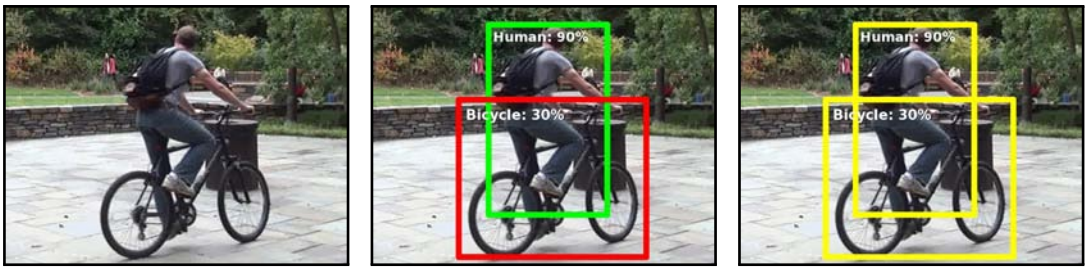
Introduction

The field of computer vision attempts to develop algorithms to better understand the visual world. Successful computer vision algorithms are applicable to a wide variety of problems. For example, in robotics, understanding the surrounding environment enables robots to navigate and interact with the physical world. In surveillance, streaming videos could greatly benefit from autonomous analysis in anomaly detection, object recognition, and object tracking. In this thesis, we consider the problem of *statistical* computer vision by demonstrating that reasoning over statistics of complex distributions in computer vision is not only useful, but feasible for a variety tasks.

One of the ultimate goals of computer vision is to semantically understand scenes. This task is difficult because of the intricate dependencies that exist amongst objects in the scene, only which are obfuscated even more by the camera sensor that translates the semantically-meaningful world into millions of tiny pixels. Due to the complexity of the overarching problem, computer vision research often focuses on one specific aspect of the problem, such as image segmentation, object recognition, or object tracking. Obtaining a hard decision for any of these individual task is typically easier than finding the uncertainty in the decision. Furthermore, when the individual task is treated as the end goal, a single hard decision may suffice. For example, in object detection, knowing that there are three detected humans in the scene may be satisfactory.

However, when considering the scene as a whole, reliably combining the individual solutions of the separate problems into a unified decision becomes more difficult. For example, consider the task of detecting whether a human is riding a bicycle in the image in Figure 1.1. If the human is detected but the bicycle is missed because of the threshold of the detector, there is little hope for an algorithm to declare that a human is riding a bicycle.. Alternatively, if *probabilities* of detection are used, the uncertainty in the detections can be used by the higher-level reasoning to make a more informed decision. As such, it is critical to capture the *uncertainty* of the individual lower-level tasks (such as object and gesture recognition) before declaring the final decision. A unified Bayesian formulation of the individual tasks provides a mathematically consistent framework for such an integration.

The benefit of reasoning about the uncertainty of a solution also exists in the individual tasks of computer vision. For example, consider image segmentation, which aims



(a) Original Image (b) Hard Declaration (c) Probabilistic Declaration

Figure 1.1: Detecting whether a human is riding a bicycle. Green and red bounding boxes represent hard positive and negative detections, respectively, and yellow bounding boxes represent probabilistic detections. (b) A bike is not found, resulting in the declaration that there is no human riding a bike. (c) A probabilistic detection defers the declaration to the higher-level reasoning.

(a) Original Image (b) Segmentation 1 (c) Segmentation 2

Figure 1.2: Two human-annotated segmentations from [88].

to divide the visual world into different, semantically-meaningful objects. This task can be formalized as assigning a discrete label to each pixel of the image, where two pixels with the same label belong to the same object. An example of the image segmentation task from the Berkeley Segmentation Dataset [88] is shown in Figure 1.2 for two different human annotators. While there is clearly a high amount of agreement between the human segmentations, there is also quite a bit of variability. Typical approaches to image segmentation attempt to find a solution by optimizing some user-specified energy. However, regardless of how well the energy approximates the actual human visual system, an optimization scheme cannot capture the *variability* in results that exists in human processing.

■ 1.1 A Bayesian Approach

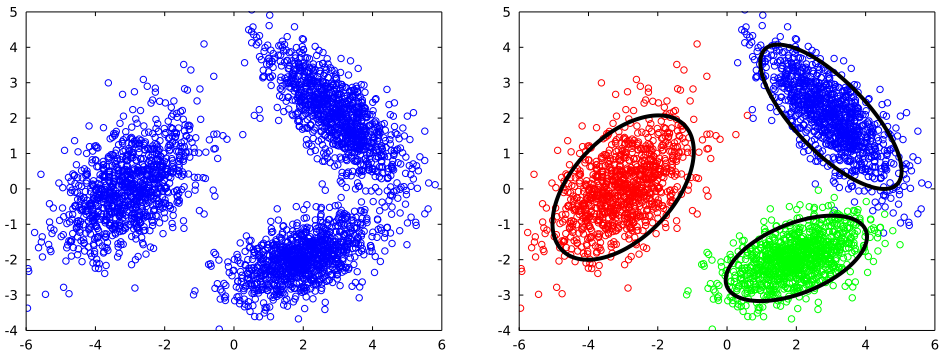
The Bayesian formulation provides a mechanism where the uncertainty in a solution is reasoned about in a mathematically consistent framework. Generally speaking, a full model of uncertainty is captured by a probability distribution; however, in the context

of computer vision, such distributions are complex and exist in very high-dimensional spaces. For example, in a reasonably sized image of 640×480, there exists 3×10^5 pixels and $2^{3 \times 10^5}$ possible segmentations into two regions. This leads to certain challenges in exploiting Bayesian models.

One tool that is often used in complicated distributions (such as those encountered in computer vision) is Monte Carlo simulation. Monte Carlo simulation relies on the law of large numbers, which essentially states that the expectation of any function of a random variable is well-approximated with the average of the function evaluated at sample realizations of the random variable. Furthermore, when distributions cannot be sampled from directly, Markov chain Monte Carlo (MCMC) methods can be used. MCMC methods draw a sample from a user-specified, *target* distribution by simulating a specific Markov chain. Under certain mild conditions, the state of the Markov chain will converge to its stationary distribution after an adequate amount of time. Furthermore, as we review in Chapter 2, one can guarantee that the stationary distribution is exactly the target distribution of interest by ensuring certain conditions. At the heart of many MCMC problems is the design of transition distributions that reduce the convergence time to the stationary distribution. This is especially true when scaling the algorithm to large amounts of data.

While the use of MCMC methods is fairly common in a variety of domains, challenges exist when applying them to computer vision problems. MCMC methods are typically slower than optimization procedures since they must reason about the underlying distribution instead of finding the single point that corresponds to the best configuration. This computation is exacerbated in the computer vision domain due to the large scale of the data. Moreover, quantities of interest in computer vision are often quite complex and exhibit intricate dependencies. For example, representing the boundary of an object and the temporal evolution that couples the appearance and shape is challenging to represent in a probabilistic framework.

In this thesis, we demonstrate that some challenges in applying MCMC methods to computer vision can be addressed by developing efficient algorithms. As we shall see, these methods are extensible to fields outside of computer vision as well. For example, in probabilistic modeling, one common problem focuses on clustering sets of data with a mixture model (e.g., see Figure 1.3). In such scenarios, it is already common to use MCMC methods such as Gibbs sampling [40] or Metropolis-Hastings sampling [51] to infer information about distributions instead of relying on point-estimates. However, current methods can take quite a long time to converge, and/or require approximations so that the methods can be parallelized and scaled to the large size of the data. We address these issues in this thesis. Consequently, the developed algorithms are broadly applicable to problems in both computer vision and machine learning.



(a) Data Observations (b) Colored Labeling

Figure 1.3: An example of the mixture model.

1.2 Thesis Outline and Contributions

In this thesis, we address some of the aforementioned issues in two types of discrete-labeling problems. The first is when the labels of interest are interdependent and assumed to take on values in a finite set. The interdependence of the labels occur in many formulations when there is an assumed *smoothness* in the labels. For example, we will focus on the problems of image segmentation and object tracking, where labels are known to be smooth in space and time. The second labeling problem that we consider is when the labels correspond to cluster assignments in a mixture model. These labels are assumed to be conditionally *independent*, and can take on values from an unknown, potentially infinite, number of labels. For example, in Bayesian nonparametrics, the Dirichlet process mixture model and the Hierarchical Dirichlet process both fit within this framework. For each of the two problems, we develop fast MCMC sampling methods and show their application to various computer vision and machine learning tasks.

The methods that we will develop exploit the Metropolis-Hastings algorithm [51], which is an MCMC framework that allows one to control the stationary distribution of the Markov chain. As we shall see, we will augment each model with additional auxiliary variables. While one might expect that the expanded sample space may complicate the problem, we show that clever choices of auxiliary variables that are tailored to the problem at hand can drastically help the convergence of the Markov chain. We now give an overview of the thesis while summarizing the specific problems and contributions.

Chapter 2: Background

We begin the thesis with a discussion of relative background material in Chapter 2. After motivating Bayesian frameworks with the use of prior information, a formal setup of the problem of posterior inference is stated. We then discuss the class of tractable,

conjugate distributions, and detail the distributions used in the rest of this thesis. The background chapter continues to introduce directed graphical models and undirected graphical models (i.e., Markov random fields). Following this discussion, we outline various MCMC sampling frameworks (Gibbs, Metropolis-Hastings, and Reversible-Jump MCMC), importance sampling, and particle filtering.

Relevant sections related to each chapter are then discussed, including level-set methods, shape sampling methods, digital topology, and finite mixture models. We conclude the chapter with a detailed description of three nonparametric processes that are used throughout much of this thesis: the Gaussian process, the Dirichlet process, and the hierarchical Dirichlet process. We review previous work and current inference algorithms for each of these stochastic process.

Chapter 3: Implicit Shapes and Discrete MRFs

In Chapter 3, we develop an MCMC sampling method for implicitly defined shapes represented as a Markov random field (MRF) over a discrete set of labels. The proposed method is called the Permutation-based Gibbs-Inspired Metropolis-Hastings (PGIMH) algorithm, and can be used for nearly any distribution that imposes local constraints on the labels. For example, we show how PGIMH can be applied to the commonly-used Ising or Potts model in MRFs or for the curve-length penalty in level-set methods. We additionally show how PGIMH can be used for global constraints that can be computed locally, such as constraints on digital topology or desired area.

PGIMH makes large, localized changes to the set of labels by augmenting the sample space with an explicit ordering of the pixels. The proposed algorithm is similar to blocked Gibbs sampling in that it samples a localized group of pixels simultaneously. Also like blocked Gibbs sampling, PGIMH simplifies to traditional Gibbs sampling when the localized changes only act on a single label. However, the complexity of the PGIMH algorithm scales linearly with the block size as opposed to scaling exponentially as in blocked Gibbs sampling.

We show that using a sampling method such as PGIMH naturally transforms any segmentation algorithm into a probabilistic boundary detection algorithm. Moreover, boundary detections obtained using this approach vastly outperform hard boundary declarations from their optimization-based counterparts on the Berkeley Segmentation Dataset [88]. Results in low signal-to-noise-ratio images are also improved when considering marginal statistics. We conclude with a demonstration on how to control the topology of the underlying 2D shape to satisfy prior information.

Chapter 4: Shape Dynamics in Object Tracking

Next, we address the problem of object tracking in Chapter 4. Object tracking is a particular application where describing uncertainty in distributions is critical. We show that reasoning about this uncertainty is possible by relying on the development of PGIMH in Chapter 3. Specifically, we present a generative model for scenes that models the support and ordering of objects in videos. A model for the motion of the

camera and each object in the scene is formulated as a Gaussian process, and coupled with the evolution of both the shape and appearance. Additionally, we enforce strict topology constraints on each object through PGIMH.

We present an efficient sampling-based approach to perform posterior inference. The developed particle filter departs from traditional approaches by not needing weight updates or sequential importance resampling techniques. Experimental results show that the proposed model outperforms other tracking and segmentation algorithms on the SegTrack dataset [118].

Chapter 5: Parallel Split-Merge MCMC for the DPMM

Next, in Chapter 5, we switch to the discrete labeling problem in mixture model analysis. We develop a new sampling method for mixture models with a directed focus on the infinite Dirichlet process mixture model (DPMM). Unlike many previous approaches, the proposed approach, called the DP Sub-Cluster algorithm, can be parallelized without finite model approximations, and can be used in models with non-conjugate priors. One advantage of the DP Sub-Cluster algorithm is that large split and merge moves are efficiently proposed to drastically improve convergence. The model is augmented with auxiliary *sub-clusters* that fit a 2-component mixture model to each regular cluster. These sub-clusters are then used to propose likely splits. Unlike all previous DPMM split proposals, the DP Sub-Cluster splits are improved with each iteration by explicitly instantiating the sub-clusters.

Our empirical results show that the DP Sub-Cluster algorithm outperforms all previous sampling approaches across multiple data types (e.g., multivariate-Gaussian observations and multinomial observations) in both synthetic and real-world datasets. In fact, the DP Sub-Cluster algorithm converges to a better solution, and does so approximately 10–10^3 times faster than other sampling algorithms.

Chapter 6: Parallel Split-Merge MCMC for the HDP

We extend the DP Sub-Cluster algorithm to the hierarchical Dirichlet process (HDP) in Chapter 6, with a focus on the problem of topic-modeling in document analysis. Some additional tools are needed to extend the model properly to HDPs. For example, this extension is slightly complicated by the additional top-level Dirichlet process which represents the posterior on global topic-weights.

One interesting insight of this chapter is that local splits and merges, which only act on a pair of clusters, are insufficient when distributions are highly overlapped (e.g., in topic modeling). Consequently, we develop *global* split and merge moves that alter the DP Sub-Cluster algorithm by jointly changing all labels at once.

We demonstrate on a variety of corpora that the proposed HDP Sub-Cluster algorithm converges more reliably than current HDP sampling algorithms. Additionally, we have found through the comparisons that cross-validation techniques do not accurately indicate convergence of MCMC algorithms. In our experiments, the cross-validation metrics converge quickly, but the inferred latent representation (e.g., the number of

clusters) does not converge for a considerable amount of time after that.

Chapter 7: Intrinsic Image Decomposition via the SV-DPGMM

Lastly, we return to computer vision and show an application of the DP Sub-Cluster algorithm to the problem of intrinsic image decomposition. We develop the spatially-varying Dirichlet process Gaussian mixture model (SV-DPGMM), a new Bayesian non-parametric model that allows the mixture parameters to change *jointly* in space via a Gaussian process. When the SV-DPGMM is applied to the problem of intrinsic image decomposition, the mixture model captures the reflectance image, and the Gaussian process captures the shading image.

We develop efficient inference algorithms based on the DP Sub-Cluster algorithm that marginalize over both the Gaussian process and the mixture parameters. Our results outperform similar models working in the image domain on the MIT Intrinsic Image Dataset [48], and is comparable with models that infer complete 3D models.

Chapter 8: Conclusion and Future Work

Finally, in Chapter 8, we summarize the work and contributions in this thesis. We give some recommendations for extensions and future work pertaining to the developed PGIMH and Sub-Cluster sampling algorithms.

Appendices

Some details of derivations that would detract from the flow of the narrative have been included in Appendices A–D. These typically include a considerable amount of algebra in deriving relationships (e.g., Hastings ratios in split/merge steps).

List of Symbols

A list of symbols used in each chapter is included at the end of this thesis for convenience.

Chapter 2

Background

MANY computer vision algorithms can be interpreted in a probabilistic framework. In this chapter, we review relevant background material to this thesis. We begin by motivating the Bayesian formulation, followed by discussing the tractable class of conjugate distributions. A brief introduction is then given for graphical models, Monte Carlo sampling approaches, level-set methods, digital topology, and finite mixture models. We conclude with a detailed discussion about Gaussian processes, Dirichlet processes, and hierarchical Dirichlet processes.

■ 2.1 Posterior Inference

In many applications, the quantity of interest (denoted z) is not directly observable, but related observations (denoted x) are readily available. It may additionally be known that the distribution of the observations *conditioned* on the hidden variable, $p(x|z)$, follows a specific function. In such a setting, one is often concerned with finding the *posterior* distribution of the hidden variables conditioned on the observations, $p(z|x)$. We note that we use $p(x)$ to abstractly denote the distribution of the random variable x. This is sometimes denoted as $p_X(x)$, where the random variable is X, and x is the specific value taken by X. We do not make such a distinction here.

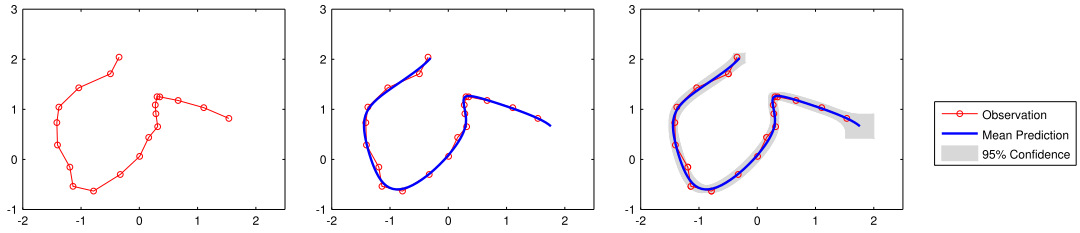
Using Bayes' rule, this posterior distribution can be expressed as

$$p(z|x) = p(x|z)\frac{p(z)}{p(x)}. \tag{2.1}$$

Here, we note that $p(x)$ is the marginal data likelihood, and does not depend on the hidden, *latent* variables. We can therefore express the posterior distribution up to a scale factor as

$$p(z|x) \propto p(z)p(x|z). \tag{2.2}$$

This decomposition lends itself to a readily interpretable explanation: $p(z)$ is the *prior* distribution and $p(x|z)$ is the data likelihood distribution. In particular, the prior distribution captures information about the latent variable that is known without any observations, and the data likelihood distribution captures how well the data is explained by a particular realization of the latent variable.



(a) Observations (b) Mean Prediction (c) 95% Confidence

Figure 2.1: A simple example of posterior inference in a tracking problem. Measurements are assumed to be generated from the ground truth locations subject to i.i.d. additive Gaussian noise. A Gaussian process prior is used on the trajectory.

We now consider a simple example of posterior inference that will be used for the remainder of the background section.

Example 2.1.1 (GPS Tracking)**.** Consider the problem of tracking the precise location of a vehicle. While the exact latitude and longitude of the vehicle cannot be directly obtained, a GPS unit can give very accurate *measurements* of this quantity. An illustration of such measurements is shown in Figure 2.1a.

The noise characteristics of the measurements are reflected in the data likelihood term, $p(x|z)$. One simple assumption that may be used is that the noise is independent and identically distributed (i.i.d.) Gaussian noise. Even though the observations are quite jagged, the true location of the vehicle may be known to follow a smooth trajectory. Such information can be encoded into the prior distribution, $p(z)$, using, for example, a Gaussian process. As shown in Section 2.9.1, one can then perform maximum *a posteriori* (MAP) estimation, which seeks to find

$$z^* = \arg\max_z p(z|x). \tag{2.3}$$

The MAP estimate of z is shown in Figure 2.1b.

Alternatively, it may be worthwhile to know the *confidence* of the prediction, instead of simply finding the optimal estimate. In this particular example, finding the confidence interval can be done in closed-form, and is illustrated in Figure 2.1c.

The $p(x)$ term in Equation (2.1) normalizes the function so that $p(z|x)$ properly integrates to 1. However, in most situations, only $p(z)$ and $p(x|z)$ are assumed to be known. The marginal data likelihood, $p(x)$, which can be expressed as

$$p(x) = \int p(x, z)dz = \int p(x|z)p(z)dz, \tag{2.4}$$

cannot typically be expressed analytically. In such situations, inferring the entire posterior distribution may not be tractable, but finding the MAP solution may still be possible since the optimization scheme is independent of the normalization. However,

when other statistics of the posterior distribution are desired, one must take a different approach. In the Bayesian framework, there are two broad approaches to address this issue, often which are both exploited: choose priors and likelihoods that lend themselves to efficient inference, and perform approximate inference. We now overview a subset of each of these techniques.

■ 2.2 Conjugate Priors

In this section, we consider a small class of prior and data likelihood distributions that result in analytical posterior distributions. We begin by defining the concept of *exchangeability*.

Definition 2.2.1 (Exchangeability)**.** Let $\{x_1, \ldots, x_N\}$ denote N observations drawn from the joint distribution, $p(x_1, \ldots, x_N)$. Let $P \in \mathcal{P}$ define a permutation of the integers $\{1, \ldots, N\}$ from the set of all permutations, $\mathcal{P}$, and P_i denote the i^{th} number in the permutation. If the following conditions holds

$$p(x_1, x_2, \ldots, x_{N-1}, x_N) = p(x_{P_1}, x_{P_2}, \ldots, x_{P_{N-1}}, x_{P_N}), \quad \forall P \in \mathcal{P}, \tag{2.5}$$

then the random variables, $\{x_1, \ldots, x_N\}$, are said to be exchangeable.

An exchangeable set of random variables can be thought of as an unordered set. de Finetti proved that for any set of exchangeable variables, there must exist a prior distribution that decouples the observations into the product of independent and identical distributions.

Theorem 2.2.1 (de Finetti's Theorem)**.** *Let $\{x_1, \ldots, x_N\}$ denote a set random variables. If these random variables are exchangeable, there must exist a distribution, $p(\theta)$, such that the joint distribution can be expressed as*

$$p(x_1, \ldots, x_N) = \int p(\theta) \prod_{i=1}^{N} p(x_i|\theta) d\theta. \tag{2.6}$$

Proof. See [52]. □

Consider the case of de Finetti's theorem where the distribution, $p(x_i|\theta)$, is known. For example, suppose $p(x_i|\theta)$ is Gaussian and θ captures the mean and variance of the Gaussian. Additionally, prior knowledge of the parameter can be encoded into $p(\theta)$. The posterior distribution of interest is then $p(\theta|x)$, where x without subscripts implies the entire set of variables, $\{x_1, \ldots, x_N\}$.

We first introduce some notation. Assume that the data likelihood takes on a specific functional form parametrized by θ. We denote this with

$$p(x_i|\theta) \triangleq f_x(x_i; \theta). \tag{2.7}$$

Furthermore, assume that the prior distribution takes on a different functional form parametrized by λ:

$$p(\theta) \triangleq f_\theta(\theta; \lambda). \tag{2.8}$$

The posterior distribution can then be expressed as

$$p(\theta|x) = p(x|\theta)\frac{p(\theta)}{p(x)} \propto p(\theta)p(x|\theta) = f_\theta(\theta;\lambda)\prod_{i=1}^{N} f_x(x_i;\theta). \tag{2.9}$$

In general, the form of the posterior distribution will not be in the same parametric form as the prior, f_θ. However, for a certain class of paired priors and data likelihoods, the posterior distribution actually stays within the same family of functions as the prior. This class of paired distributions is typically referred to as the class of *conjugate distributions*, and the prior distribution that pairs with a particular data likelihood is called the *conjugate prior*. This relationship is precisely stated in the following definition.

Definition 2.2.2 (Conjugate Prior)**.** Let $p(x_i|\theta) = f_x(x_i;\theta)$ and $p(\theta) = f_\theta(\theta;\lambda)$ denote a particular data likelihood and prior distribution, respectively. If the following relationship holds

$$p(\theta|x) = f_\theta(\theta; \lambda^*(x)), \tag{2.10}$$

then f_θ is said to be conjugate to f_x. The *posterior hyper-parameters*, $\lambda^*(x)$, typically depend on the set of observations, x, and the prior hyper-parameters, λ.

Because $\lambda^*(x)$ can often be expressed analytically, posterior inference in conjugate distributions can be done in closed-form. In many situations, other quantities of interest can also be expressed analytically. We review two such quantities here.

■ 2.2.1 Marginal Data Likelihood

The data likelihood marginalizing over the parameters, $p(x)$, is typically intractable to due to the integral over θ:

$$p(x) = \int p(x|\theta)p(\theta)d\theta = \int f_x(x;\theta)f_\theta(\theta;\lambda)d\theta. \tag{2.11}$$

However, this expression simplifies to the following when a conjugate prior is used:

$$p(x) = \frac{p(\theta)}{p(\theta|x)}p(x|\theta) = \frac{f_\theta(\theta;\lambda)}{f_\theta(\theta;\lambda^*(x))}f_x(x;\theta). \tag{2.12}$$

While it seems that this equation depends on a particular value of θ, the analytical expression for any particular class of conjugate distributions will not depend on θ. With a slight abuse of notation, we denote the above distribution as

$$p(x) \triangleq f_x(x;\lambda), \tag{2.13}$$

which is generally a different functional form from $f_x(x;\theta)$.

■ 2.2.2 Predictive Distribution

The predictive distribution expresses the probability of a single, new observation conditioned on all current observations. We denote the predictive distribution as $p(\hat{x}|x)$, where $\hat{x}$ denotes the new observation. We note the following relationship with conjugate priors:

$$p(\hat{x}|x) = \int p(\hat{x}|\theta)p(\theta|x)d\theta = \int f_x(\hat{x};\theta)f_\theta(\theta;\lambda^*(x))d\theta. \tag{2.14}$$

Notice that this expression is of the same form as Equation (2.11). Thus, we can directly conclude from Equation (2.13) that the predictive distribution is

$$p(\hat{x}|x) = f_x(x;\lambda^*(x)). \tag{2.15}$$

■ 2.3 Conjugate Distributions

In this section, we briefly review some discrete and continuous conjugate distributions used throughout this thesis. For each conjugate pair, we derive the posterior hyperparameter expressions and the form of the marginal data likelihood, keeping in mind that the predictive distribution is expressed as the combination of these two through Equation (2.15).

■ 2.3.1 Categorical Distribution

A categorical distribution is a discrete distribution over a fixed alphabet. A discrete random variables x is said to be drawn from a D-dimensional categorical distribution parametrized by π if x is distributed according to

$$x \sim \text{Cat}(x;\pi) \triangleq \begin{cases} \pi_d, & x \in \{1,\dots,D\} \\ 0, & \text{otherwise} \end{cases}. \tag{2.16}$$

The parameter $\pi \in (0,1)^D$ is a D-dimensional vector that sums to 1. We note that we occasionally denote a categorical distribution with the explicit discrete form of

$$\text{Cat}(x;\pi) = \sum_{d=1}^{D} \pi_d \mathbb{1}[x = d]. \tag{2.17}$$

■ 2.3.2 Multinomial Distribution

A multinomial distribution is a generalization of the categorical distribution to multiple trials. In particular, it is a joint distribution over counts of occurrences for N independent, categorical trials. If x is drawn from a D-dimensional multinomial distribution for N trials, $\text{Mult}(x;\pi,N)$ represents the probability that element d was selected x_d

times for all dimensions, d. The probably mass function can be expressed as

$$x \sim \text{Mult}(x; \pi, N) \triangleq N! \prod_{d=1}^{D} \frac{\pi_d^{x_d}}{x_d!}. \tag{2.18}$$

Again, $\pi \in (0,1)^D$ is a D-dimensional vector that sums to 1. Since x represents counts rather than indices (as was the case in the Categorical distribution), x is now a D-dimensional vector of non-negative integers that sums to N.

A Categorical distribution is a special case of the Multinomial distribution with one trial ($N = 1$), and where the Categorical random variable takes on the value of the specific dimension of the non-zero value. As such, the literature often refers to a Categorical distribution as a Multinomial distribution. We will be precise in our naming since we will make use of both the Categorical and Multinomial distributions.

■ 2.3.3 Dirichlet Distribution

The conjugate distribution to the Multinomial distribution is the Dirichlet distribution. Furthermore, because the Categorical distribution is a special case of the Multinomial, the Dirichlet is also conjugate to the categorical distribution. A D-dimensional Dirichlet distribution is distributed according to

$$\pi \sim \text{Dir}(\pi; \alpha_1, \ldots, \alpha_D) \triangleq \Gamma(A) \prod_{d=1}^{D} \frac{\pi_d^{\alpha_d - 1}}{\Gamma(\alpha_d)}, \tag{2.19}$$

where all α's are positive real numbers, and $A \triangleq \sum_{d=1}^{D} \alpha_d$. We additionally refer to a *symmetric* Dirichlet distribution as a special case of the Dirichlet distribution where all α values are equal. A variable that is drawn from a symmetric Dirichlet distribution is denoted with

$$\pi \sim \text{Dir}(\pi; \alpha). \tag{2.20}$$

The vector π sums to 1, and the parameters, α, represent pseudo-counts of the prior.

Categorical Conjugacy

The Dirichlet distribution is a conjugate prior of the categorical distribution. Assuming $\{x_1, \ldots, x_N\}$ are independent samples from a categorical distribution, the posterior

distribution can be expressed as

$$\begin{aligned} p(\pi|x) \propto p(\pi) \prod_{i=1}^{N} p(x_i|\pi) &= \left[\Gamma(A) \prod_{d=1}^{D} \frac{\pi_d^{\alpha_d - 1}}{\Gamma(\alpha_d)}\right] \left[\prod_{i=1}^{N} \pi_{x_i}\right] \\ &= \left[\Gamma(A) \prod_{d=1}^{D} \frac{\pi_d^{\alpha_d - 1}}{\Gamma(\alpha_d)}\right] \left[\prod_{d=1}^{D} \pi_d^{N_d}\right] \\ &= \Gamma(A) \prod_{d=1}^{D} \frac{\pi_d^{\alpha_d + N_d - 1}}{\Gamma(\alpha_d)} \\ &\propto \mathrm{Dir}(\pi; \alpha_1 + N_1, \ldots, \alpha_D + N_D). \end{aligned} \tag{2.21}$$

Thus, the posterior hyper-parameters, $\alpha^*(x)$, are as follows:

$$\alpha_d^*(x) = \alpha_d + N_d, \quad \forall d \in \{1, \ldots, D\}. \tag{2.22}$$

Using Equation (2.12), the marginal data likelihood can then be expressed as

$$\begin{aligned} p(x) &= \frac{\Gamma(A) \prod_{d=1}^{D} \frac{\pi_d^{\alpha_d - 1}}{\Gamma(\alpha_d)}}{\Gamma(A+N) \prod_{d=1}^{D} \frac{\pi_d^{\alpha_d + N_d - 1}}{\Gamma(\alpha_d + N_d)}} \prod_{d=1}^{D} \pi_d^{N_d} \\ &= \frac{\Gamma(A)}{\Gamma(A+N)} \prod_{d=1}^{D} \frac{\Gamma(\alpha_d + N_d)}{\Gamma(\alpha_d)} \\ &\triangleq \mathrm{DirCat}(x; \alpha_1, \ldots, \alpha_D). \end{aligned} \tag{2.23}$$

Using Equation (2.15) and a bit of algebra, the predictive distribution can then be expressed as

$$p(\hat{x}|x) = \mathrm{DirCat}(\hat{x}; \alpha_1^*(x), \ldots, \alpha_D^*(x)) = \frac{\alpha_{\hat{x}}^*(x)}{A^*(x)}, \tag{2.24}$$

where $A^*(x) = \sum_{d=1}^{D} \alpha_d^*(x)$.

Multinomial Conjugacy

The Dirichlet distribution is also conjugate to the multinomial distribution. Assuming $\{x_1, \ldots, x_N\}$ are drawn from a multinomial distribution of N trials, the posterior

distribution can be expressed as

$$\begin{aligned} p(\pi|x) \propto p(\pi)\prod_{i=1}^{N} p(x_i|\pi) &= \left[\Gamma(A)\prod_{d=1}^{D}\frac{\pi_d^{\alpha_d-1}}{\Gamma(\alpha_d)}\right]\left[N!\prod_{d=1}^{D}\frac{\pi_d^{x_d}}{x_d!}\right] \\ &= \Gamma(A)N!\prod_{d=1}^{D}\frac{\pi_d^{\alpha_d+x_d-1}}{\Gamma(\alpha_d)x_d!} \\ &\propto \mathrm{Dir}(\pi;\alpha_1+x_1,\ldots,\alpha_D+x_D). \end{aligned} \tag{2.25}$$

Thus, the posterior hyper-parameters, α^*, are as follows:

$$\alpha_d^*(x) = \alpha_d + x_d, \quad \forall d \in \{1,\ldots,D\}. \tag{2.26}$$

Using Equation (2.12), the marginal data likelihood can then be expressed as

$$\begin{aligned} p(x) &= \frac{\Gamma(A)\prod_{d=1}^{D}\frac{\pi_d^{\alpha_d-1}}{\Gamma(\alpha_d)}}{\Gamma(A+N)\prod_{d=1}^{D}\frac{\pi_d^{\alpha_d+x_d-1}}{\Gamma(\alpha_d+x_d)}}N!\prod_{d=1}^{D}\frac{\pi_d^{x_d}}{x_d!} \\ &= \frac{N!}{\prod_{d=1}^{D}x_d!}\frac{\Gamma(A)}{\Gamma(A+N)}\prod_{d=1}^{D}\frac{\Gamma(\alpha_d+x_d)}{\Gamma(\alpha_d)} \\ &\triangleq \mathrm{DirMult}(x;\alpha_1,\ldots,\alpha_D). \end{aligned} \tag{2.27}$$

Using Equation (2.15), the predictive distribution can then be expressed as

$$p(\hat{x}|x) = \mathrm{DirMult}(\hat{x};\alpha_1^*(x),\ldots,\alpha_D^*(x)) = \frac{\hat{N}!}{\prod_{d=1}^{D}\hat{x}_d!}\frac{\Gamma(A^*(x))}{\Gamma(A^*(x)+\hat{N})}\prod_{d=1}^{D}\frac{\Gamma(\alpha_d^*(x)+\hat{x}_d)}{\Gamma(\alpha_d^*(x))}. \tag{2.28}$$

■ 2.3.4 Multivariate Gaussian Distribution

The one-dimensional Gaussian (or Normal) distribution is a continuous distribution with support on the entire real line. The multivariate Gaussian distribution is a generalization to D dimensions that has support on $\mathbb{R}^D$. The distribution is parametrized by a mean vector, $\mu \in \mathbb{R}^D$, and a positive-definite covariance matrix, $\Sigma \in \mathbb{R}^{D\times D}$, and can be expressed as

$$x \sim \mathcal{N}(x;\mu,\Sigma) \triangleq (2\pi)^{-D/2}\,|\Sigma|^{-1/2}\,e^{-\frac{1}{2}(x-\mu)^\top\Sigma^{-1}(x-\mu)} \tag{2.29}$$

Self Conjugacy on Mean

The Gaussian distribution exhibits some interesting properties. For example, because Gaussian functions are closed under products and convolutions, a Gaussian prior on

the mean is conjugate to a Gaussian likelihood.

We first show that the joint likelihood over multiple independent and identically distributed (i.i.d.) can be calculated via sufficient statistics. The product of likelihoods can be expressed as

$$\prod_{i=1}^{N} \mathcal{N}(x_i; \mu, \Sigma) = (2\pi)^{-ND/2} \left|\Sigma\right|^{-N/2} \exp\left[-\frac{1}{2}\sum_{i=1}^{N}(x_i-\mu)^\top \Sigma^{-1}(x_i-\mu)\right]. \tag{2.30}$$

By denoting the sufficient statistics as $T_1 = \sum_i x_i$ and $T_2 = \sum_i x_i x_i^\top$, the term inside the exponential can be expressed as

$$-\tfrac{N}{2}\left(\mu - \tfrac{T_1}{N}\right)^\top \Sigma^{-1}\left(\mu - \tfrac{T_1}{N}\right) + \tfrac{1}{2N} T_1^\top \Sigma^{-1} T_1 - \tfrac{1}{2}\operatorname{tr}(\Sigma^{-1} T_2) \tag{2.31}$$

With some algebra, the product of Gaussians then simplifies to

$$\prod_{i=1}^{N} \mathcal{N}(x_i; \mu, \Sigma) = N^{-\frac{D}{2}} \left((2\pi)^D \left|\Sigma\right|\right)^{\frac{1-N}{2}} \exp\left[\tfrac{1}{2N} T_1^\top \Sigma^{-1} T_1 - \tfrac{1}{2}\operatorname{tr}(\Sigma^{-1} T_2)\right] \mathcal{N}\left(\mu; \tfrac{T_1}{N}, \tfrac{\Sigma^x}{N}\right) \tag{2.32}$$

We now show that the Gaussian prior on the mean is conjugate to the Gaussian likelihood. More precisely, a multivariate-Gaussian prior on the mean is chosen to be

$$p(\mu) = \mathcal{N}(\mu; \theta, \Delta)\,, \tag{2.33}$$

where θ is the prior mean of the mean parameter, and Δ is the prior covariance of the mean parameter. Using Equation (2.32), this choice of prior results in the following posterior

$$p(\mu|x) \propto p(\mu) \prod_{i=1}^{N} p(x_i|\mu) \propto \mathcal{N}(\mu; \theta, \Delta)\, \mathcal{N}\left(\mu; \tfrac{T_1}{N}, \tfrac{\Sigma^x}{N}\right) = \mathcal{N}(\mu; \theta^*, \Delta^*)\,, \tag{2.34}$$

where the posterior hyper-parameters, θ^* and Δ^* are

$$\theta^* = (\Delta^{-1} + N\Sigma^{-1})^{-1}(\Delta^{-1}\theta + \Sigma^{-1} T_1) \tag{2.35}$$

$$\Delta^* = (\Delta^{-1} + N\Sigma^{-1})^{-1} \tag{2.36}$$

Using Equation (2.12) and a bit of algebra, the marginal data likelihood can then be expressed as

$$p(x) = N^{-\frac{D}{2}} \left((2\pi)^D \left|\Sigma\right|\right)^{\frac{1-N}{2}} \exp\left[\tfrac{1}{2N} T_1^\top \Sigma^{-1} T_1 - \tfrac{1}{2}\operatorname{tr}(\Sigma^{-1} T_2)\right] \mathcal{N}\left(\tfrac{T_1}{N}; \theta, \Delta + \tfrac{\Sigma}{N}\right) \tag{2.37}$$

Using Equation (2.15) and a bit of algebra, the predictive distribution can then be

expressed as

$$p(\hat{x}|x) = \mathcal{N}(\hat{x}; \theta^*(x), \Delta^*(x) + \Sigma) \tag{2.38}$$

■ 2.3.5 Normal Inverse-Wishart Distribution on Mean and Covariance

When the mean and covariance of a multivariate Gaussian distribution are both unknown, the conjugate prior follows a Normal Inverse-Wishart (NIW) distribution of the following form

$$\mu, \Sigma \sim \text{NIW}(\mu, \Sigma; \kappa, \theta, \nu, \Delta) \triangleq \mathcal{N}\big(\mu; \theta, \tfrac{1}{\kappa}\Sigma\big)\, \mathcal{W}^{-1}(\Sigma; \nu, \Delta), \tag{2.39}$$

where $\mathcal{W}^{-1}(\Sigma; \nu, \Delta)$ denotes the following Inverse-Wishart distribution

$$\mathcal{W}^{-1}(\Sigma; \nu, \Delta) \triangleq \frac{|\nu\Delta|^{\frac{\nu}{2}}}{2^{\frac{\nu D}{2}}\Gamma_D(\frac{\nu}{2})} |\Sigma|^{-\frac{\nu+D+1}{2}} \exp\left[-\frac{1}{2}\operatorname{tr}\left(\nu\Delta\Sigma^{-1}\right)\right]. \tag{2.40}$$

Here, $\Gamma_D(\cdot)$ denotes the multivariate gamma function defined as

$$\Gamma_D(x) \triangleq \pi^{\frac{D(D-1)}{4}} \prod_{d=1}^{D} \Gamma\left(x + \tfrac{1-d}{2}\right). \tag{2.41}$$

The hyper-parameters κ and ν capture pseudo-counts of the prior on the mean and covariance, respectively. A large pseudo-count value corresponds to a prior that is more peaked. The hyper-parameter θ captures the prior mean of the mean parameter, and the prior mean of the covariance is related to Δ via

$$\mathbb{E}[\Sigma] = \frac{\nu\Delta}{\nu - D - 1}. \tag{2.42}$$

As ν increases, the expected value approaches Δ. Thus, it is convenient to think of Δ as the mean of the covariance, though this is not exactly correct. In some literature, the NIW distribution is alternatively parametrized with a scale matrix, $\Psi \triangleq \nu\Delta$, instead of Δ. We find the above parametrization to be more intuitive, since Δ can thought of as the mean of the covariance, instead of having to consider $\frac{1}{\nu}\Psi$, and use this parametrization for the remainder of the thesis.

Multivariate Gaussian Conjugacy

The algebra is quite complicated to derive the posterior hyper-parameters, marginal data likelihood, and predictive distributions with a Normal Inverse-Wishart prior. We therefore only show the resulting distributions. Additionally, we denote the posterior hyper-parameters without the explicit dependence on x for compactness.

The posterior hyper-parameters are as follows

$$\kappa^* = \kappa + N, \qquad \theta^* = \frac{1}{\kappa^*}\left[\kappa\theta + \sum_{i=1}^{N} x_i\right], \tag{2.43}$$

$$\nu^* = \nu + N, \qquad \Delta^* = \frac{1}{\nu^*}\left[\nu\Delta + \kappa\theta\theta^\top - \kappa^*\theta^*\theta^{*\top} + \sum_{i=1}^{N} x_i x_i^\top\right]. \tag{2.44}$$

The marginal data likelihood can be expressed as

$$p(x) = \frac{1}{\pi^{\frac{ND}{2}}} \frac{\Gamma_D\left(\nu^*/2\right)}{\Gamma_D\left(\nu/2\right)} \frac{|\nu\Delta|^{\nu/2}}{|\nu^*\Delta^*|^{\nu^*/2}} \left(\frac{\kappa}{\kappa^*}\right)^{D/2}. \tag{2.45}$$

The predictive distribution can be expressed as

$$p(\hat{x}|x) = \text{Student-t}_{\nu^*-D+1}\left(\hat{x}; \theta^*, \frac{\kappa^*+1}{\kappa^*(\nu^*-D+1)}\nu^*\Delta^*\right), \tag{2.46}$$

where the multivariate Student-t distribution is defined as

$$\text{Student-t}_\nu(x;\mu,\Sigma) = \frac{\Gamma\left(\frac{\nu+D}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)(\nu\pi)^{\frac{D}{2}}} |\Sigma|^{-\frac{1}{2}} \left[1 + \frac{1}{\nu}(x-\mu)^\top\Sigma^{-1}(x-\mu)\right]^{-\frac{\nu+D}{2}}. \tag{2.47}$$

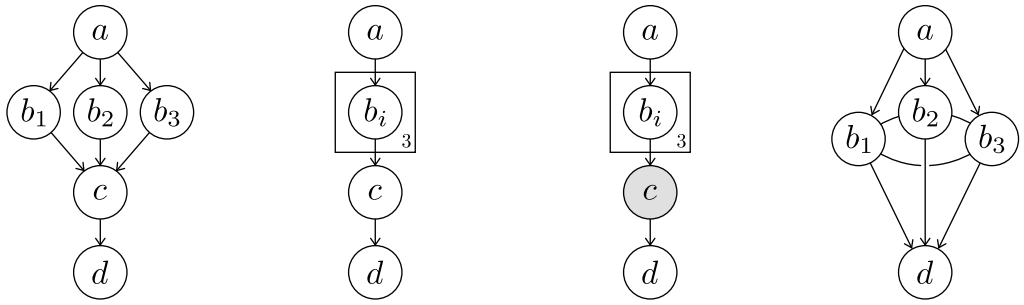
■ 2.4 Probabilistic Graphical Models

In simple problem formulations, such as the GPS tracking problem in Example 2.1.1, using conjugate priors makes posterior inference tractable. However, in more complex models, this is not always the case. In fact, it can be difficult to even represent the dependencies in complicated models. Probabilistic graphical models (cf. [66]) are a useful representation to visualize these complex dependencies. We now briefly review two types of graphical models: the directed graphical model, and the undirected graphical model. In these models, a random variable is represented with a circular graphical node, and dependencies between variables are represented with an edge.

■ 2.4.1 Directed Acyclic Graphical Models

Directed graphical models give an explicit expression for the joint distribution. A directed edge from a node z to a node x encodes information about the conditional distribution, $p(x|z)$. Directed graphs can often be thought of as being a *generative* model, where one generates random variables conditioned on all of its parents. As such, they are not well defined for cyclic graphs. In the remainder of this thesis, we will assume that directed graphical models are *acyclic*.

As an example, we consider the generative process for the graphical model depicted in Figure 2.2a. One can generate a sample from this model by completing the following



(a) Directed Model (b) Plate Notation (c) With Observations (d) Posterior Graph

Figure 2.2: Example directed graphical model. If the b_i's in (a) are independent and identically distributed conditioned on a, the compact plate notation of (b) can be used. Observed nodes are shaded, like the random variable c in (c). The resulting posterior distribution, conditioned on observed variables, connects the parents to each other and to the children.

steps:

1. Sample $a \sim p(a)$.
2. Sample $b_1 \sim p(b_1|a)$, $b_2 \sim p(b_2|a)$, and $b_3 \sim p(b_3|a)$.
3. Sample $c \sim p(c|b_1, b_2, b_3)$.
4. Sample $d \sim p(d|c)$.

The entire joint distribution of the model can then be expressed as the following

$$p(a, b_1, b_2, b_3, c, d) = p(a)\ p(b_1|a)p(b_2|a)p(b_3|a)\ p(c|b_1, b_2, b_3)\ p(d|c). \tag{2.48}$$

Additionally, if the set of random variables, $\{b_1, b_2, b_3\}$, are independent and identically distributed conditioned on a, one can use the compact *plate notation* of Figure 2.2b, where the number in the rectangular *plate* denotes the number of replicated variables.

Shaded nodes in a graphical model denote an observation. For example, Figure 2.2c represents a model where the random variable c is observed. We note that when a particular random variable is observed, the dependency structure for the graph changes. In particular, the parents of the observed become interdependent, and the children of the observed node depend on the parents. Figure 2.2d depicts the resulting dependence after observing c. This operation is commonly referred to as *moralizing* the graph, since parents are connected with an additional edge. We note that the interdependence among the parents cannot typically be modeled with a directed relationship and is therefore represented with a generic undirected edge. More specifically, the posterior

(a) Latent Markov Chain (b) Markov Chain with Observations

Figure 2.3: Latent and observed Markov chains.

distribution is proportional to the joint

$$p(a, b_1, b_2, b_3, d|c) = \frac{p(a, b_1, b_2, b_3, c, d)}{p(c)} \propto p(a, b_1, b_2, b_3, c, d), \tag{2.49}$$

and the posterior distribution of the b_i's conditioned on all other variables can be expressed as

$$p(b_1, b_2, b_3|a, c, d) = p(b_1, b_2, b_3|a, c) \propto p(b_1|a)p(b_2|a)p(b_3|a)\ p(c|b_1, b_2, b_3). \tag{2.50}$$

The newly introduced undirected edges exactly capture the $p(c|b_1, b_2, b_3)$ term, and is clearly missing the generative nature of directed edges.

Markov Chains

A Markov chain is a specific type of directed graphical model where each latent node has one parent and one child. An example of a Markov chain with no observations is shown in Figure 2.3a. In many applications, the Markov chain persists through time, and the subscript index refers to a particular time instance. While a purely latent Markov chain is rarely of interest, one can also include observations in the graphical model, as depicted in Figure 2.3b. Observations in Markov chains can occur at every time point or at arbitrary points as depicted in the figure.

We note that this model is applicable to many problems with temporal dynamics such as Example 2.1.1. In such a problem, the data likelihood distribution, $p(x^t|z^t)$, which is often referred to as the emission distribution in Markov chains, can be modeled with a 2D Gaussian distribution in Euclidean space. The prior distribution on the evolution of the latent location, $p(z^t|z^{t-1})$, which is often referred to as the dynamics of the Markov chain, can also be modeled with a Gaussian. We note that the actual solution used in Example 2.1.1 involves modeling smoothness in trajectory and requires a higher-order Markov chain where nodes have more than a single parent and child. Regardless, when both the emission and dynamics follow Gaussian or categorical distributions, the posterior distribution of any or all of the latent variables can be found in closed form using the Kalman filter [67] or the Forward-Backward Algorithm [99]. Other distributions require approximate inference methods, such as those discussed in Section 2.5.2.

2.4.2 Undirected Graphical Models

As eluded to in the previous section, some distributions cannot be easily decomposed into sequential conditional distributions that form a generative model. One alternative representation in such situations is to use an undirected graphical model. Undirected graphical models are also often referred to as Markov random fields (MRFs).

We first begin by defining a clique in an undirected graphical model, which is a set of nodes that is fully connected (i.e., every node in the clique is connected directly to every other node in the clique). Each individual node is called a *singleton* clique, and each pair of nodes that are joined by an edge form a *pairwise* clique. Higher-order cliques can also exist (e.g., the b_i's of Figure 2.2d).

In an undirected graphical model, the distribution over the random variables, z, can be factored into the product of functions of cliques as follows

$$p(z) = \frac{1}{Z} \prod_{c \in C(G)} \psi_c(z_c) \propto \prod_{c \in C(G)} \psi_c(z_c), \tag{2.51}$$

where G is the graph, $C(G)$ is the set of all cliques in the graph, and $\psi_c(\cdot)$ denotes a function for clique c. These clique functions are often referred to as clique *potentials*. Z in the above equation is often referred to as the *partition function*, and ensures that the resulting distribution integrates to 1. The partition function for arbitrary MRFs can be difficult to calculate. Fortunately, it is not needed for all inference algorithms. We note that when only singleton and pairwise cliques exist in the graphical model, the distribution can be expressed as

$$p(z) = \frac{1}{Z} \prod_i \psi_i(z_i) \prod_{\{i,j\} \in \mathcal{E}(G)} \psi_{ij}(z_i, z_j), \tag{2.52}$$

where ψ_i and ψ_{ij} denote the singleton and pairwise clique potentials, and $\mathcal{E}(g)$ denotes the set of edges in the graph.

2.5 Sampling Algorithms

When posterior distributions are not analytical, one option for inference is to sample from the model. Multiple samples can be combined in a Monte Carlo method to calculate event probabilities to any desired precision. In this section, we review some sampling algorithms that can be used when the target distribution of interest cannot be sampled directly.

2.5.1 Markov Chain Monte Carlo Sampling

One method for sampling from an arbitrary target distribution is to use a Markov chain Monte Carlo (MCMC) algorithm. MCMC methods simulate a Markov chain with a particular transition distribution such that the *stationary* distribution of the

chain is exactly the target distribution of interest. Certain conditions must be specified to ensure that this condition holds. A sample from the target distribution can then be generated by simulating a Markov chain until it converges to its stationary distribution, followed by taking the value of the chain when it is terminated. We review relevant Markov chain theory here.

We begin by defining certain properties of Markov chains. The state of a Markov chain at iteration t is denoted as $z^{(t)}$. Suppose that the Markov chain evolves according to some transition distribution, $q^*(z^{(t+1)}|z^{(t)})$. A *stationary distribution* of a Markov chain is a distribution over states that is invariant under the transition distribution, q^*. We define this concept more formally.

Definition 2.5.1 (Stationary Distribution)**.** If a distribution, $f_z(\cdot)$, over the state of the Markov chain satisfies the following relationship

$$f_z(z^{(t+1)}) = \int f_z(z^{(t)}) q^*(z^{(t+1)}|z^{(t)}) dz^{(t)}, \quad \forall z^{(t+1)} \tag{2.53}$$

then $f_z(\cdot)$ is defined to be a stationary distribution of the Markov chain.

Stationarity of a Markov chain with respect to a transition distribution essentially means that if the chain is currently in the stationary distribution, simulating a transition from q^* will not alter the distribution. Multiple such distribution can exist for a Markov chain. MCMC sampling algorithms must consequently ensure uniqueness by enforcing ergodicity of the Markov chain. An ergodic chain must satisfy a set of properties, which we discuss after some additional definitions.

Definition 2.5.2 (Accessible States)**.** A state j is said to be accessible from state i if, for some $t \geq 0$, the following condition holds for a given transition distribution:

$$\Pr[z^{(t)} = j \mid z^{(0)} = i] > 0. \tag{2.54}$$

Definition 2.5.3 (Communicating States)**.** A state is said to communicate with another state if they are accessible from each other.

Definition 2.5.4 (Communicating Class)**.** A communicating class is a set of states that satisfies the condition where every pair of states in the class communicate with each other.

Definition 2.5.5 (Irreducible Markov Chains)**.** A Markov chain is said to be irreducible if the chain contains a single communicating class.

Definition 2.5.6 (Period of State)**.** A state i is said to be periodic with period k if the chain can only return to state i after a multiple of k iterations.

Definition 2.5.7 (Aperiodicity)**.** A Markov chain is said to be aperiodic if every state of the chain has period 1.

We are now finally ready to discuss the definition of an ergodic chain.

Definition 2.5.8 (Ergodicity)**.** A finite-state Markov chain is said to be ergodic if it is aperiodic and irreducible. Ergodic Markov chains will converge to a unique stationary distribution regardless of the initial state.

Ergodicity is clearly a desirable trait in MCMC sampling because of the convergence guarantees. In many situations, proving that a Markov chain is aperiodic is difficult (cf. [42]) and one typically only shows that it is irreducible (i.e., that every state can be reached from every other state). In the unlikely case that the simulated Markov chain is periodic, average statistics of the sample path will still be correct.

Metropolis-Hastings Sampling

The idea of MCMC sampling is to simulate a Markov chain that has the target distribution as a stationary distribution. Ergodicity ensures convergence to the chain, but one has to use additional methodologies to ensure the correct stationary distribution. The Metropolis-Hastings algorithm is one such method.

Suppose that one can sample from some arbitrary distribution, $q(\hat{z}|z^{(t)})$. We refer to this distribution as the *proposal* distribution, and note that it is different from the transition distribution of the Markov chain, $q^*(z^{(t+1)}|z^{(t)})$. Metropolis et al. [90] developed an algorithm that constructs a transition distribution, q^*, from a symmetric proposal distribution, q, such that the stationary distribution is exactly the target distribution. Hastings [51] later generalized this algorithm to allow for non-symmetric proposal distributions. The latter algorithm is commonly referred to as the Metropolis-Hastings (MH) algorithm.

The concept underlying the Metropolis-Hastings algorithm is the notion of *detailed balance*. The detailed balance condition for Markov chains is defined as the following.

Theorem 2.5.1 (Detailed Balance)**.** *Let $f_z(z)$ denote the target distribution. If a Markov chain is constructed with a transition distribution, q^*, that satisfies*

$$f_z(z_1)q^*(z_2|z_1) = f_z(z_2)q^*(z_1|z_2), \tag{2.55}$$

then the chain is said to satisfy the detailed balance condition. Furthermore, $f_z(z)$ is guaranteed to be a stationary distribution of the chain.

Proof. Stationarity of a chain must satisfy the condition described in Definition 2.5.1. That is, we must show that if the Markov chain is currently in the stationary distribution, f_z, transitioning with respect to q^* does not change the resulting distribution.

This can be seen from the following

$$\begin{aligned} p(z^{(t+1)}) &= \int f_z(z^{(t)}) q^*(z^{(t+1)}|z^{(t)}) dz^{(t)} \\ &= \int f_z(z^{(t+1)}) q^*(z^{(t)}|z^{(t+1)}) dz^{(t)} \\ &= f_z(z^{(t+1)}) \end{aligned} \tag{2.56}$$

Consequently, satisfying detailed balance guarantees that f_z is a stationary distribution of the Markov chain. □

Detailed balance is a *sufficient* condition to ensure that $f_z(z)$ is a stationary distribution, but it is not necessary. In other words, a Markov chain can have $f_z(z)$ as a stationary distribution without satisfying Equation (2.55). In typical applications, however, ensuring stationarity with respect to $f_z(z)$ is difficult without satisfying detailed balance.

The MH algorithm is now detailed. Let $\hat{z} \sim q(\hat{z}|z^{(t)})$ denote a sample from the user-specified proposal distribution. Hastings showed that if the transition distribution, q^*, is constructed according to

$$q^*(z^{(t+1)}|z^{(t)}, \hat{z}) = \begin{cases} \min\left[1, \frac{f_z(\hat{z})}{f_z(z^{(t)})} \frac{q(z^{(t)}|\hat{z})}{q(\hat{z}|z^{(t)})}\right], & z^{(t+1)} = \hat{z} \\ 1 - \min\left[1, \frac{f_z(\hat{z})}{f_z(z^{(t)})} \frac{q(z^{(t)}|\hat{z})}{q(\hat{z}|z^{(t)})}\right], & z^{(t+1)} = z^{(t)} \end{cases}, \tag{2.57}$$

then the resulting Markov chain satisfies the detailed balance condition. The constructed transition distribution subjects the newly proposed sample to an accept or reject step. The ratio in the probability of acceptance, $H = \frac{f_z(\hat{z})}{f_z(z^{(t)})} \frac{q(z^{(t)}|\hat{z})}{q(\hat{z}|z^{(t)})}$, is typically referred to as the *Hastings ratio*. We note that it will sometimes be convenient to denote the Hastings ratio with the following

$$H = \frac{p(\hat{z})}{p(z)} \frac{q(z|\hat{z})}{q(\hat{z}|z)}, \tag{2.58}$$

where $z = z^{(t)}$ and $p(z) = f_z(z)$.

We now show that the MH construction of transition distributions from proposal distributions must satisfy detailed balance. Denoting the acceptance ratio as

$$\alpha(z, \hat{z}) \triangleq \min\left[1, \frac{f_z(\hat{z})}{f_z(z)} \frac{q(z|\hat{z})}{q(\hat{z}|z)}\right] \tag{2.59}$$

the transition distribution can be expressed as

$$q^*(z^{(t+1)} = \hat{z}|z^{(t)} = z) = q(\hat{z}|z)\alpha(z, \hat{z}) + (1 - \alpha(z, \hat{z}))\, \delta(\hat{z} - z). \tag{2.60}$$

Additionally, we note the following relationship:

$$f_z(z)q(\hat{z}|z)\alpha(z,\hat{z}) = f_z(\hat{z})q(z|\hat{z})\alpha(\hat{z},z) \tag{2.61}$$

It is then trivial to show that the resulting chain satisfies detailed balance:

$$\begin{aligned}
&f_z(z)q^*(z^{(t+1)} = \hat{z}|z^{(t)} = z) \\
&= f_z(z)q(\hat{z}|z)\left[\alpha(z,\hat{z}) + (1-\alpha(z,\hat{z}))\,\delta(\hat{z}-z)\right] \\
&= f_z(\hat{z})q(z|\hat{z})\left[\alpha(\hat{z},z) + \left(\frac{f_z(z)q(\hat{z}|z)}{f_z(\hat{z})q(z|\hat{z})} - \alpha(\hat{z},z)\right)\delta(\hat{z}-z)\right] \\
&= f_z(\hat{z})q(z|\hat{z})\left[\alpha(\hat{z},z) + (1-\alpha(\hat{z},z))\,\delta(\hat{z}-z)\right] \\
&= f_z(\hat{z})q^*(z^{(t)} = z|z^{(t+1)} = \hat{z}),
\end{aligned}$$

which is exactly the detailed balance condition. □

The Metropolis-Hastings algorithm therefore guarantees that the target distribution is a stationary distribution of the chain. Furthermore, Markov chain theory states that the resulting Markov chain is guaranteed to converge *uniquely* to the stationary distribution if it is ergodic.

We summarize the steps in Metropolis-Hastings in Algorithm 2.1. The MH algo-

Algorithm 2.1 The Metropolis-Hastings Algorithm

1. Initialize $z^{(0)}$ arbitrarily and set $t \leftarrow 0$
2. Sample a new proposed sample from $\hat{z} \sim q(\hat{z}|z^{(t)})$
3. Accept the proposal with probability $\Pr[z^{(t+1)} = \hat{z}] = \min\left[1, \frac{f_z(\hat{z})}{f_z(z^{(t)})}\frac{q(z^{(t)}|\hat{z})}{q(\hat{z}|z^{(t)})}\right]$.
4. Otherwise, reject the proposal and leave the state unchanged (i.e., $z^{(t+1)} = z^{(t)}$).
5. Increment t and repeat from Step 2.

rithm allows the sampling of an arbitrary distribution from some user-specified proposal distribution. Consequently, the choice of the proposal distribution will often greatly impact the "burn-in" time, which is the time it takes for the chain to reach its stationary distribution. In the limiting case where $q(\hat{z}|z) = f_z(\hat{z})$, the Hastings ratio trivially simplifies to 1 and every sample is accepted. Directly sampling from the target distribution is typically infeasible (which motivates the need for MCMC sampling), but this observation gives insight in designing good proposals. In particular, the closer the proposal distribution is to the target distribution, the better the convergence.

Gibbs Sampling

Gibbs sampling [40] is a special case of Metropolis-Hastings where the transition distribution only acts on a subset of the variables (or, a subset of dimensions of a multi-dimensional variable). In particular, the Gibbs sampler chooses the proposal distribu-

tion to be the true posterior distribution of the subset of variables conditioned on all variables. We outline the steps of Gibbs sampling in Algorithm 2.2, denoting $\backslash i$ as all the indices excluding i.

Algorithm 2.2 The Gibbs Sampling Algorithm

1. Initialize $z^{(0)}$ arbitrarily and set $t \leftarrow 0$
2. Sample an index, i, that selects a dimension of the random variable: $i \sim q(i)$
3. Sample a new z: $z_i^{(t+1)} \sim p\big(z_i^{(t+1)} | z_{\backslash i}^{(t)}\big)$.
4. Copy the other values of z: $z_{\backslash i}^{(t+1)} = z_{\backslash i}^{(t)}$.
5. Increment t and repeat from Step 2.

Under some mild conditions of the target distribution, it is simple to see that the resulting Markov chain produced by Gibbs sampling is irreducible. Any random index can be selected in Step 2, and it can change to any value in Step 3. Consequently, any state is reachable from any other state assuming that the conditional distributions do not result in disconnected states [80].

It is also quite simple to see that the resulting chain satisfies detailed balance, which ensures that the target distribution is a stationary distribution of the Markov chain. Any Metropolis-Hastings algorithm satisfies detailed balance if the Hastings ratio is implemented correctly. Gibbs sampling precludes the need of an accept/reject step because it results in a Hastings ratio that evaluates to 1. We therefore only need to verify this claim. Letting $q(\hat{z}|z^{(t-1)}) = q(i)p\big(\hat{z}_i|z_{\backslash i}^{(t-1)}\big)$, the Hastings ratio can be expressed as

$$H = \frac{p(\hat{z})}{p(z^{(t)})}\frac{q(z^{(t)}|\hat{z})}{q(\hat{z}|z^{(t)})} = \frac{p\big(z_{\backslash i}^{(t)}\big)p\big(\hat{z}_i|z_{\backslash i}^{(t)}\big)}{p\big(z_{\backslash i}^{(t)}\big)p\big(z_i^{(t)}|z_{\backslash i}^{(t)}\big)}\frac{q(i)p\big(z_i^{(t)}|z_{\backslash i}^{(t)}\big)}{q(i)p\big(\hat{z}_i|z_{\backslash i}^{(t)}\big)} = 1. \tag{2.62}$$

This shows that the Gibbs sampling algorithm can accept all proposed samples while still satisfying detailed balance.

Gibbs sampling is often preferred over Metropolis-Hastings because it does not require one to specify a proposal distribution. However, it can only be used when the conditional posterior distributions are known. Furthermore, Gibbs sampling may result in slow convergence because only a single random dimension of the latent variable is sampled at a time. This procedure can explore the space very slowly due to the local changes proposed by the sampling algorithm. Alternatives such as blocked Gibbs sampling attempt to address this issue, but come at a significant computational cost.

Reversibile-Jump MCMC

Green [44] developed the Reversible-Jump MCMC (RJMCMC) algorithm, which is a generalization of the Metropolis-Hastings algorithm where auxiliary variables are

used to propose moves in mismatched dimensions. Let $z \in \mathbb{R}^D$ denote the current D-dimensional variable and let $\hat{z} \in \mathbb{R}^{D+d}$ denote the desired proposal. RJMCMC augments the current space to be $[z, v]$ and the proposed space to be $[\hat{z}, \hat{u}]$. The auxiliary variables v and $\hat{u}$ must be chosen so that both spaces have the same dimensionality (i.e., $v \in \mathbb{R}^{C+d}$ and $\hat{u} \in \mathbb{R}^C$, where C is any integer that satisfies $C + d \geq 0$ and $C \geq 0$). RJMCMC then deterministically changes from the current space to the proposed space using a user-specified function f that performs $[\hat{z}, \hat{u}] = f(z, v)$.

Let J_f denote the Jacobian matrix of function $f(z, v)$, of the form

$$J_f = \begin{bmatrix} \frac{\partial \hat{z}_1}{\partial z_1} & \cdots & \frac{\partial \hat{z}_1}{\partial z_D} & \frac{\partial \hat{z}_1}{\partial v_1} & \cdots & \frac{\partial \hat{z}_1}{\partial v_{C+d}} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \hat{z}_{D+d}}{\partial z_1} & \cdots & \frac{\partial \hat{z}_{D+d}}{\partial z_D} & \frac{\partial \hat{z}_{D+d}}{\partial v_1} & \cdots & \frac{\partial \hat{z}_{D+d}}{\partial v_{C+d}} \\ \frac{\partial \hat{u}_1}{\partial z_1} & \cdots & \frac{\partial \hat{u}_1}{\partial z_D} & \frac{\partial \hat{u}_1}{\partial v_1} & \cdots & \frac{\partial \hat{u}_1}{\partial v_{C+d}} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \hat{u}_C}{\partial z_1} & \cdots & \frac{\partial \hat{u}_C}{\partial z_D} & \frac{\partial \hat{u}_C}{\partial v_1} & \cdots & \frac{\partial \hat{u}_C}{\partial v_{C+d}} \end{bmatrix}. \tag{2.63}$$

Green showed that the steps outlined in Algorithm 2.3 preserve detailed balance and ensure that the target distribution is a stationary distribution of the resulting Markov chain.

Algorithm 2.3 The RJMCMC Proposal

1. Generate auxiliary variables for the current state from some proposal distribution:
$$v \sim q(v|z). \tag{2.64}$$

2. Apply a deterministic function, f to the current state to obtain a new proposal:
$$[\hat{z}, \hat{u}] = f(z, v). \tag{2.65}$$

3. Accept the proposal with the following probability
$$\min \left[1, \frac{p(\hat{z})q(\hat{u}|\hat{z})}{p(z)q(v|z)} \left| \det(J_f) \right| \right]. \tag{2.66}$$

We note that the Metropolis-Hastings algorithm is a special instance of the RJMCMC algorithm. More precisely, assuming q denotes the MH proposal distribution, the MH algorithm can be constructed as follows: (1) sample auxiliary variables from the proposal distribution, $v \sim q(v|z)$; and (2) use the deterministic function that maps $\hat{z} = v$ and $\hat{u} = z$. The Jacobian of this deterministic function is simply 1, resulting in accepting proposals with probability

$$\min \left[1, \frac{p(\hat{z})q(\hat{u}|\hat{z})}{p(z)q(v|z)} \left| \det(J_f) \right| \right] = \min \left[1, \frac{p(\hat{z})q(z|\hat{z})}{p(z)q(\hat{z}|z)} \right].$$

This is exactly the acceptance probability of MH in Equation (2.58).

Because of the similarities between RJMCMC and MH, we will often incorrectly refer to the ratio, $\frac{p(\hat{z})q(\hat{u}|\hat{z})}{p(z)q(v|z)}\left|\det(J_f)\right|$ as the Hastings ratio for simplicity. It should be implied from the context of the proposal distribution whether using the RJMCMC proposal is necessary.

Determining Convergence

There are typically two quantities of interest when evaluating the effectiveness of the Markov chain used in an MCMC sampling algorithm: the *burn-in* time and the *mixing* time. The burn-in time refers to the time it takes for the chain to reach the stationary distribution. Once the chain has burned-in, the state of the Markov chain contains a single sample from the target distribution. The state of the chain after one more transition will be highly correlated with the previous sample. Consequently, the chain is simulated for multiple iterations before taking another sample. The mixing time refers to the time required between samples such that the samples are independent.

When designing MCMC sampling algorithms, it is generally preferred to minimize both the burn-in time and the mixing time. If multiple random initializations are used to explore different modes, the burn-in time may have more of an impact. If only a single chain is used to draw samples, then mixing time may be a more important factor. It is generally very difficult to say anything concretely about either of these quantities except in very specific situations (e.g., [65]). As a result, burn-in is often showed empirically by monitoring joint model likelihoods, and mixing time is often measured via an autocorrelation on a user-specified scalar statistic of the model.

■ 2.5.2 Importance Sampling

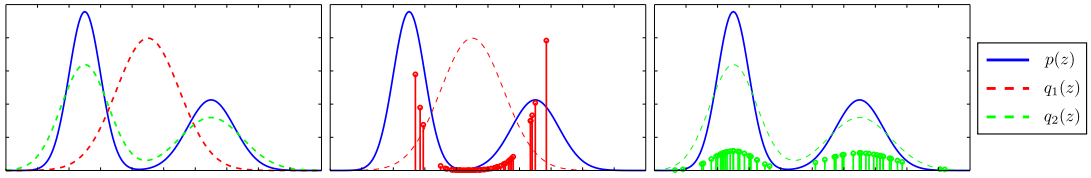
An alternative to Markov chain Monte Carlo sampling approaches are Monte Carlo algorithms that do not simulate a Markov chain. One such approach, called importance sampling, approximates the target distribution with a set of *weighted* samples. Importance sampling first draws N independent samples from a proposal distribution, denoted $q(z)$:

$$z_s \sim q(z), \quad \forall s \in \{1, \ldots, N\}. \tag{2.67}$$

Each sample is then assigned a corresponding *importance weight* according to

$$w_s \triangleq w(z_s) = \frac{p(z_s)}{q(z_s)}. \tag{2.68}$$

Importance sampling allows one to approximate expectations of an arbitrary function $h(\cdot)$ over the true target distribution via a weighted sum of the proposed samples.



(a) Target / Proposals (b) Samples from $q_1(z)$ (c) Samples from $q_2(z)$

Figure 2.4: An importance sampling example. (a) shows the target distribution (blue) and two proposal distributions (red and green). (b) and (c) show 100 samples and associated weights for the two proposal distributions. Better approximations occur when the proposal distribution is more similar to the target distribution (green).

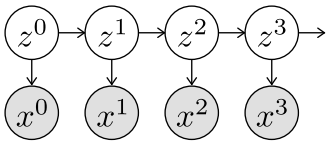


Figure 2.5: Markov chain that can be inferred via particle filtering.

This can be seen from the following

$$\frac{1}{N}\sum_{s=1}^{N} w_s h(z_s) \approx \mathbb{E}_q[w(z)h(z)] = \int q(z)w(z)h(z)dz = \int p(z)h(z)dz = \mathbb{E}_p[h(z)] \tag{2.69}$$

In general, the closer the proposal distribution is to the target distribution, the better the approximation. For example, consider the distributions shown in Figure 2.4. The green proposal distribution is more similar to the target distribution than the red proposal distribution. Consequently, the associated importance weights are more evenly distributed in the second case than the first, indicating a better approximation.

Particle Filtering

Importance sampling can also be extended to incorporate temporal dependence in a Markov chain. Consider a general Markov chain where z denotes the set of hidden variables, and x denotes the set of observed variables (as depicted in Figure 2.5). Similar to importance sampling, a particle filter [59] then represents the poster distribution, $p(y^t|x^{0:t})$, at time t with a set of weighted samples, $\{z_s^t, w_s^t\}$. We use the superscript $(0:t)$ to mean the set of indices in $\{0, \ldots, t\}$.

Particle filtering then propagates particles through time and updates the corresponding importance weights such that they still approximate the true target distribution correctly. In typical algorithms, a particle, z_s^t is propagated to the new time point using the distribution over temporal dynamics, $p(z_s^{t+1}|z_s^t)$. In this case, the resulting

weight updates must reflect the new data likelihood term

$$z_s^{t+1} \sim p(z^{t+1}|z^t)\,, \quad w_s^{t+1} = p(x^{t+1}|z_s^{t+1})\, w_s^t. \tag{2.70}$$

Particle weights may decay over time for many likelihood-dominated applications, indicating a poor representation of the desired distribution. Sequential importance resampling (SIR) techniques (cf. [43]) are typically utilized to mitigate this issue. SIR replaces particles that have small weights with particles that have large ones when the representation is poor.

■ 2.6 Implicit Shapes Representations via Level-Set Methods

Level-set methods provide a way to implicitly represent and evolve an N-dimensional (or less) hyper-surface in an N-dimensional space. The works of Osher and Fedkiw [95] and Sethian [105] provide the original development of level-set methods and a wealth of knowledge on this subject. When applied to image segmentation, a scalar function, φ, is defined by values on a two-dimensional Cartesian grid. In practice, this function is stored as an image, and the height of the level-set function is defined for each pixel in the image.

The implicit hyper-surface as it pertains to image segmentation is just a curve that exists in the two-dimensional support of the image. Any level-set (the intersection of the surface with a constant height plane) of φ can be used as the implicit hyper-surface, but the zero level-set is typically chosen for the representation. The implied curve, C, is defined as the set of all real-valued points on the 3-dimensional level-set function that have height zero. The zero level-set divides the image into two regions that consist of the positive and negative values of the level-set function, respectively.

The implicit curve of the level-set function is defined at a sub-pixel accuracy since the location that φ is zero may be between two pixels. In the realm of image segmentation problems, such fine-grained accuracy is often not needed. As such, we work in the discrete domain of pixels. The two regions created by the zero level-set are then distinguished by assigning a binary label to each pixel, i, denoted z_i, according to

$$z_i = \mathbb{1}[\varphi_i \geq 0]. \tag{2.71}$$

We can then define the discrete curve as the set of pixels that are on the boundary of the regions:

$$C = \left\{ i; z_i = 1, \prod_{j \in \mathbf{N}(i)} z_j = 0 \right\}, \tag{2.72}$$

where $\mathbf{N}(i)$ denotes the neighbors of i. An example level-set function is shown in Figure 2.6

Level-set methods involve representing a hyper-surface in a higher dimension, typically leading to increased memory and computation. However, the utility of representing

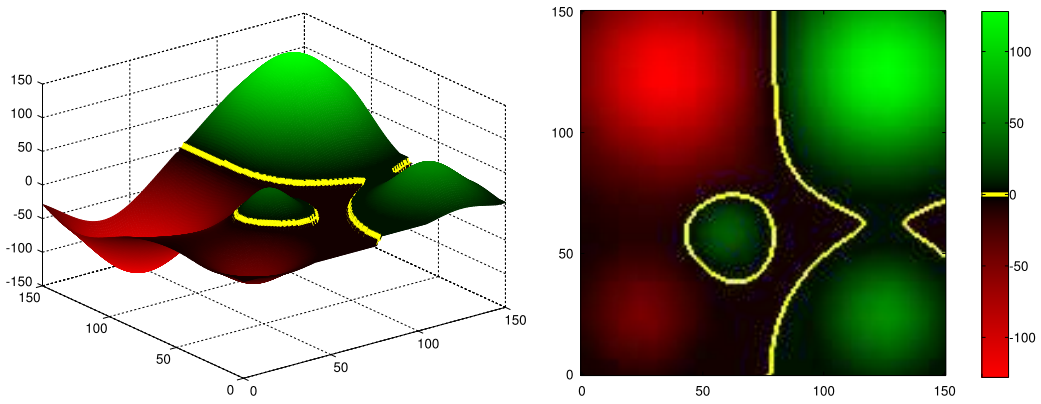


Figure 2.6: Example of a level-set function. The zero level-set is represented with the yellow curve.

a curve with level-set methods is that the curve is implicitly represented. Creating or removing a new region is a matter of perturbing the underlying surface. If an explicit representation (e.g., snakes [68]) is used, it requires the user to maintain the explicit set of points on each curve. Creating or removing regions with an explicit representation requires bookkeeping and suffers from what is known as reparametrization of the curve (i.e., resampling points on the curve as it changes shape). In fact, the overhead of representing the entire underlying surface with an implicit representation typically outweighs the nuisance of an explicit representation.

■ 2.6.1 Signed Distance Function

In image segmentation algorithms using level-set methods, the user is only concerned with the zero level-set because it is the curve that segments the image. Consequently, this restricts pixels on the curve to have zero height, but pixels away from the curve need only have the same sign. An infinite number of parameterizations of level-set functions exist that have the same zero level-set.

A very common approach is to make the level-set function a signed distance function, which has the property that the absolute value at each pixel is the minimum distance to the zero level-set. An illustration of the signed distance function for the same zero level-set as Figure 2.6 is shown in Figure 2.7.

A signed distance function is a solution to the general *Eikonal equation* for $F = 1$:

$$|\nabla\varphi| = F, \tag{2.73}$$

subject to the constraint that the sign of φ is preserved. Consequently, in addition to other numerical stability benefits, the signed distance function also satisfies the condition that the gradient has magnitude 1. This relationship holds for all pixels except those that are equidistant from multiple points on the zero level-set (i.e., the

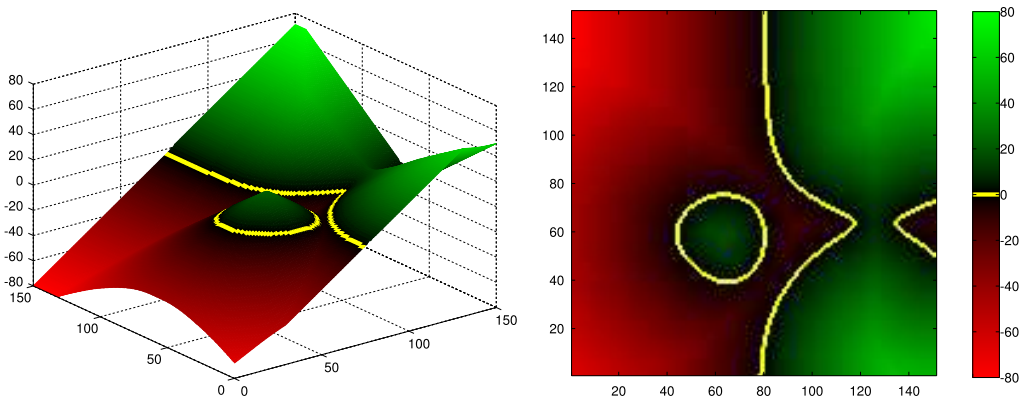


Figure 2.7: Example of a level-set function as a signed distance function.

peaks and valleys of the level-set function). Fast methods (e.g., [119]) exist for solving the Eikonal equation. A more in-depth description of the formulation and benefits of using a signed distance function can be found in [95].

■ 2.6.2 Sampling-Based Inference

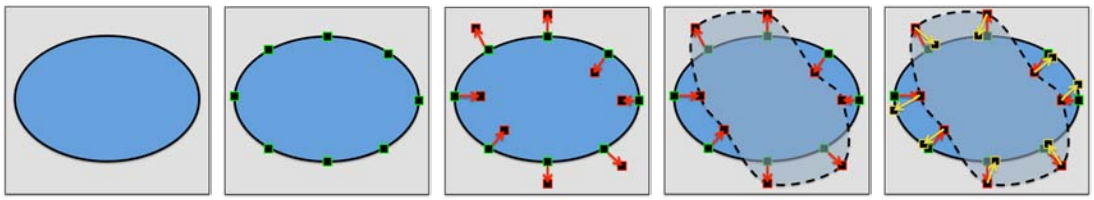
In many applications, one defines an energy functional over the implicitly-defined curve and attempts to optimize the energy to find the optimal configuration. However, in the Bayesian setting, one may be more interested with characterizing the associated Boltzmann distribution obtained by exponentiating the negative of the energy. In such cases, one approach for inference is to use Monte Carlo Markov Chain sampling methods. We now briefly discuss two such sampling methods, both which use a Metropolis-Hastings framework.

The first method was developed by Fan et al. [31], and is outlined in Algorithm 2.4. For additional details, please refer to its original presentation. We note that this

Algorithm 2.4 Alternating Implicit/Explicit Shape Sampling

1. Change from an implicit representation to explicit, arc-length parametrized markers.
2. Sample a perturbation magnitude for each marker point.
3. Construct a perturbation that changes each marker point in the normal direction by the sampled magnitude and that extends the velocity to other points on the curve.
4. Accept the proposal according to the Hastings ratio.
5. Repeat from Step 1 until convergence.

algorithm alternates between an implicit and explicit representation. An example of the entire proposal is depicted in Figure 2.8. We remind the reader that the Hastings ratio requires computing the probability of the forward proposal and the probability of the backward proposal. Because the normals of the curve change after the perturbation in



(a) Initialization (b) Markers (c) Perturbation (d) Proposal (e) Reverse Move

Figure 2.8: An example proposal from the alternating implicit/explicit shape sampling algorithm. The proposed move is indicated by the dotted shape. The forward move is indicated by the red arrows, and the reverse move is indicated by the yellow arrows.

Algorithm 2.4, the resulting backward proposal is difficult to compute. A visualization of this complication is illustrated in Figure 2.8e. Notice that the proposed shape at the red markers do not have the same normals as the original shape at the green markers. Consequently, the reverse move perturbs along a different normal direction and must perturb the curve with a different magnitude than the forward move. Finding the locations of the yellow markers complicates the calculation of the Hastings ratio. See [31] for a cumbersome exact calculation or for an approximation solution.

The second method was developed by Chen and Radke [23], and is outlined in Algorithm 2.5. For additional details, please refer to its original presentation. An

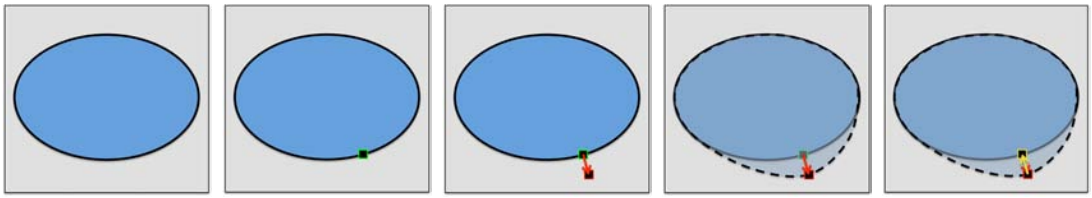
Algorithm 2.5 Foot-Point-Based Shape Sampling

1. Select a point on the curve randomly and call it the foot point.
2. Sample a perturbation magnitude the foot point.
3. Construct a perturbation that changes the foot point in the normal direction and smoothly changes the surrounding curve.
4. Accept the proposal according to the Hastings ratio.
5. Repeat from Step 1 until convergence.

example of this proposal is depicted in Figure 2.9 In this algorithm, only one marker point is used, denoted the *foot point*. The velocity is extended in the surrounding curve such that the proposed perturbation is smooth and preserves the normal direction at the foot point. As such, the reverse move can be calculated efficiently since it is just the probability of generating the reverse magnitude perturbation.

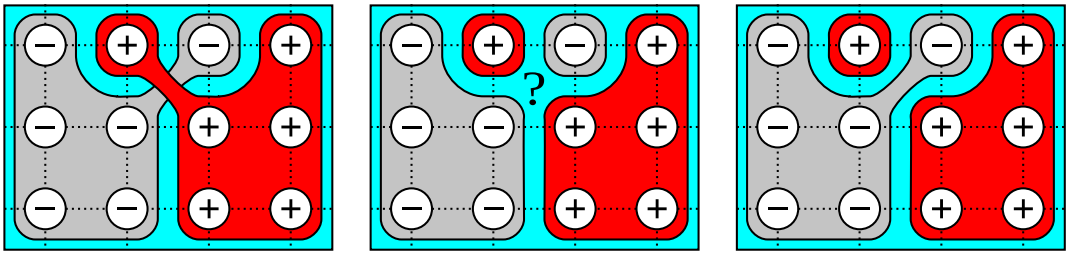
■ 2.7 Digital Topology

The topology of a continuous, compact surface is often described by its genus (i.e., the number of "handles"). Digital topology [72] is the discrete counterpart of continuous topology, where regions are represented via binary variables on a lattice grid.



(a) Initialization (b) Foot Point (c) Perturbation (d) Proposal (e) Reverse Move

Figure 2.9: An example proposal from the foot-point-based shape sampling algorithm. The proposed move is indicated by the dotted shape. The forward move is indicated by the red arrow, and the reverse move is indicated by the yellow arrow.



(a) $(n, \overline{n}) = (8, 8)$ (b) $(n, \overline{n}) = (4, 4)$ (c) $(n, \overline{n}) = (4, 8)$

Figure 2.10: Examples of topology paradoxes. $(n, \overline{n})$ indicate the connectivities for the foreground and background, respectively. (a) and (b) illustrate the paradoxes when the connectivities for the foreground and background are the same. In (a), part of the continuous space belongs to both the foreground and background. In (b), part of the continuous space is ambiguously owned by the two regions. When the connectivities are jointly chosen, as in (c), there is no such paradox.

■ 2.7.1 Connectiveness

In digital topology, *connectiveness*, which describes how pixels in a local neighborhood are connected, must be defined in pairs for the foreground (FG) and background (BG). For example, in 2D, a 4-connected region corresponds to a pixel being connected to its neighbors above, below, left, and right. An 8-connected region corresponds to being connected to the eight pixels in a 3×3 neighborhood. Connectivities must be jointly defined for the foreground (n) and background ($\overline{n}$) to avoid topological paradoxes. As shown in [72], valid connectivities for 2D are $(n, \overline{n}) \in \{(4, 8), (8, 4)\}$.

■ 2.7.2 Topological Numbers and Simple Points

Given a pair of connectivities, the topological numbers [6] at a particular pixel, T_n (for the FG) and $T_{\overline{n}}$ (for the BG) count the number of connected components a pixel is connected to in a 3x3 neighborhood. The connected components are computed ignoring the center pixel of interest. Figure 2.11 shows a few neighborhoods with their

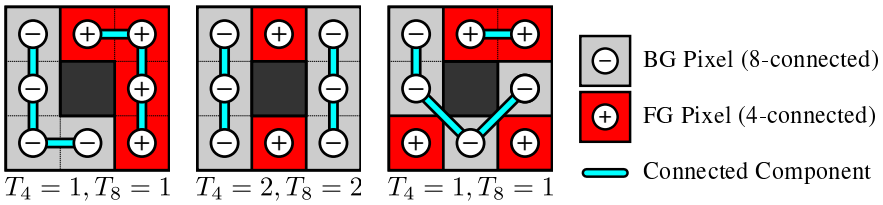


Figure 2.11: Examples of topological numbers with $(n, \overline{n}) = (4, 8)$.

corresponding topological numbers.

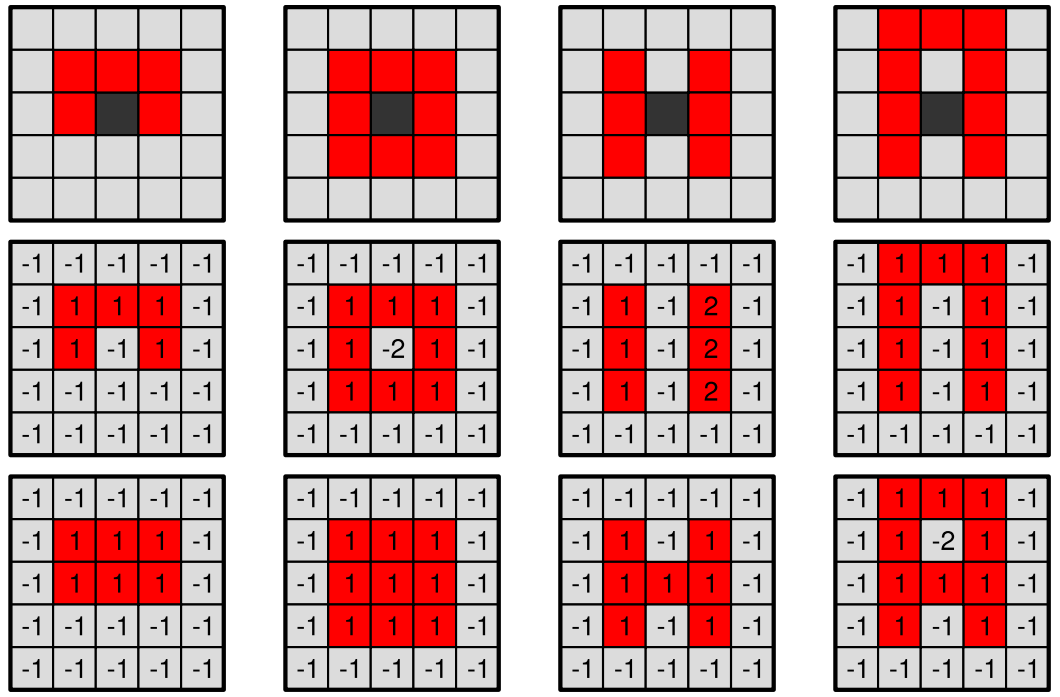
Topological numbers can be used to identify when changing the label of a pixel causes a topology change. A *simple point* is defined to be a point that can change labels without changing the digital topology of the background or foreground regions. Simple points can be uniquely identified by the condition that $T_n = T_{\overline{n}} = 1$. That is, when either topological number is not 1, switching labels of the corresponding pixel will necessarily change the topology of one of the regions. The work of [50] exploited this fact by only allowing simple points to change label during optimization, thereby not allowing the topology of the regions to change.

■ 2.7.3 Extended Topological Numbers

While these topological numbers can be used to detect a topological change, they cannot distinguish the splitting or merging of a region from the creation or destruction of a handle. Segonne [104] defines two additional, extended topological numbers, T_n^+ and $T_{\overline{n}}^+$, which count the number of *unique* connected components a pixel is connected to over the entire image domain. T_n^+ and $T_{\overline{n}}^+$ depend on how pixels are connected outside of the 3x3 region and allow one to distinguish all topological changes. The connected components for the extended topological numbers are also computed ignoring the center pixel of interest. Four examples of the extended topological numbers are shown in the top row of Figure 2.12.

By labeling each connected component in the foreground and background, Segonne shows that T_n^+ can be computed efficiently when the pixel currently belongs to the background. Consider the examples shown in Figure 2.12, and notice that T_n^+ can be computed when the current pixel is in the background (second row) by simply counting the unique foreground labels. Similarly, $T_{\overline{n}}^+$ can also be computed efficiently when the pixel currently belongs to the foreground by counting the unique background labels.

Unfortunately, T_n^+ cannot be computed efficiently when a pixel is moved from the foreground to the background, and $T_{\overline{n}}^+$ cannot be computed efficiently when a pixel is moved from the background to the foreground. For example, the number of unique labels in both Figures 2.12c and 2.12d is 1 when the pixel currently is in the foreground (bottom row), even though their corresponding values of T_n^+ are different. In 3D, one *must* calculate all extended topology numbers to uniquely identify all topology changes. This is clearly prohibitive since it requires connected component analysis for every single



(a) $T_n^+ = 1, T_{\overline{n}}^+ = 1$ (b) $T_n^+ = 1, T_{\overline{n}}^+ = 0$ (c) $T_n^+ = 2, T_{\overline{n}}^+ = 1$ (d) $T_n^+ = 1, T_{\overline{n}}^+ = 2$

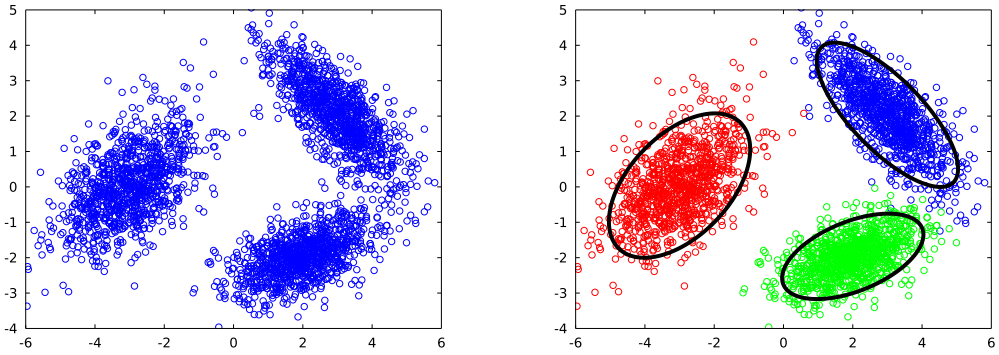
Figure 2.12: Extended topological numbers. The top row shows the configuration, where red indicates foreground, light gray indicates background, and dark gray indicates the pixel of interest. The second row shows the connected component labeling assuming the pixel belongs to the background. The third row shows the connected component labeling assuming the pixel belongs to the foreground.

pixel change.

■ 2.8 Finite Mixture Models

In this section, we describe the finite mixture model, which is a probabilistic model that is used in many lines of research. A single parametric distribution (e.g., Gaussian) is often not rich enough to capture interesting data. For example, approximating all intensities of pixels in an image with a single Gaussian distribution is nonsensical since we know that objects are composed of multiple distinct colors. A mixture model improves the expressiveness of these simple, parametric models by using a convex combination of multiple, parametric distributions. For the remainder of this section, we will describe finite mixture models that are assumed to be composed of K separate distributions. A generalization to an unknown number of components is described in Section 2.9.2.

A mixture model assumes that the observations are drawn from the following dis-



(a) Original Data (b) Latent Variables

Figure 2.13: An example of a finite mixture model. The color of the data points in the latent variables correspond to different values of z_i and the ellipses indicate the Gaussian parameters for each cluster (mean and covariance).

tribution

$$x_i \sim \sum_{k=1}^{K} \pi_k f_x(x_i; \theta_k), \tag{2.74}$$

where $\sum_{k=1}^{K} \pi_k = 1$, and $f_x(\cdot; \theta_k)$ denotes some density parametrized by θ_k (e.g., Gaussian). An equivalent formulation introduces a corresponding z_i for each data point that indicates which of the K possible distributions the data was drawn from. This formulation can be expressed as

$$z_i \sim \text{Cat}(\pi_1, \ldots, \pi_K), \tag{2.75}$$

$$x_i \sim f_x(x_i; \theta_{z_i}). \tag{2.76}$$

It is easily verified that the resulting x_i has the same distribution as Equation (2.74) when marginalizing over z_i. The z_i's are typically referred to as labels or assignments since their label assigns a data point to a particular mixture component. Furthermore, when the mixture model is framed in this formulation, a natural *clustering* of the data arises from the posterior inference on the latent z_i's. An example of data and desired latent variables in a finite mixture model is shown in Figure 2.13.

■ 2.8.1 Priors on Parameters

In general, the only variables of the mixture model that are observed are the data, x. Bayesian mixture models consequently place priors on both π and θ. Some methods (e.g., K-means) place uniform priors on these parameters, but it is also common to place a Dirichlet prior on π and a prior on θ that is conjugate to $f_x(x_i; \theta_k)$ (see Section 2.2). The corresponding probabilistic graphical model is shown in Figure 2.14. The full

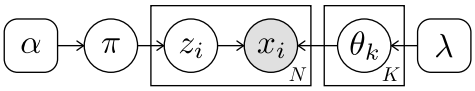


Figure 2.14: Graphical model for a Bayesian finite mixture model.

generative model is summarized by the following steps

$$(\pi_1, \ldots, \pi_K) \sim p(\pi_1, \ldots, \pi_K) = \mathrm{Dir}(\pi_1, \ldots, \pi_K; \alpha), \tag{2.77}$$
$$\theta_k \sim p(\theta_k) = f_\theta(\theta_k; \lambda), \qquad \forall k \in \{1, \ldots, K\}, \tag{2.78}$$
$$z_i \sim p(z_i|\pi) = \mathrm{Cat}(\pi_1, \ldots, \pi_K), \qquad \forall i \in \{1, \ldots, N\}, \tag{2.79}$$
$$x_i \sim p(x_i|z_i, \theta) = f_x(x_i; \theta_k), \qquad \forall i \in \{1, \ldots, N\}. \tag{2.80}$$

■ 2.8.2 Posterior MCMC Inference

Because the priors are all chosen to be conjugate, posterior inference via Gibbs sampling is quite straightforward. Exploiting the results of Section 2.2, posterior inference then iterates between the following steps

$$\pi \sim p(\pi|z) = \mathrm{Dir}(\pi_1, \ldots, \pi_K; \alpha + N_1, \ldots, \alpha + N_K), \tag{2.81}$$
$$\theta_k \sim p(\theta_k|x, z) = f_\theta(\theta_k; \lambda^*(x_{\mathcal{I}_k})), \qquad \forall k \in \{1, \ldots, K\}, \tag{2.82}$$
$$z_i \sim p(z_i|x_i, \pi, \theta) \propto \sum_{k=1}^{K} \pi_k f_x(x_i; \theta_k)) \mathbb{1}[z_i = k], \qquad \forall i \in \{1, \ldots, N\}, \tag{2.83}$$

where we denote $\mathcal{I}_k \triangleq \{i; z_i = k\}$, and $x_{\mathcal{I}_k}$ as the subset of all the data with label $z_i = k$. Alternatively, one can marginalize over π and potentially θ if conjugate priors are used. This method is not discussed here.

■ 2.9 Non-parametric Bayesian Statistics

Probabilistic modeling essentially attempts to fit real-world data with some approximate, tractable stochastic model. There is a long of history of using parametric distributions, some of which lend themselves to very efficient inference (e.g., conjugate priors discussed in Section 2.2). However, as the number of observations grows, one might expect the approximation of the real-world data with a finite parametric distribution to suffer. In these situations, it may be fitting to use a *non-parametric model*, where the number of model parameters grows with the number of observations.

There has been a lot of recent success in applying non-parametric Bayesian models to both synthetic and real-world data. In this section, we briefly review three such models that will be used in this thesis: the Gaussian process, the Dirichlet process, and the hierarchical Dirichlet process.

■ 2.9.1 Gaussian Processes

We begin with a brief discussion on Gaussian processes. For a more in-depth overview, please consult [100].

A Gaussian process (GP) can be thought of as the limit of a multivariate Gaussian distribution as the number of dimensions approaches infinity. As such, it is common to specify a mean *function* and a covariance *kernel* instead of a finite-length mean vector and covariance matrix. We will denote a sample from a GP as

$$g(i) \sim \mathrm{GP}\left(g(i); m(i), \kappa(i,j)\right), \tag{2.84}$$

where i and j denote the locations realized by the GP, $m(i)$ is the mean function, and $\kappa(i,j)$ is the covariance kernel. One main difference between a GP and a finite, multivariate Gaussian random variable is that GPs handle new data effortlessly; the covariance between any two data points (whether they were previously defined or newly observed) is completely characterized by $m(i)$ and $\kappa(i,j)$. The same cannot generally be said with a multivariate Gaussian random variable.

Exact Posterior Inference

Posterior distributions in Gaussian processes can be expressed in closed form. Let x denote an independent set of noisy observations of g, distributed according to

$$x_i \sim p(x_i|g_i) = \mathcal{N}\big(x_i; g_i, \sigma_x^2\big), \quad \forall i \in \{1, \ldots, N\}, \tag{2.85}$$

where $g_i \triangleq g(i)$. Assume that these observations are at a set of locations, $\mathcal{I}$ (i.e., $i \in \mathcal{I}$), and that we are interested in the posterior distribution of g at a set of possibly different locations, $\mathcal{I}^*$. We denote the covariance matrix $\Sigma^{\mathcal{I}\mathcal{I}}$ as the covariance between all observations, and $\Sigma^{\mathcal{I}^*\mathcal{I}^*}$ as the covariance between all desired locations. Similarly, the non-square matrices, $\Sigma^{\mathcal{I}\mathcal{I}^*}$ and $\Sigma^{\mathcal{I}^*\mathcal{I}}$ denote the covariance kernel evaluated at observed and desired locations. These matrices can be expressed element-wise as

$$\Sigma_{ij}^{\mathcal{I}\mathcal{I}} = \kappa(i,j), \quad \forall i \in \mathcal{I}, \quad j \in \mathcal{I}, \tag{2.86}$$
$$\Sigma_{ij}^{\mathcal{I}^*\mathcal{I}^*} = \kappa(i,j), \quad \forall i \in \mathcal{I}^*, \quad j \in \mathcal{I}^*, \tag{2.87}$$
$$\Sigma_{ij}^{\mathcal{I}\mathcal{I}^*} = \kappa(i,j), \quad \forall i \in \mathcal{I}, \quad j \in \mathcal{I}^*, \tag{2.88}$$
$$\Sigma_{ij}^{\mathcal{I}^*\mathcal{I}} = \kappa(i,j), \quad \forall i \in \mathcal{I}^*, \quad j \in \mathcal{I}. \tag{2.89}$$

As shown in [100], the posterior distribution of g evaluated at $\mathcal{I}^*$ is the following multivariate Gaussian distribution

$$g \sim \mathcal{N}\Big(g\,;\, \left[\Sigma^{\mathcal{I}\mathcal{I}^*}\left(\Sigma^{\mathcal{I}\mathcal{I}} + \sigma_x^2 \mathrm{I}_N\right)^{-1} x\right]\,,\, \left[\Sigma^{\mathcal{I}^*\mathcal{I}^*} - \Sigma^{\mathcal{I}^*\mathcal{I}}\left(\Sigma^{\mathcal{I}\mathcal{I}} + \sigma_x^2 \mathrm{I}_N\right)^{-1}\Sigma^{\mathcal{I}\mathcal{I}^*}\right]\Big), \tag{2.90}$$

where I_N denotes an $N \times N$ identity matrix.

The above expression completely captures the posterior distribution of the Gaussian process. This computation requires the inversion of $N \times N$ matrices, and the product of matrices of size $N^* \times N$, $N \times N$, and $N \times N^*$, where $N \triangleq |\mathcal{I}|$ and $N^* \triangleq |\mathcal{I}^*|$. The computational complexity for general, non-sparse, covariance matrices is then $O(N^3 + N^2N^* + NN^{*2})$, and can be efficiently computed when N and N^* are both fairly small.

This exact approach was used to infer the results of Example 2.1.1, where the x_i's were noisy 2D coordinates of a measurement of the underlying ground truth trajectory, g. While exact inference is desirable, this approach clearly does not scale well. For example, in images, N and N^* can both be on the order of a million. In such regimes, matrices can hardly be stored, let alone be inverted or multiplied.

Covariance Kernels

Because exact inference may not be tractable, it will be convenient to select a *stationary* covariance kernel to aid approximate inference. Here, stationarity implies that the covariance only depends on the relative difference in locations, i.e., $\kappa(i, j) \equiv \kappa(i - j)$. We begin this discussion with two examples of covariance kernels. In this thesis, we will only make use of zero-mean GPs with stationary covariance kernels, which we denote as

$$g \sim \mathrm{GP}(g; \kappa). \tag{2.91}$$

The locations will typically be the image domain, but will be specified if it they are not.

The choice of a particular covariance kernel greatly affects samples from the resulting Gaussian processes. There are many different parametric forms for stationary covariance kernels. Here, we consider two commonly used classes: the squared-exponential kernel, and the class of Matérn kernels.

The squared-exponential (SE) kernel (sometimes referred to as the radial-basis function) uses the following Gaussian function as the kernel

$$\kappa_{\mathrm{SE}}(r) = \sigma_g^2 \exp\left[-\frac{r^2}{2l^2}\right], \tag{2.92}$$

where we have denoted the change in locations as $r \triangleq i - j$. The SE kernel contains two parameters: the signal variance, σ_g^2, and the characteristic length-scale, l. The signal variance essentially just multiplies the entire Gaussian process by a constant scale factor. The characteristic length-scale determines the rate at which the correlation decays. Samples from Gaussian processes with a SE kernel are shown in Figure 2.15 for three different characteristic length-scales.

We note that the characteristic length-scale only scales the rate that the correlation decays. While it may appear that samples from $l = 10$ are much smoother than $l = 1$, a magnification of the $l = 1$ case by $\times 5$ and $\times 10$ in Figure 2.16 shows this to not be the case. The resulting magnified GPs look similar to the $l = 5$ and $l = 10$ case. In other words, the smoothness is a function of the *class* of covariance kernel; the characteristic

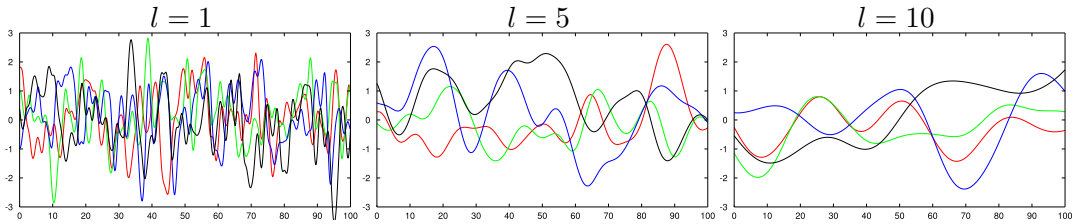


Figure 2.15: Four samples from Gaussian processes with a squared exponential kernel at three different characteristic length-scales, l.

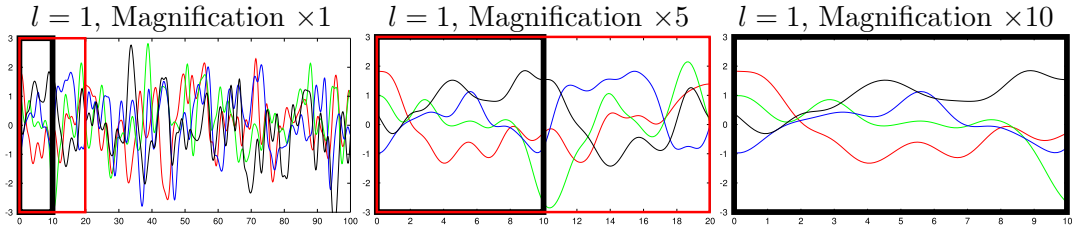


Figure 2.16: A visualization of samples under different magnifications. The left plot shows 4 samples of the GP with $l = 1$. The right plots show magnifications of the left plot at $\times 5$ and $\times 10$. The corresponding magnified regions are indicated with bold rectangles.

length-scale only scales the space.

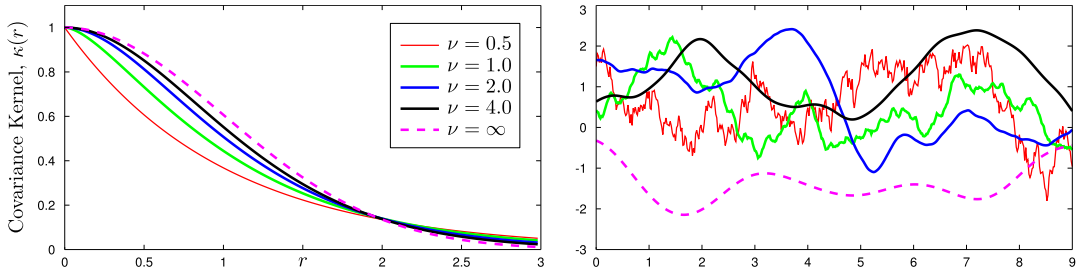
The SE kernel is often very smooth, and [111] argues that it may even be too smooth for physical processes. In such cases, the Matérn class of covariance kernels may be more useful. The covariance kernel for the Matérn class can be expressed as

$$\kappa(r\ ;\ \sigma_g^2, \nu, l) = \sigma_g^2\ \frac{2^{1-\nu}}{\Gamma(\nu)}\Big(\frac{r\sqrt{2\nu}}{l}\Big)^{\nu} K_\nu\Big(\frac{r\sqrt{2\nu}}{l}\Big), \tag{2.93}$$

where there are three parameters: σ_g^2, l, and ν. The first two hyper-parameter capture the same meaning as in the SE kernel case. The extra ν controls the smoothness of the kernel, where higher values correspond to smoother functions. The class of Matérn functions captures a wide range of smoothness. For example, in the limit, as ν goes to ∞, the Matérn kernel is equivalent to the squared exponential kernel. When $\nu = 0.5$, the covariance kernel is equal to an exponential function. Some kernels and resulting samples are shown in Figure 2.17.

Approximate Sampling and Inference via Equivalent Kernels

As stated previously, posterior distributions on GPs can be expressed analytically, but cannot be computed efficiently for large datasets. In this section, we briefly consider the problem of sampling from the posterior of a high-dimensional Gaussian process (GP) under the following two conditions: (1) a stationary covariance kernel is used, and (2)



(a) Covariance Kernels (b) Sample Realizations

Figure 2.17: Covariance kernels and samples from the Matérn class of kernels. All kernels have $l = 1$ and $\sigma_g^2 = 1$ but vary the hyper-parameter ν.

the observation locations and target locations are both the same and form a lattice grid.

Let g be a sample from a GP on a 2D grid of equally spaced locations. This finite realization of g can then be expressed as a joint Gaussian with

$$g \sim \mathcal{N}(g; 0, \Sigma^g)\,, \tag{2.94}$$

where Σ^g is the covariance obtained by evaluating the kernel, κ, at the grid locations. Let x be noisy observations of g as follows

$$x \sim \mathcal{N}\big(x; g, \sigma_x^2\big)\,. \tag{2.95}$$

Applying Equation (2.90) results in the following posterior distribution:

$$p(g|x) = \mathcal{N}\big(g\ ;\ \Sigma^g \Lambda^{g+x} x\ ,\ \Sigma^g - \Sigma^g \Lambda^{g+x} \Sigma^g\big) = \mathcal{N}(g; \mu^*, \Sigma^*)\,, \tag{2.96}$$

where we define $\Lambda^{g+x} = (\Sigma^{g+x})^{-1} \triangleq (\Sigma^g + \sigma_x^2 \mathrm{I}_N)^{-1}$.

In the limit, as the size of the 2D grid approaches ∞, equivalent kernel methods [110] state that the posterior mean approaches

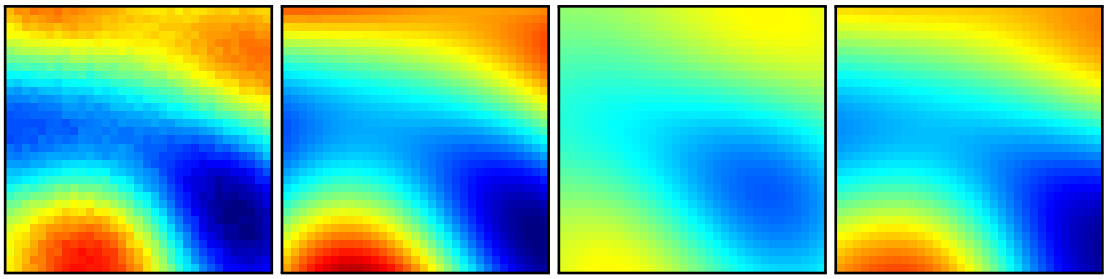
$$\lim_{N \to \infty} \mu^* = h_\mu * x, \tag{2.97}$$

where the *equivalent kernel*, h_μ, is defined to be

$$h_\mu = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(\kappa)}{\mathcal{F}(\kappa) + \sigma_x^2}\right), \tag{2.98}$$

and $\mathcal{F}$ and $\mathcal{F}^{-1}$ denote the Fourier and inverse Fourier transforms, respectively. We exploit this limiting behavior by approximating μ^* with

$$\mu^* \approx h_\mu * \hat{x}, \tag{2.99}$$



(a) Observation x (b) True Posterior (c) Zero-Padded (d) Nearest-Padded

Figure 2.18: Comparing inference methods for Gaussian processes. (a) A noisy observation generated from the GP. (b) The true mean of the posterior GP. We note that the true mean can only be computed for very small images (30×30 in this case) since it requires the inversion of an $N \times N$ matrix, where N is the number of pixels. (c) Equivalent kernel approximation with zero-padding outside of the borders. (d) Equivalent kernel approximation after padding boundaries with the nearest value of x.

where $\hat{x}$ denotes padding the boundaries of x with the nearest values. We show results with and without padding in Figure 2.18.

The above expression approximates the MAP solution to the latent Gaussian process. To accurately draw a sample from the posterior, we must also categorize the posterior covariance. We first remind the reader of the useful property of Gaussian random variables:

$$g \sim \mathcal{N}\left(\mu_g^*, \Sigma_g^*\right) \equiv \mu_g^* + \mathcal{N}\left(0, \Sigma_g^*\right). \tag{2.100}$$

That is, the randomness in g can be completely captured by the second term involving the covariance. Additionally, it is well known that if $y \sim \mathcal{N}(0, \mathrm{I}_N)$ (i.e., a vector of standard normals), then multiplying this vector by a matrix A results in correlated Gaussian variables:

$$Ay \sim \mathcal{N}\left(0, AA^\top\right). \tag{2.101}$$

While this relationship is most often used to generate multivariate normals by setting A to be the Cholesky decomposition of the covariance matrix, the identity holds for any matrix, A. We first find the corresponding filter that is equivalent to multiplying a vector with Σ_g^* as

$$h_{\Sigma,2} = \mathcal{F}^{-1}\left\{\mathcal{F}\left\{\kappa\right\} - \frac{\mathcal{F}\left\{\kappa\right\}^2}{\mathcal{F}\left\{\kappa\right\} + \sigma_x^2}\right\}. \tag{2.102}$$

Then, we choose a particular decomposition in Equation (2.101) where A is symmetric, resulting in:

$$h_{\Sigma} = \mathcal{F}^{-1}\left\{\sqrt{\mathcal{F}\left\{\kappa\right\} - \frac{\mathcal{F}\left\{\kappa\right\}^2}{\mathcal{F}\left\{\kappa\right\} + \sigma_x^2}}\right\}. \tag{2.103}$$

An approximate sample from the Gaussian process of Equation (2.90) with:

$$g = [h_\mu * \hat{x}] + [h_\Sigma * \mathcal{N}(0, \mathrm{I}_N) *] \tag{2.104}$$

Approximate Likelihood Computation

To to perform parameter-learning in Gaussian processes, it is common to need to calculate the likelihood. Assuming a zero-mean Gaussian process, g, the likelihood can be expressed as

$$p(g|\kappa) = |\Sigma^g|^{-\frac{1}{2}} (2\pi)^{-\frac{N}{2}} \exp\left[-\frac{1}{2} g^\top (\Sigma^g)^{-1} g\right], \tag{2.105}$$

where the covariance matrix, Σ^g is dependent on the covariance kernel, κ. This likelihood computation poses a computational difficulty because $g^\top (\Sigma^g)^{-1} g$ is a large vector-matrix-vector product, and $|\Sigma^g|$ is the determinant of a large matrix. The vector-matrix-vector product can be well approximated using equivalent kernel methods and FFTs. We now discuss the computation of the determinant.

We note that because Toeplitz matrices approach circulant matrices in the infinite data regime, one can approximate determinants of large Toeplitz matrices. Let C denote a symmetric Toeplitz matrix, defined as

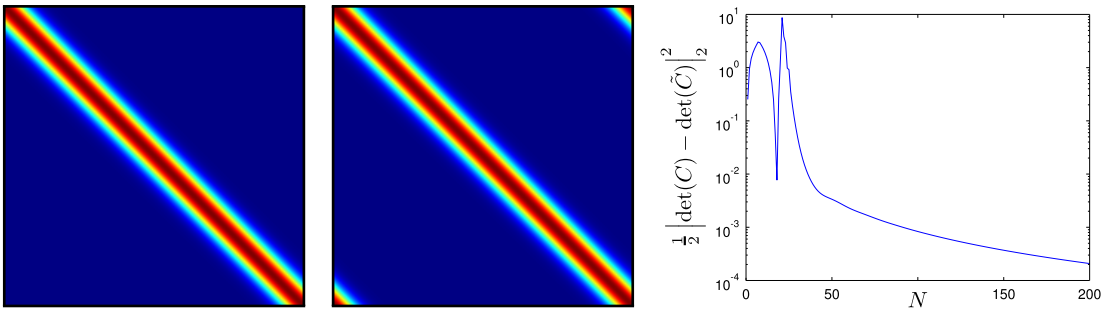
$$C = \begin{bmatrix} c_0 & c_1 & \dots & c_{N-1} & c_N \\ c_1 & c_0 & \dots & c_{N-2} & c_{N-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ c_{N-1} & c_{N-2} & \dots & c_0 & c_1 \\ c_N & c_{N-1} & \dots & c_1 & c_0 \end{bmatrix}. \tag{2.106}$$

If c_i is monotonically decreasing to 0 (as is the case for most covariance kernels of interest), we can then approximate the Topelitz matrix with the following circulant matrix with

$$\tilde{C} = C + \begin{bmatrix} 0 & c_N & \dots & c_2 & c_1 \\ c_N & 0 & \dots & c_3 & c_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ c_2 & c_3 & \dots & 0 & c_N \\ c_1 & c_2 & \dots & c_N & 0 \end{bmatrix} \tag{2.107}$$

$$= \begin{bmatrix} c_0 & c_1 + c_N & \dots & c_2 + c_{N-1} & c_1 + c_N \\ c_1 + c_N & c_0 & \dots & c_3 + c_{N-2} & c_2 + c_{N-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ c_2 + c_{N-1} & c_3 + c_{N-2} & \dots & c_0 & c_1 + c_N \\ c_1 + c_N & c_2 + c_{N-1} & \dots & c_1 + c_N & c_0 \end{bmatrix} \tag{2.108}$$

Because the determinant of a circulant matrix is known to be the product of its Fourier



(a) Toeplitz, C (b) Circulant, $\tilde{C}$ (c) Approximation Error

Figure 2.19: Approximating determinants of symmetric Toeplitz matrices.

series coefficients, the determinant of the covariance matrix can be calculated efficiently using the Fast Fourier Transform. We visualize the approximation in Figure 2.19 for varying amounts of observations, N. We note that this approximation is related to the the Szegő limit theorems.

■ 2.9.2 Dirichlet Processes

In this section, we give a brief overview of Dirichlet processes. For a more in-depth understanding, we refer the reader to [112]. The seminal work of Ferguson [33] proves the existence of the random process defined in Definition 2.9.1.

Definition 2.9.1 (Dirichlet Process)**.** Let H be a measure on a measureable space, Ω. If for any finite partition, $(A_1, A_2, \ldots, A_K)$ of the space, the measure, G on the partition follows the following Dirichlet distribution

$$(G(A_1), G(A_2), \ldots, G(A_K)) \sim \text{Dir}(\alpha H(A_1), \alpha H(A_2), \ldots, \alpha H(A_K)), \tag{2.109}$$

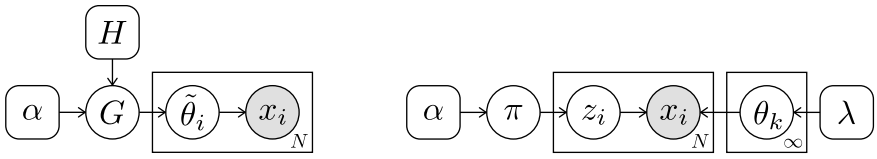
for some positive scalar α, then G is said to be a Dirichlet process with concentration parameter α and base measure H. This relationship is denoted as $G \sim \text{DP}(\alpha, H)$.

Sethuraman [106] later developed the following constructive proof of the existence of the DP.

Theorem 2.9.1 (Stick-Breaking Construction of a Dirichlet Process)**.** *Let π be an infinite length vector where each component is sampled according to the following:*

$$\beta_k \sim \text{Beta}(1, \alpha), \qquad \forall k \in \mathbb{Z}^+, \tag{2.110}$$

$$\pi_k = \beta_k \left(1 - \sum_{l=1}^{k-1} \pi_l\right) \qquad \forall k \in \mathbb{Z}^+. \tag{2.111}$$



(a) Model with atoms (b) Model with explicit labels

Figure 2.20: Two equivalent graphical models for the DPMM.

Furthermore, let each π_k *be associated with a corresponding* θ_k *drawn from*

$$\theta_k \sim H, \quad \forall k \in \mathbb{Z}^+. \tag{2.112}$$

If G *is constructed from* π *and* θ *as follows*

$$G = \sum_{k=1}^{\infty} \pi_k \delta_{\theta_k}, \tag{2.113}$$

then G *is guaranteed to be a Dirichlet process distributed according to* $G \sim \mathrm{DP}(\alpha, H)$.

Proof. Please see [106]. □

The above construction is referred to as the stick-breaking construction since one can think of each draw of π_k as breaking off part of a unit length stick. For each new stick break, π_k, β_k determines the proportion of the remainder of the stick to allocate. We note that it is common to denote the stick-breaking process as $\pi \sim \mathrm{GEM}(1, \alpha)$ (GEM stands for the authors Griffiths, Engen, and McCloskey). A draw from a Dirichlet process is therefore a set of discrete *atoms*, where the atom *locations* are drawn from the base measure H, and the heights of the atoms are drawn from the stick-breaking process.

A Dirichlet process mixture model (DPMM) uses a Dirichlet process as the prior over the model parameters. The graphical model for a DPMM is shown in Figure 2.20a. The generative model is summarized in the following steps:

$$G \sim \mathrm{DP}(\alpha, H), \tag{2.114}$$

$$\tilde{\theta}_i \sim G(\tilde{\theta}_i), \quad \forall i \in \{1, \ldots, N\}, \tag{2.115}$$

$$x_i \sim f_x(x_i; \tilde{\theta}_i), \quad \forall i \in \{1, \ldots, N\}. \tag{2.116}$$

For each data point, i, a corresponding parameter $\tilde{\theta}_i$ is first drawn from the Dirichlet process realization, G. Because G is a draw from a DP, it is composed of discrete atoms; therefore, multiple data points have non-zero probability of sharing the same value of $\tilde{\theta}_i$.

Alternatively, the Dirichlet process mixture model can be equivalently represented with an explicit indexing latent variable, z_i. This model is depicted in Figure 2.20b. In-

stead of sampling an explicit atom location from G, z_i samples an index into a particular atom. This generative model is summarized in the following steps:

$$\pi \sim \text{GEM}(1, \alpha), \tag{2.117}$$
$$\theta_k \sim f_\theta(\theta_k; \lambda) = H, \qquad \forall k \in \{1, \dots\}, \tag{2.118}$$
$$z_i \sim \text{Cat}(\pi), \qquad \forall i \in \{1, \dots, N\}, \tag{2.119}$$
$$x_i \sim f_x(x_i; \theta_{z_i}), \qquad \forall i \in \{1, \dots, N\}. \tag{2.120}$$

While this model is equivalent to the one presented in Equation (2.114)–(2.116), it is often simpler to understand since it is a generalization of the finite mixture model in Section 2.8.

Chinese Restaurant Process

Before detailing posterior inference algorithms in DPMMs, we first discuss another alternative generative model of the DPMM, called the Chinese Restaurant Process. We begin by developing the predictive distribution of z by marginalizing over the Dirichlet process. The Dirichlet process is essentially a generalization of the Dirichlet distribution. As shown in Section 2.2, the Dirichlet distribution is conjugate to categorical observations. As such, one would expect the weights associated with the stick-breaking process of the Dirichlet process to also be conjugate to the categorical observations, z. We now show that this intuition is indeed true.

Assume that the set of labels, $\{z_1, \dots, z_N\}$ take on K distinct values, each with an associated atom location θ_k. According to Definition 2.9.1, any partition of the sample space, Ω, must follow a Dirichlet distribution. One valid partition is the following

$$(A_1, \dots, A_K, A_{K+1}) = (\delta_{\theta_1}, \dots, \delta_{\theta_K}, \Omega \setminus \{\cup_{k=1}^K \delta_{\theta_k}\}), \tag{2.121}$$

where the first K partitions are singular points located at the values of θ_k. Furthermore, because each observation places a point mass, $\delta_{\theta_{z_i}}$, at the location θ_{z_i}, and there is measure α spread throughout Ω, the resulting distribution from this partition will be

$$(G(A_1), \dots, G(A_K), G(A_{K+1})) \sim \text{Dir}(N_1, \dots, N_K, \alpha). \tag{2.122}$$

According to the properties of the Dirichlet distribution described in Section 2.2, we know that marginalizing this distribution results in the following predictive distribution

$$p(\hat{z}|z_1, \dots, z_N) = \text{DirCat}(\hat{z}; N_1, \dots, N_K, \alpha) \propto \alpha \mathbb{1}[\hat{z} = K+1] + \sum_{k=1}^{K} N_k \mathbb{1}[\hat{z} = k]. \tag{2.123}$$

We note that when $\hat{z} = K+1$, this is equivalent to sampling from a new atom that was not previously used among the $\{z_1, \dots, z_N\}$ values.

The distribution on the set of labels, z, can be decomposed as

$$p(z) = p(z_1)p(z_2|z_1)p(z_3|z_1, z_2)\dots p(z_N|z_1, \dots, z_{N-1}). \tag{2.124}$$

Furthermore, because the observations are exchangeable in a DPMM, each term in this expression follows the form of the predictive distribution in Equation (2.123). As such, it is easily shown (cf. [1]) that z follows the following distribution

$$p(z) = \frac{\alpha^K \Gamma(\alpha)}{\Gamma(N)} \prod_{k=1}^{K} \Gamma(N_k). \tag{2.125}$$

The Chinese Restaurant Process (CRP) [97] describes the process of sampling from Equation (2.124) iteratively using Equation (2.123). In the CRP, each observation is represented as a customer in a restaurant. Customers come into the restaurant and either sit at a previously-occupied table or sit at a new table. The different tables partition the observations into separate clusters. Sitting at a table with another customer is equivalent to assigning both customers to the same atom, whereas sitting at an empty table corresponds to sampling a new atom. According to Equations (2.123) and (2.124), a valid partitioning from a DPMM can be drawn according to the following: customers sequentially enter the restaurant and sit at a table with probability proportional to the number of customers already seated there, or start a new table with probability proportional to α.

Collapsed-Weight Samplers

We are now properly equipped to discuss posterior inference in DPMMs. One main problem in DPMMs is that π and θ are both infinite-length vectors that cannot be explicitly instantiated. Fortunately, the CRP marginalizes over the infinite length vector π. Furthermore, if a conjugate prior is placed on θ (e.g., [16, 30, 85, 92, 126]), the parameters can also be marginalized. The only remaining latent variable, z, then has the following posterior distribution

$$p(z_i|x, z_{\setminus i}) \propto \alpha f_x(x_i; \lambda)\mathbb{1}[z_i = \hat{k}] + \sum_{k=1}^{K} N_{k\setminus i} f_x(x_i; \lambda^*(x_{\mathcal{I}_k \setminus i}))\mathbb{1}[z_i = k], \tag{2.126}$$

where $\setminus i$ denotes all indices excluding i, $N_{k\setminus i}$ are the number of elements in $z_{\setminus i}$ with label k, $\hat{k}$ is a new cluster label, $\mathcal{I}_k \triangleq \{i; z_i = k\}$ denotes the set of indices with label $z_i = k$, $f_x(\circ; \lambda)$ denotes the distribution of x when marginalizing over parameters, and $\lambda^*(\cdot)$ denotes the posterior hyper-parameters. This Gibbs sampling algorithm is summarized in Algorithm 2.6. When a non-conjugate prior is used, a computationally expensive Metropolis-Hastings step (e.g., [86, 93]) must be used when sampling the label for each data point.

Algorithm 2.6 Chinese Restaurant Process Sampling for DPMMs

1. Initialize z arbitrarily.
2. **Label inference across the data**:
   (a) Sample a random permutation of the integers in $[1, N]$.
   (b) Select the next index, i, from the permutation.
   (c) Sample a label, z_i, conditioned on all other labels from Equation (2.126).
   (d) Repeat from Step 2(b) until no indices remain.
3. Repeat from Step 2 until convergence.

Instantiated-Weight Samplers

The constructive proof of the Dirichlet processes [106] shows that a DP can be sampled by iteratively scaling an infinite sequence of Beta random variables. Therefore, posterior MCMC inference in a DPMM could, in theory, alternate between the following samplers

$$(\pi_1, \dots, \pi_\infty) \sim p(\pi|z, \alpha), \tag{2.127}$$

$$\theta_k \overset{\propto}{\sim} f_\theta(\theta_k; \lambda) \prod_{i \in \mathcal{I}_k} f_x(x_i; \theta_k), \qquad \forall k \in \{1, 2, \dots\}, \tag{2.128}$$

$$z_i \overset{\propto}{\sim} \sum_{k=1}^{\infty} \pi_k f_x(x_i; \theta_k) \mathbb{1}[z_i = k], \quad \forall i \in \{1, \dots, N\}, \tag{2.129}$$

where $\overset{\propto}{\sim}$ samples from a distribution proportional to the right side. When conjugate priors are used, the posterior distribution for cluster parameters is in the same family as the prior:

$$p(\theta_k|x, z, \lambda) \propto f_\theta(\theta_k; \lambda) \prod_{i \in \mathcal{I}_k} f_x(x_i; \theta_k) \propto f_\theta(\theta_k; \lambda^*(x_{\mathcal{I}_k})), \tag{2.130}$$

where $\lambda^*(x_{\mathcal{I}_k})$ denotes the posterior hyper-parameters for the data belonging to cluster k. Unfortunately, the infinite length sequences of π and θ clearly make this procedure impossible.

As an approximation, authors have considered the truncated stick-breaking representation [60] and the finite symmetric Dirichlet distribution [61]. The latter is summarized in Algorithm 2.7 assuming conjugate priors. These approximations become more accurate when the truncation is much larger than the true number of components, which is often unknown. When cluster parameters are explicitly sampled, these algorithms may additionally suffer from slow convergence issues. In particular, a broad prior will often result in a very small probability of creating new clusters since the probability of generating a parameter from the prior to fit a single data point is small. However, these approximations do excel in certain aspects compared to the collapsed-weight sam-

Algorithm 2.7 Finite Symmetric Dirichlet Approximation for DPMMs

1. Initialize z arbitrarily and choose a truncation, L.
2. Sample the L mixture weights from $\pi \sim \text{Dir}(N_1 + \frac{\alpha}{L}, \ldots, N_L + \frac{\alpha}{L})$.
3. Sample the L mixture parameters, each from $\theta_l \sim f_\theta(\theta_l; \lambda^*(x_{\mathcal{I}_l}))$.
4. Sample the N labels, each from $z_i \sim \sum_{k=1}^{L} \pi_l f_x(x_i; \theta_l) \mathbb{1}[z_i = l]$
5. Repeat from Step 2 until convergence.

plers. Instantiated-weight sampling algorithms can be easily parallelized since each z_i is independent. Moreover, non-conjugate priors on θ are also easier to handle since marginalization over θ is not required.

Super-Cluster Parallel Samplers

While parallelizable algorithms are of interest to the machine-learning community, the approximation required for using the finite models is undesirable. More recently, the works of [83] and [127] present an alternative parallelization scheme that does not require such approximations. These works draw on the nesting property of Dirichlet processes in the atom representation of Figure 2.20a, summarized in below.

Theorem 2.9.2 (Nesting Partitions of the DPMM)**.** *Let L be a positive integer. If a set of N observations, denoted $\{x_1, \ldots, x_N\}$, is drawn from a Dirichlet process mixture model with concentration parameter α and base measure H, the following steps form an equivalent generative process:*

$$G_l \sim \text{DP}(\tfrac{\alpha}{L}, H), \qquad l \in \{1, \ldots, L\}, \tag{2.131}$$

$$\phi \sim \text{Dir}(\phi_1, \ldots, \phi_L; \tfrac{\alpha}{L}, \ldots, \tfrac{\alpha}{L}), \tag{2.132}$$

$$g_i \sim \text{Cat}(g_i; \phi), \tag{2.133}$$

$$\tilde{\theta}_i \sim G_{g_i}(\tilde{\theta}_i), \qquad i \in \{1, \ldots, N\}, \tag{2.134}$$

$$x_i \sim f_x(x_i; \tilde{\theta}_i), \qquad i \in \{1, \ldots, N\}. \tag{2.135}$$

Proof. See [127]. □

Theorem 2.9.2 effectively partitions the parameter space of H into L disjoint sets. Each of the L partitions then forms its own Dirichlet process, and each data point is assigned to one such partition through g_i. As such, each of the L partitions is often referred to as a super-cluster. This observation is useful because it allows an algorithm to parallelize across the L super-clusters. That is, all of the data points assigned to super-cluster 1 are independent from the data points assigned to super-cluster 2. Assuming that each super-cluster contains the same number of points, this type of algorithm has the potential of having an L times speedup if L processors are

used in parallel. Unfortunately, in practice and in theory (cf. [38]), the size of each super-cluster is extremely unbalanced, and this optimistic speedup cannot be achieved. Furthermore, sampling from the super-cluster model can artificially impede burn-in because data can only move within the current super-cluster until the super-cluster assignment is resampled.

Split/Merge Sampling Algorithms

Jain and Neal [63], among many others, have noticed that both the previously discussed collapsed-weight and instantiated-weight sampling algorithms tend to converge very slowly. This has consequently motivated sampling methods for DPMMs that incorporate large moves, such as splitting a cluster into two, or merging two clusters into one. Two such methods are often used in practice: the "Restricted Gibbs Split Merge" presented in [63] and [64], and the "Sequentially-Allocated Merge Split" presented in [27].

RGSM. The Restricted Gibbs Split Merge (RGSM) algorithm, originally proposed in [63] and later extended to non-conjugate priors in [64], is a collapsed-weight sampling algorithm that uses a Metropolis-Hastings accept/reject framework to propose split and merge moves. In this section, we only consider the case where conjugate priors are used, and the resulting algorithm can collapse both the mixture weights and the mixture parameters. We will use the symbols $\natural$, $\flat$, and $\sharp$ to denote cluster indices that will be involved in splits and merges, where $\natural, \flat, \sharp \in \{1, \ldots, K\}$. A proposed merge move will merge clusters $\flat$ and $\sharp$ into cluster $\natural$. This notation is motivated by musical theory, where a flat accidental ($\flat$) combined with a sharp accidental ($\sharp$) results in a natural note ($\natural$).

Two data points, denoted j_1 and j_2 are first selected uniformly at random from the N possible observations. If, for the current configuration, $z_{j_1} = z_{j_2} = \natural$ (i.e., the points are both assigned to cluster $\natural$), then a split of cluster $\natural$ is proposed. If the converse is true (i.e., $z_{j_1} = \flat \neq z_{j_2} = \sharp$), then a merge of clusters $\flat$ and $\sharp$ is proposed.

A proposed split move is first randomly initialized by the following:

$$\hat{z}_i \sim \sum_{k \in \{\flat, \sharp\}} 0.5 \mathbb{1}[\hat{z}_i = k], \quad \forall i \in \{i; z_i = \natural\} \tag{2.136}$$

The proposed split move is then improved by running a *restricted* Gibbs iteration. That is, the data points are explored via a random permutation of the indices, where each data point is sampled from

$$\hat{z}_i \sim \sum_{k \in \{\flat, \sharp\}} N_{k \setminus i} f_x(x_i; \lambda^*(x_{\mathcal{I}_k \setminus i})) \mathbb{1}[\hat{z}_i = k]. \tag{2.137}$$

The process is repeated for M iterations and then accepted in a Metropolis-Hastings

algorithm with the following Hastings ratio:

$$H_{\text{split-}\natural} = \frac{p(\hat{z})p(x|\hat{z})}{p(z)p(x|z)} \frac{1}{q_{\text{split}}(\hat{z}|z)}, \tag{2.138}$$

where $q_{\text{split}}(\hat{z}|z)$ aggregates all the probabilities in Equation (2.137) for the last of the M iterations.

Similarly, a merge of clusters $\flat$ and $\sharp$ into $\natural$ is proposed by simply performing

$$\hat{z}_i = \natural, \quad \forall i \in \mathcal{I}_\flat \cup \mathcal{I}_\sharp, \tag{2.139}$$

and accepted with probability

$$H_{\text{merge-}\flat\sharp} = \frac{p(\hat{z})p(x|\hat{z})}{p(z)p(x|z)} \frac{q_{\text{split}}(z|\hat{z})}{1}. \tag{2.140}$$

One must be careful in computing the reverse move in this Hastings ratio, captured by $q_{\text{split}}(z|\hat{z})$. Jain and Neal have shown that one can simply sample a random permutation of the indices involved in the merge, followed by aggregating the terms of Equation (2.137). The overall steps in RGSM are summarized in Algorithm 2.8.

Algorithm 2.8 Restricted Gibbs Split Merge Sampling for DPMMs

1. Randomly select two data points, j_1 and j_2. If the points belong to the same cluster, go to Step 2 to propose a split. Otherwise, go to Step 5 to propose a merge.
2. **Split Proposal**:
   (a) Randomly initialize the new labels according to Equation (2.136).
   (b) Sample a random permutation of the indices involved in the split proposal.
   (c) Loop through the permutation and sample assignments from Equation (2.137).
   (d) Repeat from Step 2(b) $M - 1$ times.
   (e) Accept or reject the proposal with the Hastings ratio of Equation (2.138).
   (f) Repeat from Step 1.
3. **Merge Proposal**:
   (a) Deterministically assign the new labels according to Equation (2.139).
   (b) Sample a random permutation of the indices involved in the merge proposal.
   (c) Loop through the permutation and aggregate probabilities in Equation (2.137).
   (d) Accept or reject the proposal with the Hastings ratio of Equation (2.140).
   (e) Repeat from Step 1.

SAMS. The Sequentially-Allocated Merge Split (SAMS) algorithm, originally proposed in [27], improves upon RGSM by only requiring one pass through the data as opposed to M. The only part of the algorithm that is changed is the construction of a

split proposal. As such, we only review the specific split construction here.

Conditioned on the two randomly selected data points that currently belong to the same cluster, a split is constructed with the steps outlined in Algorithm 2.9. The resulting Hastings ratios are essentially the same as Equations (2.138) and (2.140), except that q_{split} must be calculated by aggregating the probabilities in Equation (2.141).

Algorithm 2.9 Sequentially-Allocated Split Proposal for DPMMs

1. Assign $\hat{z}_{j_1} = \flat$ and $\hat{z}_{j_2} = \sharp$.
2. Choose a random permutation of all other indices in $\mathcal{I}_{\natural} \setminus \{j_1, j_2\}$.
3. Take the next index from the random permutation, denoted i.
4. Assign the data point to cluster $\flat$ or $\sharp$ based on

$$\hat{z}_i \overset{\propto}{\sim} \sum_{k=\{\flat,\sharp\}} N_k f_x(x_i; \lambda^*(x_{\mathcal{I}_k})) \mathbb{I}[\hat{z}_i = k], \tag{2.141}$$

   where N_k and $\lambda^*(x_{\mathcal{I}_k})$ depend only on *previously* assigned data points.
5. Repeat from Step 3 until all points have been assigned.

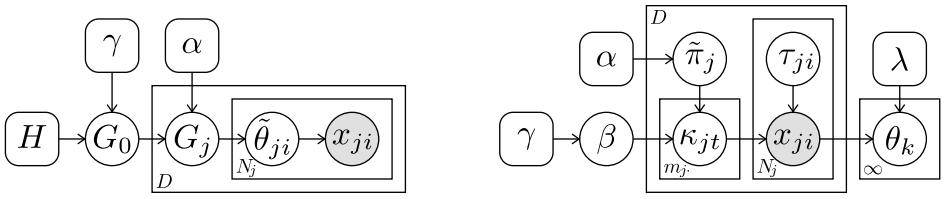
■ 2.9.3 Hierarchical Dirichlet Process

The Hierarchical Dirichlet process [116] is an extension of the Dirichlet process that allows for groups of data to share cluster statistics. Since the Hierarchical Dirichlet process (HDP) is often used in topic modeling, we motivate this section with this specific problem. Suppose a corpus of documents exist, where each document consists of a bag of words, and each word is associated with a topic. For example, 90% of the words in one document might be about finance and 10% about economics. Another document might have 70% of the words be about economics, 20% about politics, and 10% about education. In topic modeling, one would like to reason about the different topics that exist in the corpus and the proportion of each topic that exists in each document. The HDP is a generative probabilistic model that captures exactly these aspects.

There are many different ways to represent an HDP. We cover three such representations here: the explicit atom representation, the Chinese restaurant franchise representation, and the direct assignment representation.

Explicit Atom Representation

The graphical model capturing the dependencies for the explicit atom representation is depicted in Figure 2.21a. The generative model for the explicit atom representation is as follows. A global set of atoms and weights, G_0, is drawn from a Dirichlet process with concentration parameter γ and base measure H according to the stick-breaking



(a) Explicit Atoms (b) Chinese Restaurant Franchise

Figure 2.21: Graphical models for the explicit atom and Chinese restaurant franchise formulations of the HDP.

construction of Theorem 2.9.1:

$$G_0 \sim \mathrm{DP}(\gamma, H). \tag{2.142}$$

For each document, j, a document-specific set of atoms and weights, G_j, is drawn from a Dirichlet process with concentration parameter α and base measure G_0:

$$G_j \sim \mathrm{DP}(\alpha, G_0), \quad \forall j \in \{1, \ldots, D\}. \tag{2.143}$$

We note that because the base measure, G_0, on the document-specific DP is discrete, G_j is guaranteed to draw the same atom from G_0 multiple times. Each word is then assigned a specific topic distribution with $\tilde{\theta}_{ji}$, drawn from G_j:

$$\tilde{\theta}_{ji} \sim G_j(\tilde{\theta}_{ji}) \quad \forall j \in \{1, \ldots, D\}, \forall i \in \{1, \ldots, N_j\}. \tag{2.144}$$

Finally, each word is drawn from its assigned topic distribution:

$$x_{ji} \sim f_x(x_{ji}; \tilde{\theta}_{ji}) \quad \forall j \in \{1, \ldots, D\}, \forall i \in \{1, \ldots, N_j\}. \tag{2.145}$$

Chinese Restaurant Franchise Representation

An equivalent representation of the HDP breaks out the atoms and explicitly instantiates indices to each atom. This representation is depicted in Figure 2.21b. More precisely, $G_0 \sim \mathrm{DP}(\gamma, H)$ is a discrete distribution sampled using a stick-breaking construction, and can be expressed as

$$G_0 = \sum_{k=1}^{\infty} \beta_k \delta_{\theta_k}. \tag{2.146}$$

As such, β and θ can be equivalently be drawn from

$$\beta \sim \mathrm{GEM}(\gamma), \tag{2.147}$$

$$\theta_k \sim f_\theta(\theta_k; \lambda) = H, \quad \forall k \in \{1, 2, \ldots\}. \tag{2.148}$$

Similarly, for each document, G_j can also be sampled according to a stick-breaking process. However, as noted to previously, because the base measure G_0 is discrete, resulting DPs will necessarily draw exactly the same location. If we express G_j as

$$G_j = \sum_{t=1}^{\infty} \tilde{\pi}_{jt} \delta_{\tilde{\phi}_t}, \tag{2.149}$$

this repeating of atoms causes the set of atom locations $\{\tilde{\phi}_1, \tilde{\phi}_2, \dots\}$ to contain repeated values (e.g., $\tilde{\phi}_1$ might be equal to $\tilde{\phi}_2$). Since $\tilde{\phi}_t$ must be an element of the infinite set of global atom locations, $\{\theta_1, \theta_2, \dots\}$, we introduce an auxiliary variable, κ_{jt}, that explicitly assigns the τ^{th} atom of G_j to the the k_{jt}^{th} global atom

$$G_j = \sum_{t=1}^{\infty} \tilde{\pi}_{jt} \delta_{\theta_{\kappa_{jt}}}. \tag{2.150}$$

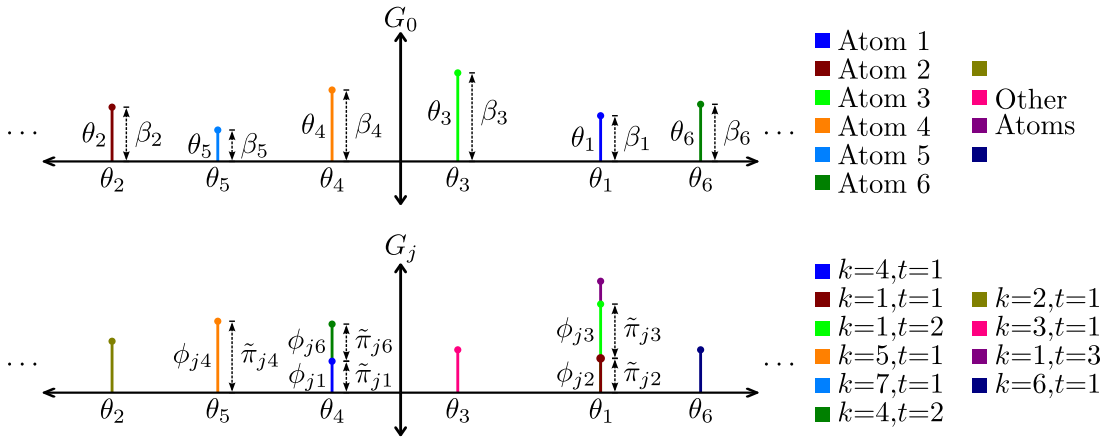
A visualization of these Dirichlet processes are shown in Figure 2.22a.

This representation is named the Chinese Restaurant Franchise (CRF) by [116] as an extension to the Chinese Restaurant Process (CRP) in DPMMs. The equivalent analogy for a CRF is as follows. Each document is represented by a restaurant, and the corpus of documents represent the entire franchise of restaurants. Similar to the CRP, A customer enters a restaurant (or, in topic modeling, a word in a document) and sits at table. The CRF then departs from a CRP by assigning a *dish* to each non-empty table from a global set of possible dishes. Tables within the same restaurant and across the different restaurants can all be assigned the same dish. Two customers that are eating the same dish (regardless of their table or restaurant) then belong to the same cluster.

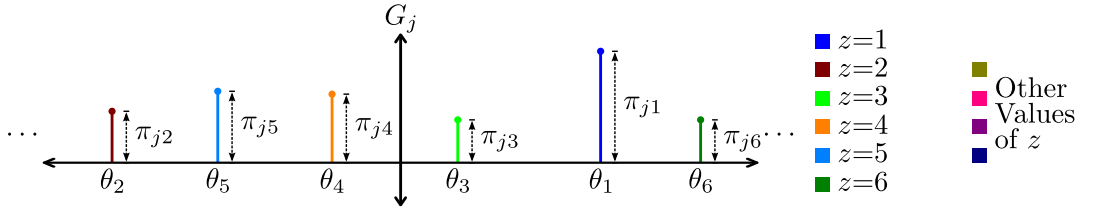
The entities of the CRF relate to the HDP as follows. Customer i in restaurant j sits at table τ_{ji}. Table t in restaurant j corresponds to the j^{th} atom of G_j. The dish served to table t, denoted by ϕ_{jt} is exactly the same as one of the atoms in G_0. Assuming ϕ_{jt} takes on the value of $\theta_{k_{jt}}$, we then denote the dish label assigned to table t as κ_{jt}. The resulting graphical model for the CRF representation with explicit assignments is depicted in Figure 2.22b.

We note that the notation of τ and κ slightly departs from the original presentation in [116]. In the original presentation, t denoted the index of a table in a restaurant, and t_{ji} denoted the table assignment to customer i in restaurant j. Similarly, k denoted the index of a particular dish, and k_{jt} denoted the dish assigned to table t in restaurant j. We find this overloaded notation to be quite confusing since it is not clear as to whether t refers to an index of a table or to the set of all table assignments. Consequently, we have introduced τ and κ to be the actual set of table and dish assignments, respectively, and t and k to be indices of tables and dishes, respectively.

It will be useful to define counts, as was done in [116]. We use n_{jtk} to denote the number of customers in restaurant j sitting at table t eating dish k. Since each table is



(a) Explicit Atom and Chinese Restaurant Franchise Representations



(b) Direct Assignment Representation

Figure 2.22: A visualization of draws from HDPs. The top plot shows the global Dirichlet process drawn from the base measure, H. (a) and (b) show a document-level Dirichlet process drawn from the base measure, G_0. (a) is the explicit atom or Chinese restaurant franchise representation that has repeated atoms, and (b) is the simplified, direct assignment representation that aggregates repeated atoms.

only assigned a single dish, n_{jtk} for a fixed j and t will only be non-zero at one value of k. Additionally, we use m_{jk} to denote the number of tables in restaurant j that are serving dish k. Marginalized counts that sum over a particular index are denoted with a dot. For example, $n_{jt\cdot}$ counts the number of customers in restaurant j sitting at table t, $n_{j\cdot\cdot}$ counts the number of customers in restaurant j, and $m_{j\cdot}$ counts the total number of tables in restaurant j. For shorthand, we also denote the number of customers per restaurant as $N_j \triangleq n_{j\cdot\cdot}$.

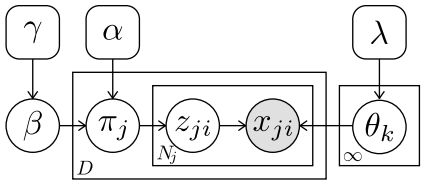


Figure 2.23: Direct assignment representation of the HDP.

The equivalent generative model for the CRF is then the following:

$$\beta \sim \text{GEM}(1, \gamma), \tag{2.151}$$
$$\kappa_{jt} \sim \text{Cat}(\kappa_{jt}; \beta), \qquad \forall j \in \{1, \dots, D\}, t \in \{1, 2, \dots\}, \tag{2.152}$$
$$\theta_k \sim f_\theta(\theta_k; \lambda) = H, \quad \forall k \in \{1, 2, \dots\} \tag{2.153}$$
$$\tilde{\pi}_j \sim \text{GEM}(1, \alpha), \qquad \forall j \in \{1, \dots, D\}, \tag{2.154}$$
$$\tau_{ji} \sim \text{Cat}(\tau_{ji}; \tilde{\pi}_j), \qquad \forall j \in \{1, \dots, D\}, i \in \{1, \dots, N_j\}, \tag{2.155}$$
$$x_{ji} \sim f_x(x_{ji}; \theta_{k_{jt_{ji}}}), \quad \forall j \in \{1, \dots, D\}, i \in \{1, \dots, N_j\}. \tag{2.156}$$

When a conjugate prior is used on θ, one can use the predictive distributions of the CRF to marginalize over β, $\tilde{\pi}$, and θ (cf. [116]) to perform posterior inference via Gibbs sampling. Since this method is more complicated than inference in the direct assignment model (which we now discuss), we do not explain the approach here.

Direct Assignment Representation

The repeated atoms in the previous discussion complicate the indexing required in the CRF representation. We now present the direct assignment (DA) representation, which collapses the repeated atoms into a single atom. Atoms from this equivalent representation are visualized in Figure 2.22b.

The graphical model for the DA representation of the HDP is shown in Figure 2.23. The main idea in the DA representation is to aggregate atoms at the same location. The resulting aggregated atom then has an associated weight according to

$$\pi_{jk} = \sum_{t=1}^{m_{j\cdot}} \tilde{\pi}_{jt} \mathbb{I}[\kappa_{jt} = k]. \tag{2.157}$$

Using Definition 2.9.1, one can show that π_j is then distributed according to $\text{DP}(\alpha, \beta)$ The new random variable, z_{ji}, then directly assigns the data point x_{ji} to cluster z_{ji}.

The resulting generative model is summarized as follows:

$$\beta \sim \text{GEM}(1, \gamma), \tag{2.158}$$
$$\theta_k \sim f_\theta(\theta_k; \lambda) = H, \quad \forall k \in \{1, 2, \dots\} \tag{2.159}$$
$$\pi_j \sim \text{DP}(\alpha, \beta), \quad \forall j \in \{1, \dots, D\}, \tag{2.160}$$
$$z_{ji} \sim \text{Cat}(z_{ji}; \pi_j), \quad \forall j \in \{1, \dots, D\}, i \in \{1, \dots, N_j\}, \tag{2.161}$$
$$x_{ji} \sim f_x(x_{ji}; \theta_{z_{ji}}), \quad \forall j \in \{1, \dots, D\}, i \in \{1, \dots, N_j\}. \tag{2.162}$$

We now consider the posterior distributions of each latent variable. Unfortunately, unlike the CRF representation, the posterior distribution for β cannot be expressed analytically conditioned only on π. As such, Teh et al. [116] propose to first sample the counts m prior to sampling β, according to [1]:

$$p(m_{jk}|z, \beta) = \frac{\Gamma(\alpha\beta_k)}{\Gamma(\alpha\beta_k + n_{j\cdot k})} s(n_{j\cdot k}, m_{jk})(\alpha\beta_k)^{m_{jk}}, \tag{2.163}$$

where $s(n, m)$ are unsigned Stirling numbers of the first kind.

Using conjugacy of the Dirichlet with the Categorical distribution, the posterior distribution on β conditioned on m is then

$$p(\beta_1, \dots, \beta_K, \beta_{K+1}|m) = \text{Dir}(\beta_1, \dots, \beta_K, \beta_{K+1}; m_{\cdot 1}, \dots, m_{\cdot K}, \gamma), \tag{2.164}$$

where we have assumed that K clusters exist in the current realization of z, and that β_{K+1} aggregates all the β's associated with empty-clusters. We note that this definition of β has slightly changed, since they are now defined over explicit partitions of the sample space.

Assuming that f_θ is conjugate to f_x, we can marginalize over θ in the label assignment, precluding the need to explicitly instantiate θ. Furthermore, since π_j is Dirichlet and conjugate to the Categorical random variable z_{ji}, π_j can also be marginalized. This results in the following posterior distribution for z_{ji}

$$\begin{aligned} p(z_{ji}|x, z_{\setminus(ji)}) \propto\ & \alpha\beta_{K+1} f_x(x_{ji}; \lambda) \mathbb{1}[z_{ji} = K + 1] \\ & + \sum_{k=1}^{K} (n_{(j\cdot k)\setminus(ji)} + \alpha\beta_k) f_x(x_{ji}; \lambda^*(x_{\mathcal{I}_k \setminus (ji)})) \mathbb{1}[z_{ji} = k], \end{aligned} \tag{2.165}$$

where $K+1$ indicates a new cluster, the subscript $\mathcal{I}_k$ denotes all indices $j, i \in \{j, i; z_{ji} = k\}$, and the subscript $\setminus(ji)$ excludes index ji.

These posterior distributions fully characterize a Gibbs sampler. The resulting sampling procedure is summarized in Algorithm 2.10.

Algorithm 2.10 Direct Assignment Sampling for HDPs

1. Initialize z and β arbitrarily, but ensuring that they are consistent with each other.
2. **Label inference across the corpus**:
   (a) Sample a random permutation of the integers in $[1, D]$ indicating documents.
   (b) Select the next document index, j, from the permutation.
   (c) **Label inference within a document:**
      i. Sample a random permutation of the integers in $[1, N_j]$ indicating words.
      ii. Select the next word index, i, from the permutation.
      iii. Sample a label, z_{ji}, conditioned on all other labels from Equation (2.165).
      iv. Repeat from Step 2(c)ii until no word indices remain.
   (d) Repeat from Step 2(a) until no document indices remain.
3. Sample dish counts, m, from Equation (2.163).
4. Sample global topic weights, β, from Equation (2.164).
5. Repeat from Step 2 until convergence.

Finite Symmetric Dirichlet Approximation

Fully instantiated inference in the DA model is complicated by the infinite-length vectors β, π, and θ. Similar to the DPMM, one can use the finite model approximations of [61] to approximate the fully instantiated model in an HDP (as was done in [34]). More precisely, the prior distribution on β is approximated with the following finite symmetric Dirichlet distribution:

$$p(\beta) \approx \text{Dir}(\beta_1, \ldots, \beta_L; \tfrac{\gamma}{L}, \ldots, \tfrac{\gamma}{L}). \tag{2.166}$$

The posterior on β again must depend on the auxiliary dish counts, m, which has the same distribution as Equation (2.163). This results in the following posterior distribution for β:

$$p(\beta|m) = \text{Dir}(\beta_1, \ldots, \beta_L; m_{\cdot 1} + \tfrac{\gamma}{L}, \ldots, m_{\cdot L} + \tfrac{\gamma}{L}). \tag{2.167}$$

Similarly, the posterior on π can be expressed as

$$p(\pi_j|\beta, z) = \text{Dir}(\pi_{j1}, \ldots, \pi_{jL}; n_{j\cdot 1} + \alpha\beta_1, \ldots, n_{j\cdot L} + \alpha\beta_L), \quad \forall j \in \{1, \ldots, D\}, \tag{2.168}$$

and the posterior on θ can be expressed as

$$p(\theta_k|x, z) \propto f_\theta(\theta_k; \lambda) \prod_{ji \in \mathcal{I}_k} f_x(x_{ji}; \theta_k) = f_\theta(\theta_k; \lambda^*(x_{\mathcal{I}_k})), \quad \forall k \in \{1, \ldots, L\} \tag{2.169}$$

This approximate inference method has the benefit of being highly parallelizable. Conditioned on the document-specific weights, π_j, and the topic parameters, θ, each word

can be sampled in parallel according to

$$z_{ji} \overset{\propto}{\sim} \sum_{k=1}^{L} \pi_{jk} f_x(x_{ji}; \theta_k) \mathbb{I}[z_{ji} = k]. \tag{2.170}$$

We outline the steps of the entire approximate method in Algorithm 2.11.

Algorithm 2.11 Finite Symmetric Dirichlet Approximation for HDPs

1. Initialize z and β arbitrarily and choose a truncation, L.
2. **Global topic parameters and weights**:
   (a) Sample the LD dish counts, m, from Equation (2.163).
   (b) Sample the global topic weights, β, from Equation (2.167).
   (c) Sample the L topic parameters, θ_k, from Equation (2.169).
3. **Label inference across the corpus**:
   (a) Select the next document index, j.
   (b) Sample the document-specific weights, π_j, from Equation (2.168).
   (c) **Label inference within a document:**
      i. Select the next word index, i.
      ii. Sample a label, z_{ji}, from Equation (2.170).
      iii. Repeat from Step 3(c)i until no word indices remain.
   (d) Repeat from Step 3(a) until no document indices remain.
4. Repeat from Step 2 until convergence.

Chapter 3

Implicit Shapes and Discrete MRFs

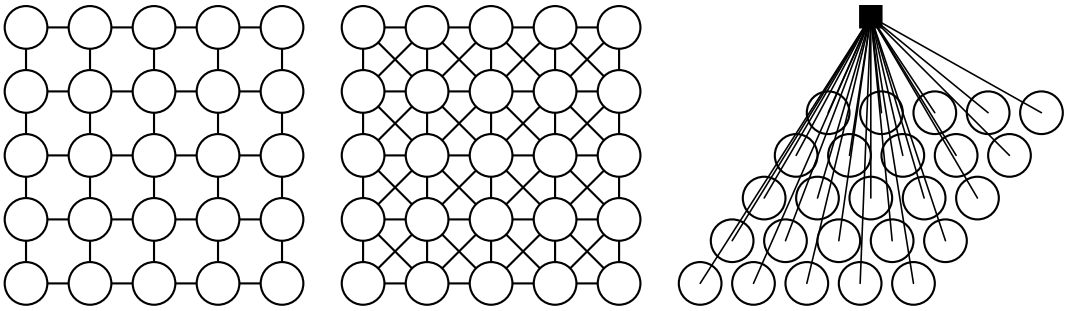
In this chapter, we consider the problem of sampling a distribution of 2-dimensional shapes. We focus on a particular implicit representation of the shape that is expressed as a set of binary variables located on a lattice grid. The resulting shape is implicitly defined by the boundary of the binary mask. We will also consider the case of sampling K shapes jointly, where the latent variables of interest take on 1 of K values instead of the simplified binary representation.

Because our shape representation is simply a discrete set of variables defined on a lattice grid, the sampling methods developed in this chapter can be generally applied to arbitrary discrete Markov random fields (MRF). While we focus on the problem of image segmentation, extensions to arbitrary MRFs is straightforward.

Implicit shape representations are useful since they eschew explicit curve and surface parametrizations while allowing topological changes. An "implicit shape representation" typically refers to using the level-set based representation (cf. [95]) which we briefly described in Section 2.6. Certain forms of prior knowledge on shapes, such as a curve-length penalty or balloon forces, are easily incorporated in a level-set representation. However, priors such as the curve-length penalty are sometimes difficult to represent in an MRF since they are defined on the underlying continuous curve. Level-set methods also typically have the benefit of being accurate to a sub-pixel level. That is, the curve that divides region does not reside on a pixel level grid since it is implicitly defined in a continuous space. However, in natural image segmentation problems, sub-pixel accuracy is often unnecessary.

In this thesis, we use the term "implicit shape" to define a broader class of shapes that contain boundaries that are not explicitly parametrized (e.g., with splines or markers). The particular implicit shape representation that we will use is quite similar to a level-set representation; for each location on a lattice-grid of pixels, we place a binary variable that indicates the region label. If the binary variables are thought of as a set of dependent random variables, the resulting model forms a Markov random field with a graphical structure defined by the problem. Example graphical structures are depicted in Figure 3.1. As we shall see, the graphical structure depicted in Figure 3.1c is useful for approximating curve-length, a commonly-used term in level-set methods.

The space of implicitly defined shapes on lattice grids quickly becomes intractable since it grows exponentially with the size of the grid. In complicated distributions



(a) 4-Connected MRF (b) 8-Connected MRF (c) 24-Connected MRF

Figure 3.1: Example MRF structures. (a) depicts a 4-connected neighborhood structure often used in Ising models [40]. (b) is a more complicated, 8-connected neighborhood structure. (c) depicts a 24-connected neighborhood structure represented as a factor graph.

such as these, it is common in Bayesian formulations to use Markov chain Monte Carlo (MCMC) methods. MCMC methods enable one to reason about entire complex *distributions* instead of relying on a single point-estimate, such as the maximum *a posteriori* (MAP) value obtained using energy minimization. Integrating these two formalisms faces two distinct challenges. First, the high dimensionality of implicit representations induces a large configuration space resulting in slow convergence for naïve implementations. Second, certain technical conditions induce a correspondence problem that, in prior efforts [23, 31], has overly constrained the applicable class of curves (e.g., simply connected shapes). In this chapter, we address these and additional issues to develop a *computationally tractable* MCMC sampling algorithm over the space of implicitly defined shapes. This, in turn, simplifies the estimation of marginal statistics defined over the distribution segmentations.

Our primary contribution in this field is to develop a *computationally tractable* MCMC sampling algorithm by which optimization-based methods of level-set formulations may be analyzed within a Bayesian framework. As with many MCMC samplers, the proposal distribution has a critical impact on the "burn-in" time, i.e., convergence to the stationary distribution from which we would like to sample. In addition to relaxing constraints on the allowed shape class (as compared to previous methods [23, 31]), we suggest a design method for the proposal distribution that dramatically reduces the burn-in time. In summary, the contributions of this work are threefold. First, we develop an MCMC sampling method for implicit shape representations and show how to control the topology of the resulting shape. Second, we extend the approach to the case of K-ary segmentations. Third, we achieve these improvements while simultaneously accelerating the sampling procedure by *orders* of magnitude over previous methods. The proposed method turns out to be a generalization of traditional Gibbs sampling.

3.1 Related Work

Sampling from the space of implicit segmentations has been suggested previously within a Metropolis-Hastings MCMC framework (cf. Section 2.5.1). Fan et al. [31] developed a hybrid method that alternates between implicit (level-set) and explicit (marker-based) representations of a single, simply connected shape. Their proposal distribution generates a sample perturbation over a set of marker points. Upon completion, the new sample is converted into an implicit form by resolving the Eikonal equation (cf. Section 2.6). While establishing the feasibility of applying MCMC methods to implicit representations, the method of [31] is constrained to binary segmentations of a single, simply connected shape. Furthermore, iterations between implicit and explicit representations incur a substantial computational burden. Fan suggests the use of jump diffusion processes [46] as a means of incorporating topological changes. However, no specific formulation is provided.

Chen and Radke [23] improved upon the method of Fan et al. by obviating the need to transition between implicit and explicit representations. They construct a smooth normal perturbation at a *single* point on the curve (denoted the "foot point") that preserves the signed distance property between proposal samples, thereby simplifying the correspondence problem and evaluation of the Hastings ratio. However, the resulting perturbations are overly smooth, and as such, explore the configuration space very slowly. As in [31] obstacles remain for incorporating topological changes, restricting this method to binary segmentations with a single simply connected shape.

When the implicitly defined shape is viewed as an MRF, another possible approach is to use a traditional Gibbs sampler [40]. This formulation requires the ability to transform a level-set based energy functional (e.g., including terms such as curve-length penalty) to a graphical model. As we shall see shortly, this formulation can be approximated efficiently. A typical Gibbs sampler selects a random variable, either at random or according to some sequence, and samples from the conditional distribution conditioned on all other variables. We outline one such Gibbs sampling algorithm when applied to a binary MRF in Algorithm 3.1, where z_i denotes one of the N possible binary variables of interest and $z_{\setminus i}$ denotes all variables except z_i. Unfortunately, Gibbs sampling is often very sensitive to the initialization and is known to converge slowly because only a single variable is sampled at a time.

Algorithm 3.1 Gibbs Sampling an MRF

1. Initialize the values of z randomly.
2. Sample a random location uniformly from $i \sim \text{Uniform}(1, N)$.
3. Sample a new z_i from the conditional distribution $z_i \sim p(z_i | z_{\setminus i})$.
4. Repeat from Step 2 until convergence.

One can alternatively consider a blocked Gibbs sampler, which samples from the conditional distribution of a *set* of random variables. Treating the set of variables as

a multi-dimensional random variable leads to a standard Gibbs sampler with a higher dimensional random variable as opposed to the naïve singleton sampling. We outline the steps of a blocked Gibbs sampler in Algorithm 3.2. We distinguish two types

Algorithm 3.2 Blocked Gibbs Sampling an MRF

1. Initialize the values of z randomly.
2. Sample a random subset of locations, denoted $\mathcal{I}$.
3. Sample a new $z_{\mathcal{I}}$ from the conditional distribution $z_{\mathcal{I}} \sim p(z_{\mathcal{I}}|z_{\setminus\mathcal{I}})$.
4. Repeat from Step 2 until convergence.

of blocked Gibbs sampling here. The first selects a set of indices that correspond to conditionally independent random variables conditioned on the other indices. For example, in a 4-connected MRF, indices that form a checkerboard pattern follow this formulation. In this type of sampling, each random variable can be sampled in parallel due to the independence relationship. However, since the conditional distribution of each single random variable is still one-dimensional, this approach does not typically help convergence.

The second type of blocked Gibbs sampling, which is of direct interest here, selects a set of indices that correspond to a *dependent* set of random variables. In many cases, this type of blocked Gibbs sampling can exhibit better convergence properties because a set of interdependent variables are sampled jointly. It is therefore desirable to choose a large set of variables to improve convergence. Unfortunately, Step 3 of Algorithm 3.2 requires one to instantiate the conditional distribution, $p(z_{\mathcal{I}}|z_{\setminus\mathcal{I}})$, which often grows exponentially with respect to the size of the block. For example, in a binary MRF, sampling from the conditional distributions requires the enumeration of $2^{|\mathcal{I}|}$ configurations.

■ 3.2 Permutation-based Gibbs-Inspired Metropolis Hastings

In this section, we present an alternative to blocked Gibbs sampling that allows for changes to a large set of variables, $\mathcal{I}$, whose complexity scales linearly with $|\mathcal{I}|$. Earlier versions of this work were originally presented in [19, 20, 22].

■ 3.2.1 Problem Statement

We begin with a formal statement of the problem. Consider a set of random variables, z, that are drawn from a prior distribution, $p(z)$, and have associated measurements, x, drawn from a distribution $p(x|z)$. The posterior distribution of interest is then

$$p(z|x) = \frac{p(z)p(x|z)}{p(x)} \tag{3.1}$$

It is often the case that $p(z)$ may only be known up to a constant factor (e.g., in general undirected graphical models). Furthermore, computing $p(x)$ is often intractable. For inference purposes, however, $p(x)$, which only depends on the observations, is a fixed scalar value for any particular realization of observations. As such, the function of interest is

$$p(z|x) \propto p^*(z)p(x|z), \tag{3.2}$$

where $p(z) = \frac{1}{Z}p^*(z)$ and Z is the partition function.

Fortunately, methods exist to perform inference even when the normalization constant is not known. Maximum a posteriori inference (which is equivalent to energy minimization) attempts to find the mode of the distribution. Our goal here is to reason about the entire *distribution* of shapes instead of finding the single most likely shape. As such, we turn to a Metropolis-Hastings MCMC (MH-MCMC) framework.

MH-MCMC only requires one to know the distribution of interest up to a proportionality (see Section 2.5.1). This property results from the evaluation of the Hastings ratio which precludes the need to calculate the partition function. In the following sections, we will often denote the posterior distribution as being proportional to $p(z)p(x|z)$ with the implied understanding that $p^*(z)$ can be substituted for $p(z)$ in MH-MCMC.

■ 3.2.2 Augmented Ordering Sample Space

The space of possible labelings is very large (2^N, where N is the number of pixels). One might expect that augmenting the model with additional random variables would only further complicate inference; however, to the contrary, we will show that a particular choice of auxiliary variables based on orderings of the pixels can significantly simplify the configureation space. A similar type of augmentation was used in the Dirichlet process mixture models sampling work of [79].

We begin the discussion with some definitions related to orderings of a set. Assume that each pixel, i, of the N total pixels is assigned an *order-index*, denoted by o_i, that takes on an integer value in $\{1, \ldots, N\}$.

Definition 3.2.1 (Total Ordering)**.** Let o_i be the order index assigned to pixel i. If the vector o contains the set of integers 1 to N in any order, then it is defined to be a total ordering. It can be shown that the set of all total orderings of size N can be placed in correspondence with the permutation group over N elements. Additionally, if $o_i < o_j$, we say that pixel i precedes pixel j in the ordering.

For example, when $N = 3$, the ordering $o = [1, 2, 2]$ would not be a total ordering because $o_2 = o_3$. The set of all total orderings has cardinality $N!$, and for $N = 3$ is the following:

$$\{[1, 2, 3], [1, 3, 2], [2, 1, 3], [2, 3, 1], [3, 1, 2], [3, 2, 1]\}.$$

The proposed algorithm will only consider changing a subset of pixels at once. As such, it will be convenient to define the *relative ordering* implied in a subset of pixels and their corresponding order-indices.

i:	1	2	3	4
o_i:	3	2	1	4
r_i:	–	2	1	3

Relative Ordering

i:	1	2	3	4
o_i:	3	2	1	4
r_i:	–	2	1	4

Not Relative Ordering

i:	1	2	3	4
o_i:	3	2	1	4
r_i:	–	1	2	3

Not Relative Ordering

Figure 3.2: Positive and negative examples of relative orderings with $\mathcal{I} = \{2, 3, 4\}$. '–' is used to denote a pixel that does not belong to $\mathcal{I}$. The second case is not a relative ordering because r is not a total ordering of size $|\mathcal{I}|$ since $r_4 = 4 > 3$. The third case is not a relative ordering because it does not preserve the implied order of o; pixel 3 precedes pixel 2 according to o, but the same does not hold according to r.

Definition 3.2.2 (Relative Ordering)**.** Let o define a total ordering on the pixels, and $\mathcal{I}$ be a subset of all the pixels. A *relative* ordering, r, of the pixels in $\mathcal{I}$ is defined to be the total ordering of size $|\mathcal{I}|$ that orders the pixels in $\mathcal{I}$ implied by o. A relative ordering satisfies the following condition:

$$r_i < r_j \iff o_i < o_j, \quad \forall i, j \in \mathcal{I}.$$

That is, pixel i precedes pixel j in the relative order iff pixel i precedes pixel j in the total order o.

We note that there is a single unique relative ordering given any total order and subset of indices. Examples of positive and negative relative orderings are shown in Figure 3.2.

Each pixel i is also associated with a discrete label, z_i, that takes on values in $\{1, \ldots, K\}$ in the K-ary case, or $\{0, 1\}$ in the binary case. We now define a *consistent* ordering, which is a property that relates orderings, o with labels, z.

Definition 3.2.3 (Consistent Ordering)**.** Let o define a total ordering on the pixels and z define the labeling of the pixels. The ordering, o, is *consistent* with the labeling, z, if for all k, the set of pixels with label k (i.e., $\{i; z_i = k\}$) have contiguous order-indices. An ordering that is consistent with a labeling must satisfy the following condition:

$$z_i = k, \ \forall i \in \{i; o_i \in \{a_k, \ldots, b_k\}\},$$

where $a_k = \min_{\{i; z_i = k\}} o_i$ and $b_k = \max_{\{i; z_i = k\}} o_i$.

We note that unlike relative orderings, there generally exists multiple consistent total orderings given a set of labels. Examples of positive and negative consistent orderings are shown in Figure 3.3.

We note that, in general, a relative ordering need not be consistent with a labeling. However, if the total ordering on pixels is consistent, all derived relative ordering must also be consistent. We now define the joint distribution over the augmented space of labels and orderings, denoted by $p(z, o) = p(z)p(o|z)$. We note that by definition, any

i:	1	2	3	4
z_i:	0	0	0	1
o_i:	3	2	1	4

Consistent Ordering

i:	1	2	3	4
z_i:	1	0	0	1
o_i:	3	2	1	4

Consistent Ordering

i:	1	2	3	4
z_i:	0	0	1	1
o_i:	3	2	1	4

Not Consistent Ordering

Figure 3.3: Positive and negative examples of consistent orderings in the binary case. The third case is not a consistent ordering since the order-indices for $k = 1$ are $\{1, 4\}$, which are not a set of contiguous numbers.

valid conditional distribution of orderings preserves the correct marginal distribution of $p(z)$. For simplicity, we choose the conditional distribution to be uniform over all consistent orderings.

We now focus our discussion on the binary case, where $z_i \in \{0, 1\}$. Denoting N_k as the number of pixels with label k, the conditional distribution can be expressed as

$$p(o|z) = \frac{1}{2N_0!N_1!}, \tag{3.3}$$

where the dependence on z is implied by the counts, N_0 and N_1. We note that the factor of 2 in the denominator arises due to the ambiguity in permuting the labels. That is, all the pixels with label 0 can either come before or after the pixels with label 1. Furthermore, all orderings that are not consistent with the given labeling, z, will have zero probability by construction. The resulting joint distribution is then

$$p(z, o) = p(z)p(o|z) = \frac{p(z)}{2N_0!N_1!}. \tag{3.4}$$

■ 3.2.3 Metropolis-Hastings in Augmented Space

We formulate a Metropolis-Hastings Markov chain Monte Carlo algorithm to sample from Equation (3.4). Conditioned on the previous values, denoted $z^{(t)}$ and $o^{(t)}$, we sample values from a user-specified proposal distribution, $q(\hat{z}, \hat{o}|z^{(t)}, o^{(t)})$. These values are then accepted or rejected with probability:

$$\begin{aligned}
\Pr\left[\{z^{(t+1)}, o^{(t+1)}\} = \{\hat{z}, \hat{o}\}\right] &= \min\left[1, \overbrace{\frac{p(\hat{z}, \hat{o})p(x|\hat{z})}{p(z^{(t)}, o^{(t)})p(x|z^{(t)})} \cdot \frac{q(z^{(t)}, o^{(t)}|\hat{z}, \hat{o})}{q(\hat{z}, \hat{o}|z^{(t)}, o^{(t)})}}^{H}\right], \\
\Pr\left[\{z^{(t+1)}, o^{(t+1)}\} = \{z^{(t)}, o^{(t)}\}\right] &= 1 - \Pr\left[\{z^{(t+1)}, o^{(t+1)}\} = \{\hat{z}, \hat{o}\}\right]
\end{aligned} \tag{3.5}$$

The ratio within the minimization is generally referred to as the Hastings Ratio, which we denote as H. The particular proposal distribution, which is chosen for efficiency, is described in Algorithm 3.3. The proposal distribution over subsets of pixels, $q(\mathcal{I})$, is typically chosen to sample a circle with a random center and a radius drawn uniformly

Algorithm 3.3 PGIMH Proposal Distribution

1. Sample a subset of pixels, $\mathcal{I} \sim q(\mathcal{I})$.
2. Sample a new set of labels, $\hat{z}_\mathcal{I} \sim q(\hat{z}_\mathcal{I}|z^{(t)}, o^{(t)}, x)$, that possibly changes labels within $\mathcal{I}$ while preserving the consistency of the *relative* ordering.
3. Sample a new consistent total ordering, $\hat{o} \sim q(\hat{o}|\mathcal{I}, \hat{z}, z^{(t)}, o^{(t)})$, uniformly from all orderings that preserve the relative ordering of the pixels in $\mathcal{I}$, and is consistent with the proposed label.

from a fixed set of sizes. Consequently, Step 1 can be computed in constant time. It is easily seen that there exist $|\mathcal{I}|+1$ possible configurations of the pixels in $\mathcal{I}$ that preserve the consistency of the relative ordering, which can be found by sorting the ordered pixels in $\mathcal{I}$. Step 2 can consequently be computed in $O(|\mathcal{I}| \log |\mathcal{I}|)$ for the sorting, and $O(|\mathcal{I}|)$ for the enumeration of the configurations. Finally, Step 3 requires one to sample a new valid consistent order for all pixels, which has complexity $O(N)$, where $N \gg |\mathcal{I}|$. This naïve implementation exhibits $O(|\mathcal{I}| \log |\mathcal{I}|+N)$ complexity, which is clearly undesirable since many iterations of Algorithm 3.3 are needed. After proving the validity of this proposal, we show how this complexity can be reduced to $O(|\mathcal{I}|)$.

Validity of Sampling Algorithm

As with any MH-MCMC algorithm, we must show that the resulting Markov chain is ergodic and that the Hastings ratio can be computed in order to preserve detailed balance. As stated in Section 2.5.1, ergodicity is often difficult to prove, and we instead show that the chain is irreducible. We remind the reader that irreducibility is essentially equivalent to ensuring that every state can reach every other state. If $q(\mathcal{I})$ includes single pixel regions, then irreducibility is trivial, since every pixel has non-zero probability of changing labels at any particular iteration in the proposal.

Next, we show that the Hastings ratio can be calculated for the specific proposal distribution described in Algorithm 3.3. The ratio of target distributions in H of Equation (3.4) is simply calculated based on the pre-specified distributions:

$$\begin{aligned}\frac{p(\hat{z},\hat{o})p(x|\hat{z})}{p(z^{(t)},o^{(t)})p(x|z^{(t)})} &= \frac{p(\hat{z})p(x|\hat{z})}{p(z^{(t)})p(x|z^{(t)})} \cdot \frac{p(\hat{o}|\hat{z})}{p(o^{(t)}|z^{(t)})} \\ &= \frac{p(\hat{z})p(x|\hat{z})}{p(z^{(t)})p(x|z^{(t)})} \cdot \frac{N_0!N_1!}{\hat{N}_0!\hat{N}_1!}. \end{aligned} \tag{3.6}$$

We now focus on the ratio of proposal distributions. We note that the number of permutations with a subset of indices following a specific order is $\frac{N!}{N_\mathcal{I}!}$, where N is the length of the full permutation, and $N_\mathcal{I} = |\mathcal{I}|$ is the size of the ordered subset. The

probability of generating from the proposal in Algorithm 3.3 can then be expressed as

$$\begin{aligned} q(\hat{z}, \hat{o}|z^{(t)}, o^{(t)}) &= q(\mathcal{I})q(\hat{z}_{\mathcal{I}}|z^{(t)}, o^{(t)}, x) \cdot q(\hat{o}|\mathcal{I}, \hat{z}, z^{(t)}, o^{(t)}), \\ &= q(\mathcal{I})q(\hat{z}_{\mathcal{I}}|z^{(t)}, o^{(t)}, x) \cdot \left[\frac{\hat{N}_{\mathcal{I}_0}!}{\hat{N}_0!} \cdot \frac{\hat{N}_{\mathcal{I}_1}!}{\hat{N}_1!}\right], \end{aligned} \tag{3.7}$$

where $\hat{N}_k$ is the number of pixels with label $\hat{z} = k$, $\mathcal{I}_k$ are the subset of pixels within $\mathcal{I}$ that have label k, and $\hat{N}_{\mathcal{I}_k} = |\hat{\mathcal{I}}_k|$. Because the proposal distribution preserves the relative ordering of pixels in $\mathcal{I}$, the reverse move, $q(z^{(t)}, o^{(t)}|\hat{z}, \hat{o})$ is always possible, and only depends on the counts associated with the subset of pixels. This likelihood can therefore be expressed as

$$\begin{aligned} q(z^{(t)}, o^{(t)}|\hat{z}, \hat{o}) &= q(\mathcal{I})q(z^{(t)}_{\mathcal{I}}|\hat{z}, \hat{o}, x) \cdot q(o^{(t)}|\mathcal{I}, z^{(t)}, \hat{z}, \hat{o}), \\ &= q(\mathcal{I})q(z^{(t)}_{\mathcal{I}}|\hat{z}, \hat{o}, x) \cdot \left[\frac{N_{\mathcal{I}_0}!}{N_0!} \cdot \frac{N_{\mathcal{I}_1}!}{N_1!}\right]. \end{aligned} \tag{3.8}$$

The expressions in Equations (3.6)–(3.8) combine to form the following Hastings ratio

$$\begin{aligned} H &= \frac{p(\hat{z})p(x|\hat{z})}{p(z^{(t)})p(x|z^{(t)})} \cdot \frac{N_0!N_1!}{\hat{N}_0!\hat{N}_1!} \cdot \frac{q(\mathcal{I})q(z^{(t)}_{\mathcal{I}}|\hat{z}, \hat{o}, x) \cdot \frac{N_{\mathcal{I}_0}!}{N_0!} \cdot \frac{N_{\mathcal{I}_1}!}{N_1!}}{q(\mathcal{I})q(\hat{z_{\mathcal{I}}}|z^{(t)}, o^{(t)}, x) \cdot \frac{\hat{N}_{\mathcal{I}_0}!}{\hat{N}_0!} \cdot \frac{\hat{N}_{\mathcal{I}_1}!}{\hat{N}_1!}} \\ &= \frac{p(\hat{z})p(x|\hat{z})}{p(z^{(t)})p(x|z^{(t)})} \cdot \frac{q(z^{(t)}_{\mathcal{I}}|\hat{z}, \hat{o}, x) \cdot N_{\mathcal{I}_0}!N_{\mathcal{I}_1}!}{q(\hat{z_{\mathcal{I}}}|z^{(t)}, o^{(t)}, x) \cdot \hat{N}_{\mathcal{I}_0}!\hat{N}_{\mathcal{I}_1}!} \end{aligned} \tag{3.9}$$

A Gibbs-Inspired Proposal

As shown in Section 2.5.1, the Gibbs sampler can be seen as a special case of Metropolis-Hastings where the acceptance ratio always evaluates to 1. This convenient property occurs because the proposal distribution is chosen to be the conditional posterior distribution, and is desirable because computation is not wasted on rejected samples. We follow in a similar fashion here. While the Hastings ratio in (3.9) results in a valid sampling algorithm for nearly any choice of $q(\hat{z}_{\mathcal{I}}|z^{(t)}, o^{(t)})$, if we choose the proposal to be

$$q(\hat{z}_{\mathcal{I}}|z^{(t)}, o^{(t)}) \propto p(\hat{z})p(x|\hat{z})\frac{1}{\hat{N}_{\mathcal{I}_0}!\hat{N}_{\mathcal{I}_1}!}, \tag{3.10}$$

the Hastings ratio evaluates to one, and every proposed sample is accepted. Moreover, because proposed labels must preserve the relative ordering of the pixels in $\mathcal{I}$, there exist only $N_{\mathcal{I}} + 1$ possible consistency-preserving configurations. Sampling from Equation (3.10) is efficient because enumerating all possible configurations only grows linearly in the size of $\mathcal{I}$. We call this sampling algorithm the "Permutation-based Gibbs-Inspired Metropolis-Hastings" (PGIMH).

■ 3.2.4 An Efficient Implementation

The previous section described an algorithm for correctly sampling from target distributions. However, for the algorithm to be useful, sampling from the proposal distribution and evaluating the Hastings ratio must be efficient. As stated previously, the computational complexity of the naïve implementation is $O(|\mathcal{I}| \log |\mathcal{I}| + N)$, which is quite large. In this section, we describe an exact implementation of this algorithm that reduces this complexity to $O(|\mathcal{I}|)$ per iteration.

As discussed in Section 2.5.1, any combination of valid proposal distributions can be mixed while still preserving the limiting convergence guarantees. We consider mixing the PGIMH sampling algorithm detailed previously with a Gibbs iteration that samples a consistent total ordering conditioned on the labels. While this may seem like it adds complexity to the model, it actually allows us to simplify the implementation. We draw on the following key observation: when a naïve PGIMH iteration is preceded by a Gibbs iteration that samples a consistent ordering, the relative ordering of the pixels within $\mathcal{I}$ will be uniformly distributed over all consistent relative orderings. Thus, this iterative procedure can be exactly reproduced with the method described in Algorithm 3.4. We

Algorithm 3.4 An iteration of sampling $p(z)$ via PGIMH

1. Sample a subset of pixels, $\mathcal{I} \sim q(\mathcal{I})$.
2. Sample a consistent relative ordering of pixels in $\mathcal{I}$ uniformly using a Knuth shuffle [71] on each $\mathcal{I}_0$ and $\mathcal{I}_1$.
3. Enumerate the $N_{\mathcal{I}}$ consistency-preserving configurations.
4. Sample a $z_{\mathcal{I}}^{(t+1)} \sim p(\hat{z})p(x|\hat{z})\frac{1}{\hat{N}_{\mathcal{I}_0}!\hat{N}_{\mathcal{I}_1}!}$ from the set of possible moves.

note that because a random relative ordering is sampled at each iteration of PGIMH, the explicit ordering, o, does not actually need to be maintained. Additionally, since the Knuth shuffle [71] can be performed in $O(|\mathcal{I}|)$ time, and the total ordering does not need to be updated, the overall complexity of an iteration is now $O(|\mathcal{I}|)$. In practice, $\mathcal{I} \sim q(\mathcal{I})$ is sampled by: (1) sampling a random circle center followed by (2) sampling a random radius around it (from some pre-specified range of valid radii). Figure 3.4 illustrates an example proposal from PGIMH.

It is interesting to note that this relationship holds for any data-independent proposal distribution of the subset, $\mathcal{I}$. In particular, if $\mathcal{I}$ is chosen to be a single random pixel, PGIMH simplifies to the typical Gibbs sampler since the denominator of Equation (3.10) evaluates to $0! \cdot 1! = 1$. Thus, PGIMH is essentially a generalization of Gibbs sampling that allows larger moves like blocked Gibbs sampling but without the computational drawbacks.

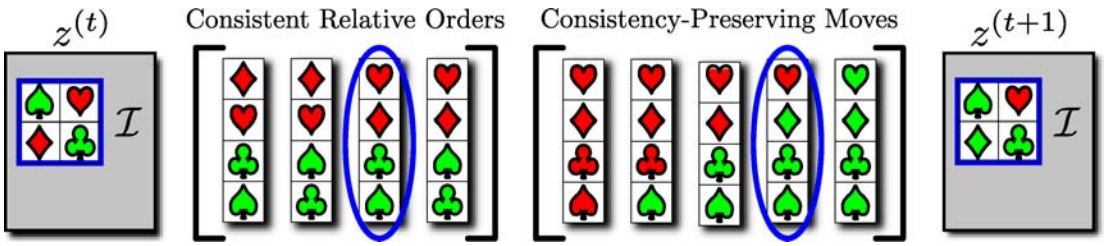


Figure 3.4: An example of the PGIMH proposals. A random subset is chosen (left), $\mathcal{I} = \{\spadesuit, \diamondsuit, \heartsuit, \clubsuit\}$ where suits indicate indices and colors indicate labels. A consistent relative order is selected at random. One of the $N_{\mathcal{I}} + 1$ possible consistency-preserving configurations is selected to produce $z^{(t+1)}$.

■ 3.3 K-Ary Sampling

The above algorithm is described for partitioning the image into two distinct regions. However, in practice, binary segmentations are of limited interest because scenes typically contain multiple objects. We now show how one can extend PGIMH to allow for K regions.

There has been significant work in extending the level-set framework to multiple objects for image segmentation (e.g., [14, 121], which both require added complexity and memory consumption). The sampling methods of [23, 31] do not address multiple regions. However, as we soon discuss, the PGIMH algorithm extends to multiple regions straightforwardly.

Assume there are K labels of interest and that the same definition of consistent ordering in Definition 3.2.3 applies to multiple labels. A multi-label proposal follows similarly to Algorithm 3.4 after selecting a pair of labels, k and ℓ. We summarize the steps in Algorithm 3.5 and follow by proving its validity.

Algorithm 3.5 An iteration of sampling $p(z)$ for multiple labels via PGIMH

1. Sample two random labels, k and ℓ, uniformly.
2. Sample a subset of pixels, $\mathcal{I} \sim q(\mathcal{I})$ that only contains pixels with label k or ℓ.
3. Sample a consistent relative ordering of pixels in $\mathcal{I}$ uniformly using a Knuth shuffle [71] on each $\mathcal{I}_0$ and $\mathcal{I}_1$.
4. Enumerate the $N_{\mathcal{I}} + 1$ consistency-preserving configurations that only changes pixels to labels k or ℓ.
5. Sample a $z_{\mathcal{I}}^{(t+1)} \sim p(\hat{z})p(x|\hat{z})\frac{1}{\hat{N}_{\mathcal{I}_0}!\hat{N}_{\mathcal{I}_1}!}$ from the set of possible moves.

Validity of K-Ary Sampling Algorithm

The chain for K labels is irreducible by the same argument as the binary case. We therefore only show how the Hastings ratio can be calculated and that Algorithm 3.5

results in a Hastings ratio that always accepts the samples. The ratio of joint distributions in H of Equation (3.4) is calculated in a similar fashion as the binary case. The only change is in the conditional distribution on orderings. Assuming that labels k and l change, this results in

$$\begin{aligned}\frac{p(\hat{z},\hat{o})p(x|\hat{z})}{p(z^{(t)},o^{(t)})p(x|z^{(t)})} &= \frac{p(\hat{z})p(x|\hat{z})}{p(z^{(t)})p(x|z^{(t)})}\cdot\frac{p(\hat{o}|\hat{z})}{p(o^{(t)}|z^{(t)})}\\ &= \frac{p(\hat{z})p(x|\hat{z})}{p(z^{(t)})p(x|z^{(t)})}\cdot\frac{N_k!N_l!}{\hat{N}_k!\hat{N}_l!}\end{aligned} \tag{3.11}$$

We now focus on the ratio of proposal distributions. The probability of generating from the proposal in Algorithm 3.5 can then be expressed as

$$\begin{aligned}q(\hat{z},\hat{o}|z^{(t)},o^{(t)}) &= q(k,\ell)q(\mathcal{I})q(\hat{z}_{\mathcal{I}}|z^{(t)},o^{(t)},x)\cdot q(\hat{o}|\mathcal{I},\hat{z},z^{(t)},o^{(t)}),\\ &= \frac{2}{K(K-1)}q(\mathcal{I})q(\hat{z}_{\mathcal{I}}|z^{(t)},o^{(t)},x)\cdot\left[\frac{\hat{N}_{\mathcal{I}_k}!}{\hat{N}_k!}\cdot\frac{\hat{N}_{\mathcal{I}_l}!}{\hat{N}_l!}\right].\end{aligned} \tag{3.12}$$

The likelihood of the reverse move can similarly be expressed as

$$\begin{aligned}q(z^{(t)},o^{(t)}|\hat{z},\hat{o}) &= q(k,\ell)q(\mathcal{I})q(z^{(t)}_{\mathcal{I}}|\hat{z},\hat{o},x)\cdot q(o^{(t)}|\mathcal{I},z^{(t)},\hat{z},\hat{o}),\\ &= \frac{2}{K(K-1)}q(\mathcal{I})q(z^{(t)}_{\mathcal{I}}|\hat{z},\hat{o},x)\cdot\left[\frac{N_{\mathcal{I}_k}!}{N_k!}\cdot\frac{N_{\mathcal{I}_l}!}{N_l!}\right].\end{aligned} \tag{3.13}$$

The expressions in Equations (3.11)–(3.13) combine to form the following Hastings ratio

$$\begin{aligned}H &= \frac{p(\hat{z})p(x|\hat{z})}{p(z^{(t)})p(x|z^{(t)})}\cdot\frac{N_k!N_l!}{\hat{N}_k!\hat{N}_l!}\cdot\frac{\frac{2}{K(K-1)}}{\frac{2}{K(K-1)}}\cdot\frac{q(\mathcal{I})q(z^{(t)}_{\mathcal{I}}|\hat{z},\hat{o},x)\cdot\frac{N_{\mathcal{I}_k}!}{N_k!}\cdot\frac{N_{\mathcal{I}_l}!}{N_l!}}{q(\mathcal{I})q(\hat{z_{\mathcal{I}}}|z^{(t)},o^{(t)},x)\cdot\frac{\hat{N}_{\mathcal{I}_k}!}{\hat{N}_k!}\cdot\frac{\hat{N}_{\mathcal{I}_l}!}{\hat{N}_l!}}\\ &= \frac{p(\hat{z})p(x|\hat{z})}{p(z^{(t)})p(x|z^{(t)})}\cdot\frac{q(z^{(t)}_{\mathcal{I}}|\hat{z},\hat{o},x)\cdot N_{\mathcal{I}_k}!N_{\mathcal{I}_l}!}{q(\hat{z_{\mathcal{I}}}|z^{(t)},o^{(t)},x)\cdot\hat{N}_{\mathcal{I}_k}!\hat{N}_{\mathcal{I}_l}!}.\end{aligned} \tag{3.14}$$

If we use the proposal for label moves of Step 5 in Algorithm 3.5, this results in

$$H = \frac{p(\hat{z})p(x|\hat{z})}{p(z^{(t)})p(x|z^{(t)})}\cdot\frac{p(z^{(t)})p(x|z^{(t)})\frac{1}{N_{\mathcal{I}_k}!N_{\mathcal{I}_l}!}\cdot N_{\mathcal{I}_k}!N_{\mathcal{I}_l}!}{p(\hat{z})p(x|\hat{z})\frac{1}{\hat{N}_{\mathcal{I}_k}!\hat{N}_{\mathcal{I}_l}!}\cdot\hat{N}_{\mathcal{I}_k}!\hat{N}_{\mathcal{I}_l}!} = 1, \tag{3.15}$$

which means that every proposed sample is accepted. This proves the validity of the multi-label sampling algorithm.

We note that the sampling of a mask must be slightly altered from the binary case. Sampling a random circle will, in general, contain pixels with labels other than k or l. Consequently, we change this proposal to conform to Algorithm 3.5 by further selecting

only the pixels in the circular mask that currently have labels k or l.

■ 3.4 Compatible Priors

The preceding sections develop a sampling algorithm that makes large moves, scales well, and maintains properties that yield a valid MCMC sampling procedure. In this section, we give some examples of priors that can be used on the labels, $p(z)$, that fit within the PGIMH framework. We cover the commonly used curve length penalty and balloon force in image segmentation. Additionally we develop a method for controlling the topology of the shape. Finally, we discuss the required properties of a prior to fit in the sampling framework.

■ 3.4.1 Priors on Curve Length

One common application for using implicitly defined shapes is image segmentation, where a prior is placed on the shape that penalizes the curve length. Unlike many MRF-based priors that are explicitly parametrized over neighbors, computing curve length can be non-local and fairly expensive. In optimization-based segmentation algorithms, the evolution of the level-set is based on the gradient of the curve length (i.e., the curvature), which is a local and efficient computation.

A *curve length penalty* is a term often used in optimization-based image segmentation algorithms, since the penalty is placed directly on the energy functional that is minimized. Typical energy functionals can be decomposed into separate data and regularization terms:

$$E(z;x) = E_{\text{data}}(x,z) + E_{\text{regularization}}(z). \tag{3.16}$$

The curve-length penalty is incorporated directly in the regularization term

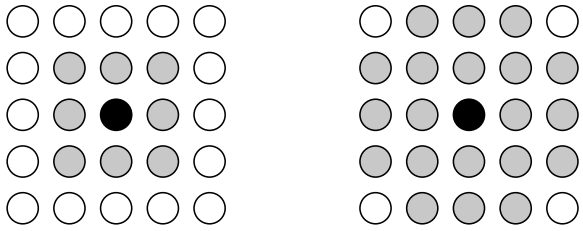
$$E_{\text{regularization}}(z) = \alpha \oint_C dl, \tag{3.17}$$

where α is a weighting parameter and C denotes the boundary of the segmentation. The energy functional also has a direct interpretation in the Bayesian setting when exponentiated. That is, we can choose the joint distribution to be related to the energy as follows

$$p(x,z) \propto e^{-E(z;x)} = \underbrace{e^{-E_{\text{data}}(x,z)}}_{\approx p(x|z)} \cdot \underbrace{e^{-\alpha \oint_C dl}}_{\approx p(z)}. \tag{3.18}$$

We note that we use the approximation symbol since the energy functional is defined over a continuous function, whereas the distribution of interest is defined over a discrete set of random variables, z. Additionally, the partition function is not typically known.

We now present an efficiently computed approximation to the non-local curve length. We denote Ω as the image domain, and $\delta(\varphi)$ as the derivative to the Heaviside function. With these definitions, we approximate the curve length by setting all pixels on the



(a) (b)

Figure 3.5: Local neighborhood dependence for computing curve-length. (a) Signed distance function dependence and (b) curve length penalty dependence. The black-colored pixel depends on the gray-colored pixels.

boundary to have an equivalent level-set height of ± 0.5, initializing a signed-distance function, and then calculating the curve length via

$$\oint_{\mathcal{C}} dl = \int_{\Omega} \delta(\varphi_i) \left|\nabla \varphi_i\right| di \approx \sum_i \hat{\delta}(\varphi_i) \left|\nabla \varphi_i\right|, \tag{3.19}$$

where the following discrete approximation to the $\delta(\cdot)$ function is used

$$\hat{\delta}(\varphi_i) = \begin{cases} 1 & |\varphi_i| \leq 1 \\ 0 & \text{else} \end{cases}. \tag{3.20}$$

We note that every time a single pixel changes labels, the resulting signed-distance function, φ, must be updated by solving the Eikonal equation (see Section 2.6). This computation is quite cumbersome and undesirable for local changes.

If a first-order approximation to the signed-distance function is obtained using a fast marching method [119], the height at a particular pixel depends only on pixels in a 3x3 neighborhood. This dependence relationship is illustrated in Figure 3.5a, where changing the sign of the center black pixel will affect the height of the signed distance function at all the gray pixels. From Equation (3.19), the local curve length calculation depends on the magnitude of the gradient of the level-set function. If a centered finite-difference is used for the x and y directions, then the local curve length computation depends on neighbors above, below, left, and right. Thus, if the heights are changed for the gray pixels in Figure 3.5a, the curve length computation changes for all the gray pixels in Figure 3.5b. The change in the curve length for changing the sign of the center pixel is consequently a function of the sign of the 21 pixels in Figure 3.5b. We precompute all possible 2^{21} (≈ 2 million) combinations so that the curve length penalty can be efficiently computed by a simple table lookup. We note that this approximation does not scale well to higher dimensions. However, if a curve-length penalty prior is not needed, the remaining framework of PGIMH extends straightforwardly to arbitrary dimensions.

■ 3.4.2 Priors on Balloon Force

The curve length penalty favors shrinking regions down to a point. The balloon force was introduced as a term to complement the bias towards small regions. Cohen and Cohen [26] show that the balloon force has the following corresponding regularization energy

$$E_{\text{regularization}}(z) = -\alpha \int_{\Omega} dA, \tag{3.21}$$

which, when minimized, tries to maximize the area of the region (assuming $\alpha > 0$).

In the Bayesian framework, exponentiating the energy functional and translating the continuous function to the discrete grid results in the following balloon force prior

$$p(z) \propto e^{\alpha \sum_i \mathbb{1}[z_i = k]}, \tag{3.22}$$

where we have assumed that the balloon force prior has been placed on region k. We note that if a balloon force is placed on all K possible regions, all with equal weights, α, the resulting prior favors partitions with equal areas.

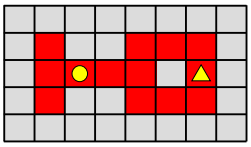
■ 3.4.3 Priors on Topology

The previous two priors were ways to regularize the smoothness of the shape. Alternatively, one may have prior knowledge of the *topology* of the shape. Restricting the topology of the region should not be confused with the fact that implicit shapes handle topology changes automatically. Rather, we exploit this property of implicit shapes to specifically allow or disallow certain topologies. This goal is similar to the works of [50, 104], which altered the level-set velocity to preclude restricted topologies.

We have reviewed some digital topology concepts in Section 2.7. We briefly describe some of our contributions to this line of research, followed by how constraints can be incorporated into a sampling algorithm. We note that digital topology is only well-defined for binary segmentations of a single foreground from the background.

As stated in Section 2.7, the topology numbers, T_n and $T_{\overline{n}}$ of [6], and the extended topology numbers, T_n^+ and $T_{\overline{n}}^+$ of [104], uniquely identify any topology changes that may occur if a pixel is moved from one region to another. Unfortunately, Segonne [104] discovered that the required extended topology numbers cannot be efficiently computed for 3D shapes. In particular, T_n^+ cannot be computed efficiently when a pixel is moved from the foreground to the background, and $T_{\overline{n}}^+$ cannot be computed efficiently when a pixel is moved from the background to the foreground. We now show that these particular situations are not needed in 2D.

Consider the two pixels marked by $\circ$ and $\triangle$ in Figure 3.6. Removing the $\circ$ pixel from the foreground splits the region and removing the $\triangle$ pixel destroys a handle. We emphasize that in this 2D case, both $T_n^+(\circ) \neq T_n^+(\triangle)$ *and* $T_{\overline{n}}^+(\circ) \neq T_{\overline{n}}^+(\triangle)$. In 3D, this is generally not the case because the two background regions bordering the $\triangle$ pixel can actually be connected in another 2D slice of the volume. In fact, in 2D, the destruction of a handle in the foreground corresponds directly to a merging of regions in



$$T_n(\circ) = 2 \quad = \quad T_n(\triangle) = 2$$
$$T_{\overline{n}}(\circ) = 1 \quad = \quad T_{\overline{n}}(\triangle) = 2$$
$$T_n^+(\circ) = 2 \quad \neq \quad T_n^+(\triangle) = 1$$
$$T_{\overline{n}}^+(\circ) = 1 \quad \neq \quad T_{\overline{n}}^+(\triangle) = 2$$

Figure 3.6: Splitting a region vs. destroying a handle when a pixel is added to BG. Topological numbers are shown on right.

Table 3.1: Topological changes as a function of topological numbers.

				Pixel Added to FG		Pixel Added to BG	
T_n	T_n^+	$T_{\overline{n}}$	$T_{\overline{n}}^+$	FG	BG	FG	BG
0	0	1	1	CR	CH	DR	DH
1	1	0	0	DH	DR	CH	CR
1	1	1	1	-	-	-	-
≥ 2	$< T_n$	X	X	CH	SR	X	X
≥ 2	≥ 2	X	X	MR	DH	X	X
X	X	≥ 2	$< T_{\overline{n}}$	X	X	SR	CH
X	X	≥ 2	≥ 2	X	X	DH	MR

'C' – Create 'D' – Destroy 'S' – Split 'M' – Merge
'H' – Handle(s) 'R' – Region(s) 'X' – any value '-' – no topological change
Omitted configurations are impossible in 2D.

the background. Likewise, the splitting of regions in the foreground corresponds directly to a creation of a handle in the background. This one-to-one mapping precludes the need to compute the expensive $T_{\overline{n}}^+$ when adding a pixel to the foreground. Table 3.1 summarizes the topological changes of the foreground and background in 2D as a function of the four topological numbers.

A naïve approach could restrict the topology of the shape by generating proposals using PGIMH and rejecting samples that violate topology constraints. Such an approach wastes significant computation generating samples that are rejected due to their topology. We take a different approach: only generate proposed samples from the set of allowable topological changes. Recalling the discussion of enumerating the possible moves in PGIMH, one can simply determine which moves correspond to allowable topologies and which do not. Moves corresponding to restricted topologies have their likelihood set to zero when topology control is desired. This methodology treats the topology as a hard constraint; however a distribution over topologies could be implemented by weighting moves based on topology changes rather than completely eliminating restricted ones.

3.4.4 Other Priors

There are a wealth of other priors that can easily be used in the PGIMH algorithm. For example, the following slight modification to the balloon force places a Gaussian prior on the actual size of a shape

$$p(z) \propto \exp\left[-\alpha\left(A - \sum\nolimits_i \mathbb{I}[z_i = k]\right)^2\right], \tag{3.23}$$

where A is the desired area.

We note that any of the aforementioned priors can be combined to form a new prior. Additionally, any prior that can be efficiently computed from local changes fits in the PGIMH framework. This even applies to some global priors, such as the prior on size, which can be efficiently computed from local changes by maintaining a summary statistic that captures the current area of the shape.

3.5 Mutual Information Energy Functional

As shown previously, most energy functionals can be exponentiated and treated as being proportional to the posterior distribution. Therefore, PGIMH is a general method with application to a variety of commonly used energy functionals over implicit representations. We choose a specific form for the remainder of this chapter. Unless otherwise stated, we use the energy functional of [69]. This approach iteratively estimates a nonparametric kernel density estimate over pixel intensities and maximizes the mutual information between the labels and pixel intensities subject to a curve length penalty. By denoting X and Z as the random variables that each x_i and z_i are realized from, the resulting energy can be expressed as:

$$E(x, z) = N \cdot I(X; Z) - \alpha \oint_C d\ell. \tag{3.24}$$

We choose this particular formulation because of the intricate relationship between mutual information and posterior distribution. The following shows that exponentiating the energy is equivalent to the posterior distribution assuming that pixels are i.i.d. conditioned on the label. We first note that since x is observed, adding a function that only depends on x does not change the function. We therefore consider the following equivalent energy:

$$E(x, z) = N \cdot (I(X; Z) - H(X)) - \alpha \oint_C d\ell = N \cdot H(X|Z) - \alpha \oint_C d\ell. \tag{3.25}$$

Exponentiating the energy results in

$$\begin{aligned}\exp\left[E(x,z)\right] &= \exp\left[N \cdot H(X|Z)\right] e^{-\alpha \oint_C d\ell} \\ &= \exp\left[N \cdot \sum_{k=1}^{K} \Pr(Z=k)\mathbb{E}[\log p(X|Z=k)]\right] e^{-\alpha \oint_C d\ell} \end{aligned} \tag{3.26}$$

Approximating $\Pr(Z=k) \approx \frac{N_k}{N}$ with the empirical counts and the expectation with sample realizations, we have

$$\begin{aligned}\exp\left[E(x,z)\right] &\approx \exp\left[N \sum_{k=1}^{K} \frac{N_k}{N} \frac{1}{N_k} \sum_{i \in \{i; z_i = k\}} \log p(x_i|z_i=k)\right] e^{-\alpha \oint_C d\ell} \\ &= \exp\left[\sum_{i=1}^{N} \log p(x_i|z_i)\right] e^{-\alpha \oint_C d\ell} = p(z) \prod_{i=1}^{N} p(x_i|z_i), \end{aligned} \tag{3.27}$$

which is equivalent to the joint distribution, $p(x,z)$, assuming that observations, x, are conditionally independent conditioned on z, and that the prior is $p(z) = e^{-\alpha \oint_C d\ell}$.

■ 3.6 Applications

In this section, we consider some examples in image segmentation using the proposed PGIMH sampling algorithm. As is typical in MCMC approaches, marginal statistics can be evaluated over samples using a simple counting measure. Similar to [31], one can compute the histogram image of a segmentation, where each pixel in the histogram contains a count of the number of times it was included in a particular region. Similarly, the 50% quantile curve corresponds to thresholding the histogram image at 0.5.

Here, we consider another marginal event probability: the probability that a pixel lies on the boundary. We refer to this as the probability of boundary image (PB). The PB at pixel i is calculated by simply counting the number of samples that pixel i lies on a boundary and normalizing by the number of samples. This statistic is of particular interest, as it allows one to evaluate results over the Berkeley Segmentation Dataset (BSDS) [88] that compares precision-recall (PR) curves on precisely this event probability. In this dataset, the maximum harmonic mean of points on the PR curve, or F-measure, is used as the metric for rating boundary detectors. Unlike boundary detectors, however, optimization-based segmentation algorithms produce a single point on the PR curve. Recent segmentation algorithms rarely report benchmark results on the BSDS due to poor F-measures owing to the inability to trade off between precision and recall. PGIMH enables these segmentation algorithms to produce a PB image for more robust comparison on the BSDS.

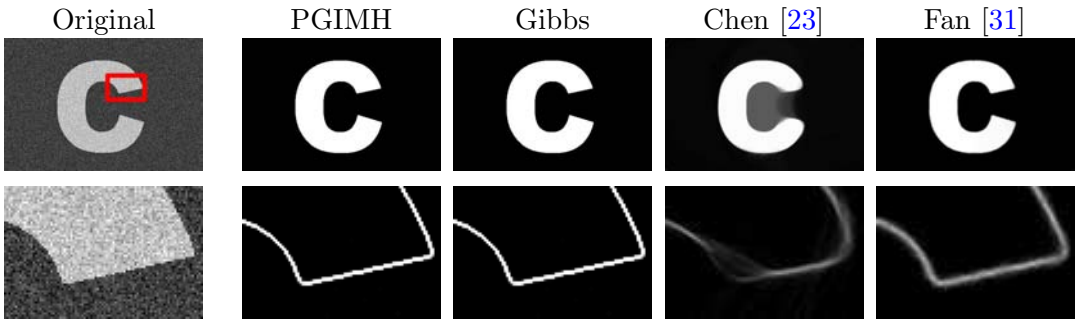


Figure 3.7: Comparison of shape sampling algorithms on a synthetic example. Each algorithm was run for 10^5 iterations. The top row shows the histogram images, and the bottom row shows a detail of the PB image.

■ 3.6.1 Convergence Times

We begin by comparing convergence between different sampling algorithms. We examine the computation times for four algorithms: PGIMH, Gibbs sampling, the method of Chen and Radke, [23], and the method of Fan et al. [31], noting that the Gibbs sampling algorithm can only be efficiently applied to image segmentation with curve length penalties because of the approximations we have developed in Section 3.4.

Consider the synthetic image of Figure 3.7 containing the letter 'C'. We run each algorithm for 10^5 iterations and compute summary statistics over 100 sample paths, which are shown in Figure 3.7. The histogram images imply that all algorithms, aside from Chen [23], have converged. Examination of a detail of the 'C' and the PB associated with each algorithm shows this not to be the case; it is clear that Chen [23] and Fan [31] have not converged. The results of Fan [31] have a blurred PB, and the results of Chen [23] are both blurred and miss corners.

The average log likelihood for each algorithm is shown versus computation time in Figure 3.8. We note that the ending time on this plot is slightly different than the full runs used to obtain the results of Figure 3.7 (because an iteration of each algorithm takes different amounts of time). This plot illustrates that PGIMH and Gibbs sampling (using our approximation of curve length) achieve at least 4 orders of magnitude in speed up as compared to the previous shape sampling algorithms. We fit a linear regression to the last 50 iterations of each algorithm to predict convergence times for the algorithms, noting that the assumption of linear growth is an optimistic *lower* bound, since the observed growth is sub-linear. Predicted convergence times are compared in Table 3.2.

We have empirically found that PGIMH and Gibbs are comparable in many experiments that are very likelihood dominated. However, Gibbs sampling suffers from convergence issues when many local extrema exist due to its local nature. For example, consider the image shown in Figure 3.9a, where the intensities in the background follow $\mathcal{N}(32, 100)$ and the foreground follow $\mathcal{N}(196, 100)$. A small subset of background pixels have additionally been incorrectly drawn from the foreground distribution. Typical

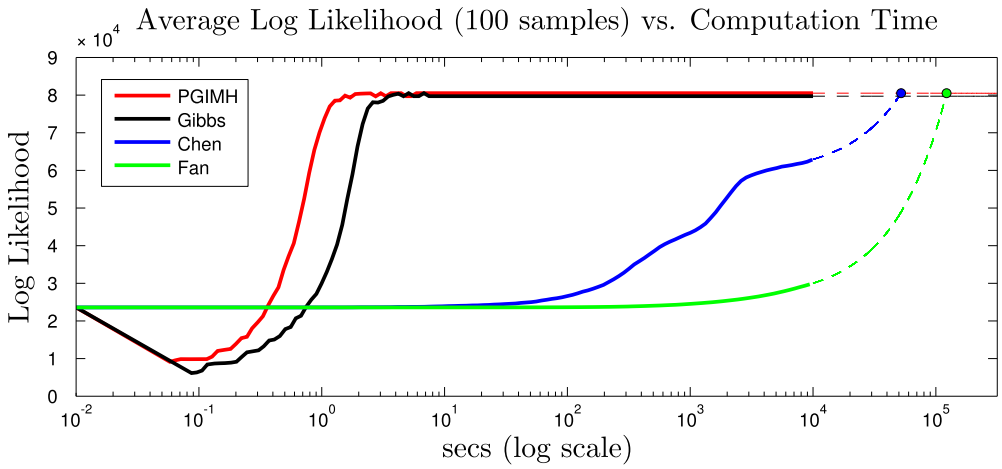
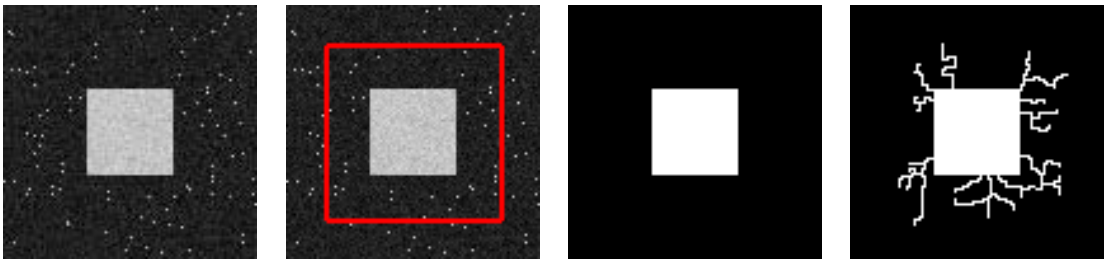


Figure 3.8: The average log likelihood across all sample paths vs. time (log scale) for multiple sampling algorithms. Dotted lines are estimated using linear regression from the last 50 iterations.

Table 3.2: Empirical Convergence Times for Shape Sampling.

	Convergence Time	Speed Gain
PGIMH	1.8 seconds	$\times 1$
Gibbs	3.4 seconds	$\times 2$
[23]	*52, 133.2 seconds	$\times 28,963$
[31]	*122, 273.7 seconds	$\times 67,930$

* denotes overly optimistic, estimated convergence times



(a) Original (b) Initialization (c) PGIMH (d) Gibbs

Figure 3.9: A problematic example for Gibbs sampling that converges to a local extremum. The observation distributions, $p(x_i|z_i)$ are assumed to be known. The total observation log likelihood , $\sum_i \log p(x_i|z_i)$ for PGIMH is -6.1×10^{-4} and while Gibbs is only -8.1×10^{-4}.

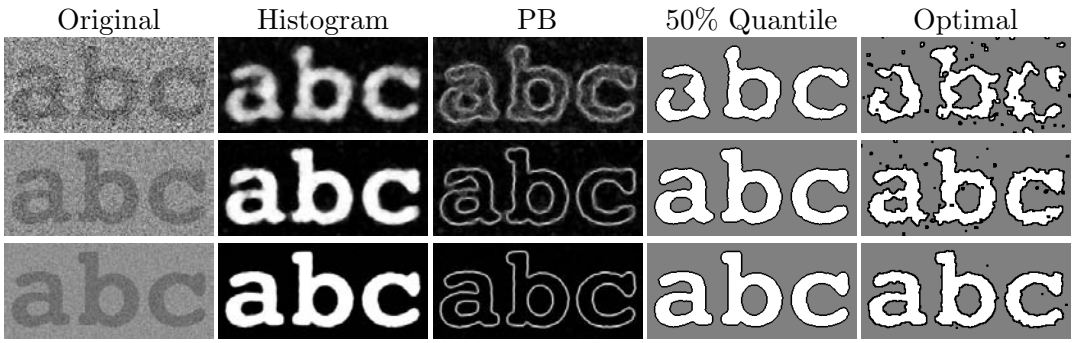


Figure 3.10: Results for three synthetic images with varying SNR values (0.5, 1.0, and 2.0, top to bottom, respectively). The columns show the original image, the histogram image, the probability of boundary image, the 50% quantile curve, and the "best" realized sample path (with the highest energy), respectively.

samples when the topology of the shape is not allowed to change are shown in Figures 3.9c-3.9d. Notice that the sample from Gibbs suffers from being stuck in a local extremum, whereas PGIMH converges to the correct solution. The errors in the Gibbs sampling output result from the following conditions. The corrupted pixels in the background look like the foreground, and are very unlikely to move to the background region. Consequently, neighboring pixels that are currently labeled foreground can also not move due to the topology restriction. While this example is admittedly quite contrived, it does highlight convergence issues that can occur from Gibbs sampling.

■ 3.6.2 Sensitivity to Noise

The previous results illustrate the computational advantages of PGIMH and its improved convergence properties as compared to other sampling algorithms. We now show results of using PGIMH in a few applications. Consider the synthetic images shown in Figure 3.10. Each image contains two regions that are drawn from Gaussian distributions with different means. We alter the variance to consider three different SNR values: 0.5, 1.0, and 2.0. The last column shows the sample path with the highest energy, which approximates the optimal configuration. In the lowest SNR case, the 50% quantile clearly produces much better results than the optimal sample path. As the SNR increases, the optimal sample path approaches the average sample path, but in low SNR scenarios, marginal event probabilities tend to be more robust than optimal configurations.

■ 3.6.3 Boundary Detection in Natural Images

As stated previously, marginal events such as the probability of boundaries are of interest. Due to their inherent topological constraints, [23] and [31] are less applicable to natural images where it is often desirable to group regions that are separated spatially

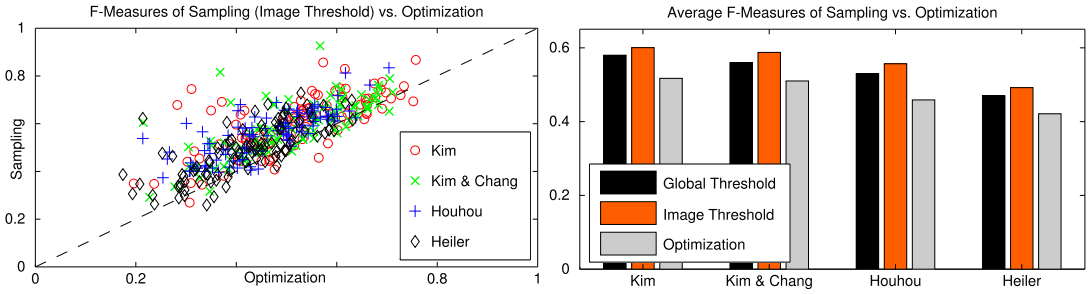


Figure 3.11: Sampling vs. optimization on the BSDS. The scatter plot shows the F-measures of each image using an image-based threshold on the PB image vs. the optimal segmentation. The bar plot shows the average F-measures using the global threshold on the PB image, the image-based threshold on the PB image, and the optimal segmentation.

and/or segment an image into more than two region labels. As such, the remaining results focus on the use of the K-ary version of PGIMH without topology constraints. We consider four different image features: the raw intensity of a pixel [69], the intrinsic intensity of a pixel [69] & [18], the shape operator [56], and the steerable pyramid output [53]. The intrinsic intensity is estimated *a priori*, meaning that a gain and bias field [18] are estimated and removed prior to segmentation. As PGIMH extends almost *any* segmentation algorithm to a boundary detector, the emphasis here is not on a particular energy functional or image feature, but rather the improved performance via marginal statistics (made feasible by PGIMH) compared to optimization. To avoid local minima in the optimization comparison, we run gradient descent with 100 random initializations and select the minimal energy configuration for each image. Results across the entire BSDS are shown in Figure 3.11. In addition to reporting performance on BSDS with the average F-measure (as is typical) we also report results using the optimal image-based threshold. While a measure of image complexity or contextual content might provide a means of approximating such a threshold, our purpose is to illustrate the achievable gains using the PB image. Regardless, results are reported using both global and image-based thresholds, and in either case, sampling improves upon the optimization approach across the majority of images in the dataset.

Figure 3.12 shows results on four specific images from the BSDS. Qualitatively, the PB image provides a superior demarcation of edges in the image. Quantitatively, the F-measure is also improved by thresholding the PB image rather than using the optimal sample path.

We remind the reader that the model is inferred based on the chosen posterior distribution, $p(z|x)$, and the error metric is computed as the F-measure with respect to ground-truth. Optimization based methods may find the best configuration of z according to $p(z|x)$, but this configuration may not correspond to the minimum error when comparing to ground-truth. In general, ground truth is not known, so one cannot

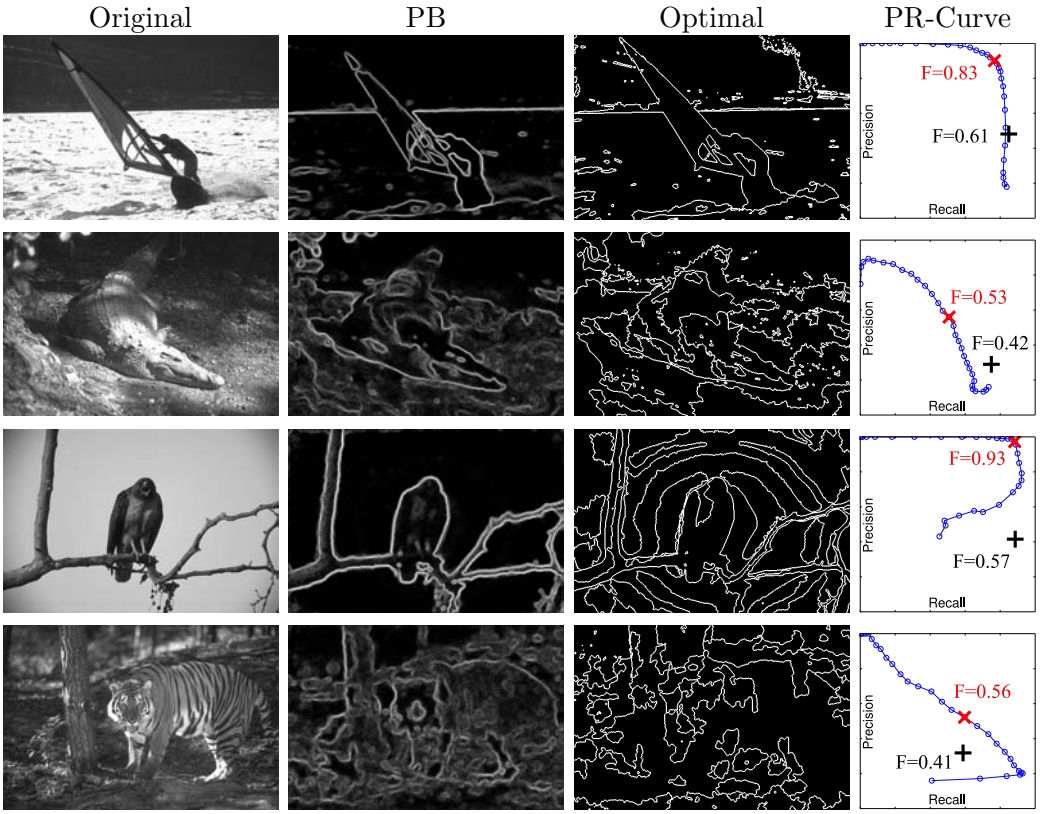


Figure 3.12: Example images from BSDS [88]. In the PR curves, the '×' marks the F-measure obtained using BFPS, and the '+' marks that of the optimal sample. The first two rows use the image feature of [69]. The third row also uses the image feature of [69] but with the gain and bias field of [18]. The fourth row uses the textural image feature of [56].

access the error metric of interest. However, the preceding experiments show that marginal event probabilities are more robust to noise and seem to better align with the ground-truth annotations for a particular task.

■ 3.6.4 Topology-Controlled Sampling

In this section, we show some capabilities of the topology controlled prior that one can impose using PGIMH. We impose four different topology constraints on the foreground: unconstrained (UC), topology-preserving (TP), genus-preserving (GP), and connected-component-preserving (CCP). The UC sampler allows any topology change, the TP sampler does not allow any topology changes, the GP sampler only allows the splitting and merging of regions, and the CCP sampler only allows the creation and destruction of handles.

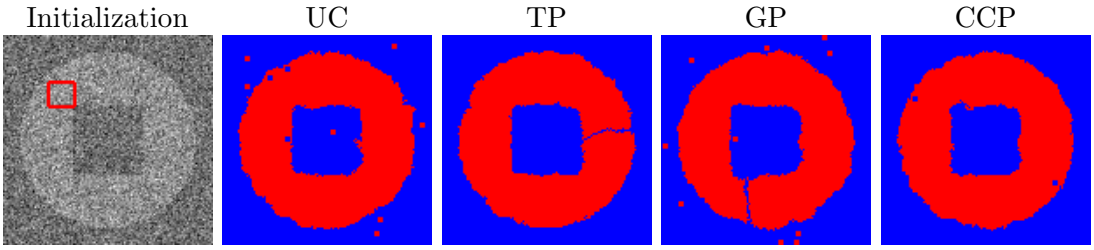


Figure 3.13: Example samples obtained by imposing different topology constraints.

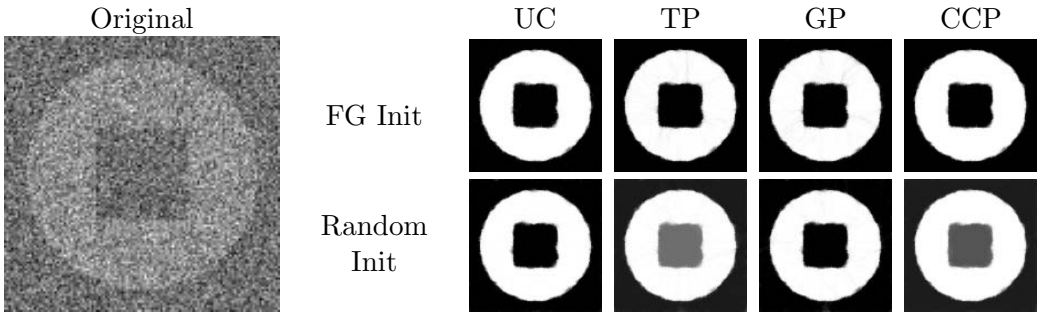


Figure 3.14: Histogram images obtained from different initializations and topology constraints.

Typical samples from each of these constraints are shown for a particular synthetic example in Figure 3.13. When the topology constraint is incorrect for the object (e.g., using TP or GP), the resulting sample may be undesirable (e.g., creating an isthmus connecting the two connected components of the background). When the topology is correct, however, more robust results can be obtained. For example, the CCP constraint removes some noise in the background.

The usefulness of the topology constraint relies on a valid initialization. The computed histogram images are shown in Figure 3.14 for each of the topology constraints. We initialize the samples either using a random circle containing the foreground (FG Init), or a random circle placed anywhere in the image (Random Init). While not always true, incorrect topology constraints are sometimes mitigated when looking at marginal statistics. For example, in the FG initialization case, the isthmus in the TP and GP constraints is no longer visible. Additionally, if the initialization only captures one connected component of the background (which may occur from random initializations), the priors that prohibit splitting regions (TP and CCP) cannot capture the entire region. This is reflected in the histogram image with the gray center.

Next, we consider the low SNR image of Figure 3.15. We have already shown that sampling improves results when compared to optimization based methods in low SNR cases. In these situations, the prior has a greater impact since the data is more ambiguous. The top row of Figure 3.15 shows the histogram images obtained using

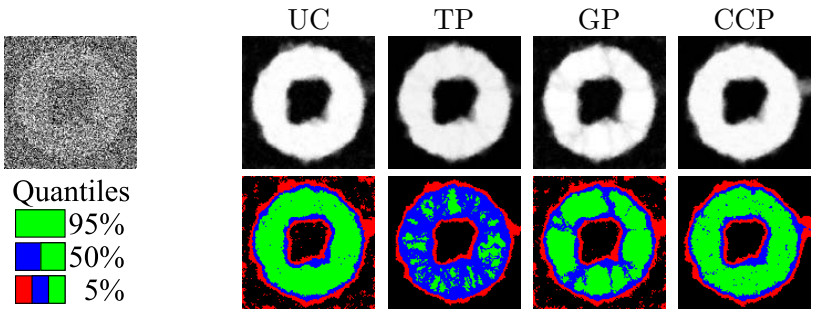


Figure 3.15: Results on low SNR images using different topology constraints. Initializations were chosen to be a random circle containing the foreground (FG Init). Histogram images are shown above, and quantiles (thresholded histograms) are shown below.

each of the topological constraints. One can see remnants of the isthmuses in the TP and GP cases. Thresholding the normalized histogram image at a value t reveals the t^{th} quantile of the segmentation. For example, if $t = 0.9$, the resulting foreground region of the thresholded histogram contains pixels that are in the foreground for at least 90% of the samples. We show the 95%, 50%, and 5% quantile segmentations in Figure 3.15. Since reducing the threshold never shrinks the foreground segmentation, we can overlay these quantiles on top of each other. In the 5% quantile, we can clearly see the isthmuses in the TP and GP cases. This result is poor because the wrong topology (i.e., the wrong prior) was used. However, if we use the CCP constraint, results improve as compared to the unconstrained case by removing a lot of the background noise.

The CCP constraint is particularly useful when an unknown number of handles exist (e.g., deformable objects). Objects with a known number of handles in 3D projected onto a 2D plane can have any number of handles. We show two example images of a human and the resulting thresholded histogram image in Figure 3.16. In the first image, the handles formed by the arms are not captured well with TP and GP. In the second image, the vignetting allows the UC and GP constraints to incorrectly group some background with foreground.

■ 3.7 Discussion

We have developed an MCMC sampling algorithm called the Permutation-based Gibbs-Inspired Metropolis-Hastings (PGIMH) algorithm. PGIMH converges to the stationary distribution of the Markov chain more than 4 orders of magnitude faster than the previous shape sampling algorithms of [23] and [31]. Moreover, because of the fast method we have developed for approximating curve-length with a local table lookup, traditional Gibbs sampling can also be used in these segmentation problems. However, it is well known that the local changes made in traditional Gibbs sampling are prone to converge to local extrema, a condition that PGIMH is more likely to overcome.

The use of a sampling algorithm such as PGIMH was applied to real-world data on

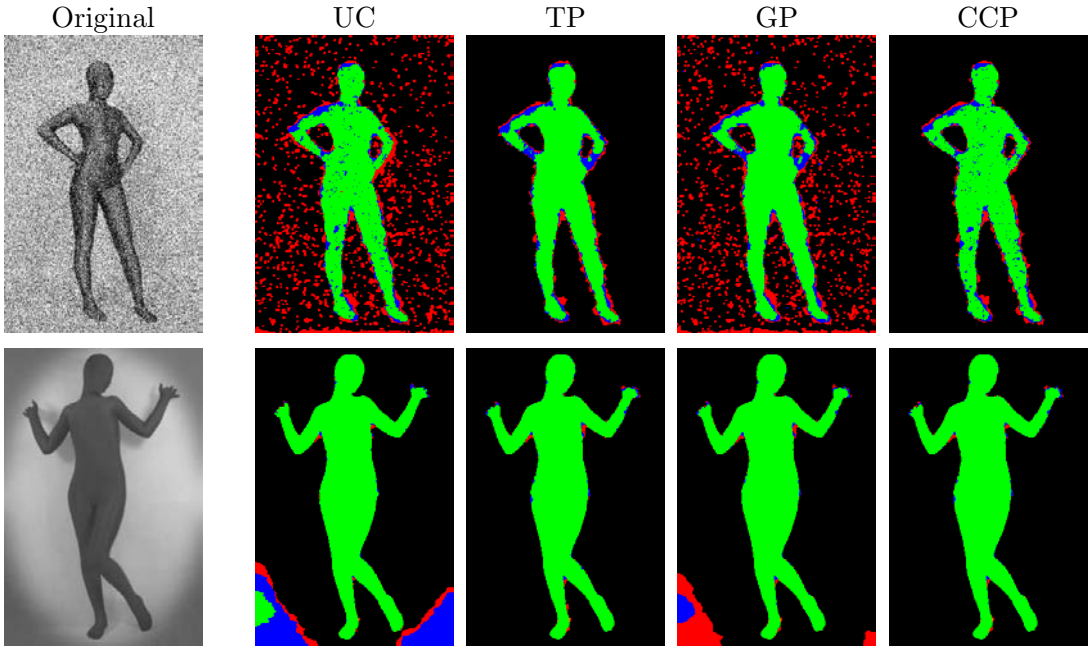


Figure 3.16: Example images illustrating the utility of topology priors.

the Berkeley Segmentation Dataset. Our results show that inference based on marginal statistics drastically outperform the point estimates obtained from maximum *a posteriori* inference. These results hold true across the vast majority of images, and even across different types of exponentiated energy functionals.

PGIMH can be used for many different types of priors, as indicated to in Section 3.4. Furthermore, our work is, to our knowledge, the first sampling-based approach to incorporate any explicit notion of topology constraints. While previous methods such as [23] and [31] were implicitly constrained to a single simply connected shape due to their limited representations, we have shown how one can actually *control* the topology of the shape. This contribution is significant even in the optimization literature, where [50] and [104] have only shown that topology-preserving and genus-preserving constraints can be enforced. In contrast, we have demonstrated that in 2 dimensions, any specific type of topology constraint can be enforced, such as the unconstrained, topology-preserving, genus-preserving, and connected-component-preserving constraints. Soft constraints (e.g., penalizing the number of handles or separate connected components) can also be integrated into the PGIMH framework.

Lastly, we have shown the relationship between PGIMH and blocked Gibbs sampling. In particular, PGIMH is more favorable than blocked Gibbs sampling since the complexity of the algorithm scales linearly with the block size instead of exponentially. It is interesting to note that PGIMH simplifies to traditional Gibbs sampling when the size of the block is 1. Because of the widespread use of Gibbs sampling, we hope that

PGIMH can aid in many discrete-labeling problems beyond just image segmentation.

We believe that one potential avenue of further research related to the PGIMH algorithm is to consider parallelizations. Some perturbations are inherently independent, conditioned on a suitable prior. For example, because the approximation for curve-length penalty only depends on a local 5×5 neighborhood, perturbations that do not have overlapping 5×5 neighborhoods can be computed independently. This becomes slightly more complicated with global priors such as those on topology or area. Regardless, a significant potential speed-up can be gained if PGIMH can be effectively parallelized.

Chapter 4

Shape Dynamics in Object Tracking

The previous chapter developed an efficient sampling framework for implicitly defined shapes and discrete Markov random fields. The development of PGIMH applies to many different energy functionals and priors, but is thus far limited to a single, static MRF. In this chapter, we consider the problem of inference in a model of multiple, temporally dependent MRFs. This situation often occurs in computer vision when working with videos instead of images, where one would like to perform object tracking or video segmentation.

Tracking and segmentation are fundamental tasks in video sequence analysis. The resulting tracks can be used to analyze past behavior and predict future trajectories of objects in the scene (e.g., [24]). Segmentation and motion analysis of video provides a preprocessor for object classification. Accurate object boundaries enable methods for learning shape models (e.g., [101]). We focus on object tracking with accurate boundaries in contrast to bounding box methods. Figure 4.1 illustrates the difference between the two related problems.

In this chapter, we present a generative probabilistic model using a layered representation of scenes. The resulting model for layered object tracking combines dynamic appearance and shape models, topology constraints, and Gaussian process flow. These concepts have been considered individually in a variety of contexts including tracking. Here, we consider an integrated model and develop efficient sampling-based algorithms. An earlier version of this was originally presented in [22].

Figure 4.1: Bounding box tracking versus boundary accurate tracking. The object boundaries were found using the algorithm described in this paper.

4.1 Related Work

Elements of the approach are certainly related to previous work. Layered models have been popular since the Bayesian model of [28], with the promising results of [123] motivating many similar approaches. The layered appearance model described herein is closely related to [62, 102], but as described in Section 4.2, differs explicitly by coupling occlusions and disocclusions with the layer supports. Similarly, Section 4.3 discusses how our flow model generalizes that of [124] by using Gaussian processes.

Owing to the wealth of literature in tracking and segmentation, we focus on relevant prior work. Some optimization-based approaches, such as [49, 77, 84], automatically *segment* objects of interest by processing the entire video offline. Others (e.g., [118]) require annotations to identify key objects or frames. Such batch processing is akin to Bayesian *smoothing*, where inference depends on both past and future observations. Alternatively, analogous to Bayesian *filtering*, one can *track* objects in an online fashion (e.g., [101, 102, 108]). The proposed method adopts a filtering approach, scaling linearly in the number of frames for computation, and having constant memory consumption.

There is also a rich literature in probabilistic tracking, most of which track bounding box trajectories instead of accurate boundaries. A notable exception is [101], which uses an unconventional particle filter that propagates particles with a number of gradient descent iterations instead of a randomized proposal. We show that this deterministic approximation of a randomized algorithm is unnecessary. Additionally, the dynamics of [101] do not correspond to an actual object motion. Rather, the dynamics are chosen to be a mean translation of the convex combination of predefined shapes learned from training data. This approach can be quite limiting, and it is not clear how many training examples are needed to accurately represent highly-deformable objects. One final difference between the proposed approach and the method of [101] is that their method essentially uses a bag-of-pixels model for the foreground and background region. They assume that pixels are distributed independently conditioned on the region label according to a Gaussian distribution with unknown mean and variance. Consequently, appearance changes cannot be coupled with shape changes, which we know must occur in the physical world.

To our knowledge, only the optimization-based methods of [108] and [35] consider topology constraints in video analysis. The first uses digital topology to penalize merges of two connected components (each corresponding to a separate object). This soft constraint does not, however, necessarily preserve connected components. Similarly, the second extends binary topology to K-ary topology to restrict objects from merging. This method does not use an appearance or flow model and also does not handle occlusions that may split the visible support of objects.

Similiar to the proposed method, Sun et al. [113, 114] estimate motion and reason about depth ordering in a layered model. In fact, [114] extends [113] to infer the *number* of layers, something the proposed method does not consider. In contrast to the proposed method, these methods segment objects via batch processing of the entire sequence. Unlike our method, which reasons over the shape *distribution*, [114] adopts an optimization

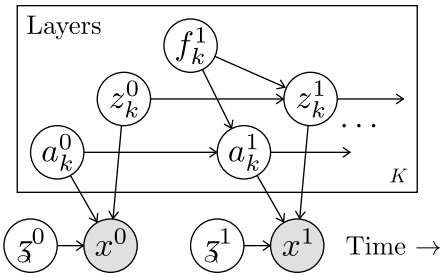


Figure 4.2: The graphical model associated with our approach. x denotes an image, z_k, a_k, and f_k are the support, appearance, and flow for layer k, and $\mathfrak{z}$ controls the ordering of the layers.

approach within a probabilistic formulation. While [113] achieves state-of-the-art optical flow results, it does so at considerable computational expense. Furthermore, these methods do not allow for the temporal evolution of layer ordering. Processing times on current standard computer hardware are on the order of hours per frame as compared to approximately one minute per frame for the proposed approach.

■ 4.2 Layered Model

We begin by developing the probabilistic model used to represent scenes, depicted as a directed graph in Figure 4.2. While it is important for any practical method to be able to detect new objects entering a scene, for purposes of exposition, we restrict ourselves to the case of tracking K previously detected objects. The method of initialization is discussed in Section 4.5.

■ 4.2.1 In-Frame Appearance

Scene models are comprised of K layers, where each object exists in one layer, and one layer is the designated background. The support of layer k at time t is denoted by $z_k^t \in \{0,1\}^N$, where N is the number of pixels. Specifically, $z_{k,i}^t = 1$ iff pixel i is in the support of layer k at time t. The layers are ordered with $\mathfrak{z}^t$, which contains a permutation of the integers 1 to K. The visible layer at pixel i, denoted v_i^t, can then be expressed as

$$v_i^t = \arg\min_{\{k; z_{k,i}^t = 1\}} \mathfrak{z}_k^t. \tag{4.1}$$

Associated with each layer is a pixel-wise appearance model, $a_k^t \in \mathbb{R}^{N\times 3}$, where $a_{k,i}^t$ is the 3-dimensional color (we use the *Lab* colorspace) at pixel i. Assuming Gaussian observation noise, the observed image, x^t is generated by

$$x_i^t | a^t, z^t, \mathfrak{z}^t \sim \mathcal{N}\left(x_i^t \,;\, a_{v_i^t,i}^t, \Sigma^x\right). \tag{4.2}$$

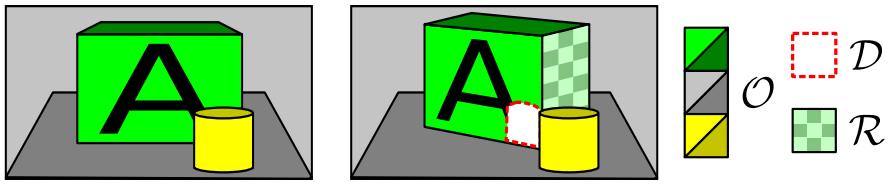


Figure 4.3: An example of the three types of pixels that can occur in a new frame.

We note that the visible pixel, v_i^t, is implicitly dependent on z^t and $\mathfrak{z}^t$ through Equation (4.1). We assume the color channels are independent (i.e., Σ^x is diagonal). While [62] and [102] use similar models, they do not address the appearance of occluded or disoccluded pixels, which we now discuss.

■ 4.2.2 Temporal Appearance Dynamics

Given the appearance and support of a previous frame, the dynamics of these random variables evolve jointly based on the underlying motion of the layer, denoted f_k^t. We defer the discussion of the motion model for Section 4.3. While deformations of 3D objects are typically diffeomorphic, projections onto the 2D image plane are not because of occlusions and disocclusions. For example, consider the illustration in Figure 4.3 where a visible pixel in the new frame can belong to three possible portions of a layer:

1. Observed regions, $\mathcal{O}$, which have been seen before;
2. Disoccluded regions, $\mathcal{D}$, which have never been seen and were previously occluded by another layer;
3. Revealed regions, $\mathcal{R}$, which have never been seen and were previously hidden by a pixel belonging to the same layer.

Consequently, we define a probability distribution over the evolving appearance model for each of these categories.

Observed regions, $\mathcal{O}_k^t$, are chosen to evolve with a Gaussian distribution from the appearance model

$$p(a_{k,i}^t | i \in \mathcal{O}_k^t, fa_k^{t-1}) = \mathcal{N}\left(a_{k,i}^t; fa_{k,i}^{t-1}, \Sigma^a\right), \tag{4.3}$$

where fa_k^{t-1} denotes the aligned appearance obtained by evolving a_k^{t-1} with the flow f_k^t, and $fa_{k,i}^{t-1}$ indexes pixel i from the image fa_k^{t-1}. Similar to the observation, we assume independent color channels with a diagonal covariance, Σ^a.

Disoccluded regions are often similar to neighboring pixels. For example, the disoccluded pixels in Figure 4.3 are likely to be green or black. Because no additional prior information is given, we model these pixels as being drawn from a mixture of

neighboring pixels according to

$$p(a^t_{m,i}|i \in \mathcal{D}^t_m, fa^{t-1}_m) \propto \sum\nolimits_{j \in \mathcal{O}^{t-1}_m} \mathcal{N}\big(j; i, \sigma^2_{\mathcal{D}}\mathrm{I}\big)\, \delta\Big(a^t_{m,i} - fa^{t-1}_{m,j}\Big), \tag{4.4}$$

where $\mathcal{N}\big(j; i, \sigma^2_{\mathcal{D}}\mathrm{I}\big)$ is a 2D Gaussian over pixel coordinates.

Lastly, revealed regions appear when objects turn and reveal a new side. These can look quite different from neighboring pixels. For example, the revealed side of the box in Figure 4.3 can be of any color. Appearances in $\mathcal{R}^t_m$ are therefore drawn from a mixture of a uniform distribution and a kernel density estimate of the observed appearances

$$\begin{aligned} p(a^t_{m,i}|i \in \mathcal{R}^t_m, a^{t-1}_m) &= (1-\alpha_R)\mathcal{U}^3(a^t_{m,i}) + \alpha_R \kappa(a^t_{m,i}), \\ \kappa(a) &= \tfrac{1}{|\mathcal{O}^{t-1}_m|} \prod\nolimits_d \sum\nolimits_{j \in \mathcal{O}^{t-1}_m} \mathcal{N}\Big(a_d; a^{t-1}_{m,j,d}, \sigma^2_k\Big)\,, \end{aligned} \tag{4.5}$$

where $\mathcal{U}^3$ is the uniform distribution over the colorspace, d indexes a color. $\kappa(\cdot)$ can be estimated with [128] and σ^2_κ is chosen to be the "rule-of-thumb" bandwidth. While these simple assumptions do not explain all situations, we have empirically found that they work well in most videos.

■ 4.2.3 Temporal Support Dynamics

The evolution of each layer support is coupled to the evolution of its appearance. However, by introducing disoccluded and revealed pixels, we must also allow the support of each layer to deviate from the aligned support, fz^{t-1}_k. Consequently, layer supports are chosen to evolve according to a distribution proportional to exponentiated symmetric area difference (SAD), where SAD can be expressed as

$$\mathrm{SAD}(z^1, z^2) = \sum\nolimits_i \mathbb{1}\big[z^1_i \neq z^2_i\big], \tag{4.6}$$

and $\mathbb{1}[\cdot]$ is the indicator function that is one iff $[\cdot]$ is true. Additionally, a curve length penalty is placed on the shape of each layer, and the support of each layer is restricted to be a single connected component. The resulting temporal dynamics on layer support are expressed as

$$p(z^t_k|z^{t-1}_k, f^t_k) \propto Q_L(z^t_k) \prod\nolimits_i Q_S(z^t_{k,i}|fz^{t-1}_{k,i}), \tag{4.7}$$

$$Q_L(z^t_k) = \mathbb{1}\big[T(z^t_k) = 1\big] \exp\left[-\alpha_L L^t_k\right], \tag{4.8}$$

$$Q_S(z^t_{k,i}|fz^{t-1}_{k,i}) = \exp\left[-\alpha_S \mathbb{1}\Big[z^t_{k,i} = fz^{t-1}_{k,i}\Big]\right], \tag{4.9}$$

where L^t_k is the contour length of z^t_k, $T(\cdot)$ counts the number of connected components, and the α_L, α_S control the relative weighting of these penalties.

The chosen appearance and shape models will not explain every situation. For example, an object that splits into two will violate the topology prior of the model.

If a light is turned on, the Gaussian diffusion of appearances may be too restrictive. However, in subsequent sections we show that they yield good empirical performance across a variety of video sequences.

■ 4.3 Gaussian Process Flow

Having detailed the observation model *conditioned* on the layered flow fields, we now describe a flow model comprised of layered Gaussian processes (GPs). A GP can be parametrized with a mean and covariance function (cf. [100]). We restrict the model to zero-mean GPs with stationary covariance kernels, which, as shown in Section 4.4, enables an efficient sampling-based inference method.

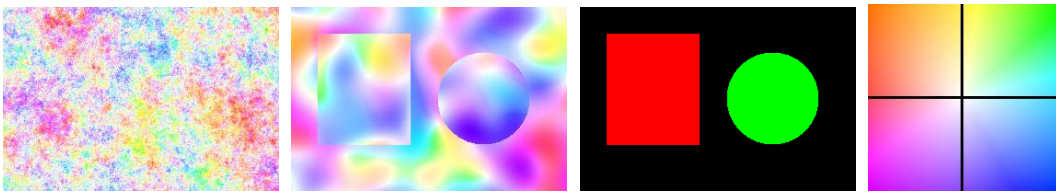
GPs have been widely used as a prior for trajectories (e.g., [11, 70, 120]). In practice, however, these formulations have been applied to object trajectories, whereas here we consider their application to dense flow between frames. GPs are not typically used to model flow for two reasons: (1) GPs are often smooth everywhere, causing errors across object boundaries; and (2) naïve inference requires inverting covariance matrices that do not scale well with the image size. We address the first issue here and the second via an approximation in Section 4.4.

One particular GP covariance kernel is closely related to the L_2 penalty in the Horn-Schunck optical flow formulation [55]. However, when observations are not present, the L_2 penalty on flow *differences* results in an improper distribution with a rank deficient covariance matrix (cf. [87]). Regardless, one can approximate this prior with a GP having a sparse precision matrix (inverse of the covariance), arising from the 4-connected neighborhood of each node. Figure 4.4a shows a sample from this type of GP, which fails to capture long range correlations in flow fields and, concurrently, overly penalizes discontinuities across object boundaries. Following [124], we instead compose smooth, layered flows using

$$\overline{f}_i = [f_{v_i}]_i \qquad \forall i \in \{1, \cdots, N\}, \tag{4.10}$$

where v_i denotes the visible layer at pixel i according to Equation (4.1). Here, $[f_k]_i$ denotes the i^{th} pixel of the flow for layer k, and $\overline{f}$ is the composite flow for visible pixels. Similar to optical flow, we assume that the x and y components of flow are independent. Figure 4.4b shows a sample from a GP using Equation (4.10) with the squared exponential (SE) kernel (cf. [100]). Composing multiple smooth GPs in this fashion mitigates the problem of discontinuities.

Unlike [124], which uses layered flows of the form of [55], a GP flow is a valid prior distribution that can capture long-range dependency. Any 2D GP has an equivalent Gaussian Markov random field (GMRF). The graphical structure of the corresponding GMRF depends on the particular covariance kernel, and is often quite complex, impacting inference. However, by utilizing a GP formulation, we show in Section 4.4 that inference is easily adapted to changes in the covariance.



(a) (b) (c) (d)

Figure 4.4: Samples from different GPs: (a) sparse neighborhood precision; (b) SE kernel with layered composition using the layers of (c). (d) Flow vectors mapping to colors.

Figure 4.5: Consecutive frames of a deformable object with self occlusions and disocclusions.

■ 4.3.1 Smooth Deformable Flow

Tracking deformable objects encounters additional complexity when parts of the objects exhibit self-occlusions or disocclusions (e.g., the arms and legs of the girl in Figure 4.5). The SE kernel often results in overly smoothed flow estimates that do not adequately capture object deformation. We mitigate this issue by adopting a covariance kernel that is the composition of the SE kernel and a delta function. It is easily verified that the resulting GP follows the distribution

$$p(f_k) = \mathcal{N}\big(f_k \,;\, 0, \Sigma^g + \sigma_f^2 \mathrm{I}\big)\,, \tag{4.11}$$

where Σ^g is the resulting covariance from the SE kernel, and $\sigma_f^2\mathrm{I}$ is the resulting covariance from the delta kernel. Notation is slightly abused since each component of the flow is not explicitly written. We show in Section 4.4 how to exploit a decomposition of this flow for efficient inference. In particular, we decompose the flow into a smooth flow (denoted g_k) and the remaining independent part with

$$p(g_k) = \mathcal{N}(g_k \,;\, 0, \Sigma^g)\,, \tag{4.12}$$

$$p(f_k|g_k) = \mathcal{N}\big(f_k \,;\, g_k, \sigma_f^2\mathrm{I}\big)\,. \tag{4.13}$$

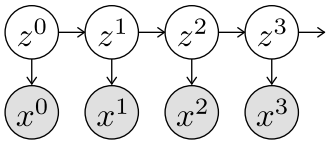


Figure 4.6: Markov chain that can be inferred via particle filtering.

4.4 Inference

Having developed a generative dynamic appearance and shape model, we present a Bayesian *filtering* procedure that reasons about the distribution of the hidden variables conditioned on past and current observations. Exact inference is generally infeasible for complex distributions, as is the case here. A typical approach is to use a particle filter [59] where the distribution at any time is represented by a set of weighted samples. Particle filters propagate these samples and corresponding weights through time, typically using the prior distribution over temporal dynamics (cf. Section 2.5.2). Many applications are, however, likelihood-dominated (e.g., segmentation and tracking). This causes particles to move to low likelihood regions and the weights to decay over time, indicating a poor representation of the desired distribution. Sequential importance resampling (SIR) techniques are typically utilized to mitigate this issue.

Alternatively, instead of propagating particles solely based on the prior, one can incorporate data information to propagate samples so that they do not move to low likelihood configurations. These methods typically require making additional approximations while still needing SIR techniques to maintain an accurate particle representation. For example, [25] updates particles using a Hybrid Monte Carlo algorithm that exploits gradient information and incorporates an expensive Metropolis-Hastings rejection step. Methods such as [17] and [29] attempt to keep particles near modes but still require weight updates and may not accurately represent tails of distributions. [101] approximates the randomized propagation with a fixed number of deterministic gradient iterations.

Here, we propose a more accurate particle propagation approach by incorporating both the prior and the data likelihood terms in the PGIMH framework presented in Chapter 3. In this case, similar to a Gibbs sampler, weight updates are unnecessary because samples are drawn from the true conditional distribution. Following this discussion for a general Markov chain, we describe the particular inference scheme for the presented model.

4.4.1 Efficient Particle Filtering without Weight Updates

Consider the general Markov chain described in Section 2.5.2 where z denotes the set of hidden variables, and x denotes the set of observed variables. The graphical model representing this relationship is reproduced in Figure 4.6.

Proposition 4.4.1. (Particle Filtering without Weight Updates) *Let* $\{z_1^t, \ldots, z_S^t\}$ *be a set*

of S samples, each drawn from some density $q(z^t)$, and w_s^t be the importance weight for sample s such that the set of weighted samples approximates the posterior distribution at time t, denoted $p(z^t|x^0,\dots,x^t)$. If at time $t+1$, a sample s is drawn from the posterior

$$z_s^{t+1} \sim p(z^{t+1}|x^{t+1}, z^t = z_s^t), \tag{4.14}$$

the weights do not need to be updated to accurately represent the new posterior distribution, $p(z^{t+1}|x^0,\dots,x^{t+1})$.

Proof. A rigorous proof is given in Appendix A.1. □

Proposition 4.4.1 should not be very surprising. Intuitively, Equation (4.14) is the posterior distribution conditioned on a sample from the previous time point. Thus, sampling from it directly must trivially correspond to a valid sample from the posterior as long as z_s^t was a valid sample, which is assumed in the proposition.

While propagating the particles with both the data and prior is more accurate and avoids weight decay, this approach is typically avoided because sampling from the full conditional distribution in Equation (4.4.1) is often computationally prohibitive. Fortunately, the PGIMH algorithm developed in Chapter 3 addresses this issue.

We now describe the overall inference procedure. The presence of many hidden variables (g, f, z, a, and $\mathfrak{z}$) suggests multiple possible sampling schemes. Here, we discuss two different layer and flow samplers, after which the layer orderings, $\mathfrak{z}$, are sampled by enumerating over the possible total orders. The first iterates between sampling smooth flow and labels, which benefits from large moves in the space of layer supports. In our experiments, this sampling algorithm occasionally fails to converge in certain situations that will be described shortly. The second sampling procedure addresses these issues by jointly sampling the flows and level sets at one particular pixel for all layers simultaneously. This procedure exhibits poor convergence in isolation because it only makes very local changes to the support. Consequently, we alternate between both samplers, noting that switching between two valid samplers will still preserve detailed balance.

Additionally, both sampling procedures marginalize out some form of the flow. As noted in [114], this type of joint inference generally exhibits better convergence properties. However, owing to our particular formulation, the samplers are more efficient than the proposed optimization moves in [114].

4.4.2 Single Layer Sampler

The first sampling algorithm iterates over the following steps

$$g^t \sim p(g^t|f^t), \tag{4.15}$$
$$z^t, a^t \sim p(z^t, a^t|g^t, z^{t-1}, a^{t-1}, x^t, \mathfrak{z}^t), \tag{4.16}$$
$$f^t \sim p(f^t|g^t, z^t, z^{t-1}, a^{t-1}, x^t, \mathfrak{z}^t), \tag{4.17}$$
$$\mathfrak{z}^t \sim p(\mathfrak{z}^t|f^t, z^t, a^{t-1}, x^t). \tag{4.18}$$

Sampling from $g^t|f^t$ is equivalent to sampling a GP with observations. With some manipulation from the typical GP regression (cf. [100]), we show in Section 2.9.1 that this distribution can be expressed as

$$g_k^t|f_k^t \sim \mathcal{N}\left(\mu_g^*, \Sigma^*\right), \tag{4.19}$$
$$\mu_g^* = \Sigma^g[\Sigma^g + \sigma_f^2\mathrm{I}]^{-1} f_k^t \ , \ \ \Sigma^* = \Sigma^g - \Sigma^g[\Sigma^g + \sigma_f^2\mathrm{I}]^{-1}\Sigma^g.$$

Sampling from this expression is difficult because of the dimension of the GP. By drawing on the work of [109], we show in Section 2.9.1 that a GP with a stationary covariance kernel, $\kappa(x - x')$, can be approximately sampled with

$$g_k^t|f_k^t \sim \left[h_\mu * f_k^t\right] + \mathcal{N}(0, \mathrm{I}) * h_\Sigma, \tag{4.20}$$
$$h_\mu = \mathcal{F}^{-1}\left\{\frac{\mathcal{K}}{\mathcal{K}+\sigma_f^2}\right\} \ , \ h_\Sigma = \mathcal{F}^{-1}\left\{\sqrt{\mathcal{K} - \frac{\mathcal{K}^2}{\mathcal{K}+\sigma_f^2}}\right\}.$$

Here, $\mathcal{K}$ denotes the Fourier transform of κ. We note that this approximation degrades closer to image boundaries.

While sampling the layer supports of Equation (4.16) utilizes thousands of the PGIMH iterations described in Algorithm 3.4, this entire process takes less than one second. As each iteration perturbs only a single layer, we refer to this procedure as the "Single Layer Sampler." A layer, k, is first chosen at random, and its support is sampled from the following posterior distribution

$$\begin{aligned} p(z_k^t, a_k^t|g^t, z^{t-1}, a^{t-1}, x^t, \mathfrak{z}^t) &\propto \int p(f_k^t|g_k^t)p(z_k^t|fz_k^{t-1})p(a_k^t|fa_k^{t-1})p(x^t|z^t, a^t, \mathfrak{z}^t)\, df_k^t \\ &= Q_L(z_k^t)\prod\nolimits_i \int p(f_{k,i}^t|g_{k,i}^t)\mathcal{L}_{k,i}^t(f_{k,i}^t - g_{k,i}^t)\, df_{k,i}^t \end{aligned} \tag{4.21}$$

where the pixel-wise likelihood term, $\mathcal{L}_{k,i}^t(j)$, is

$$\mathcal{L}_{k,i}^t(j) = Q_S(z_{k,i}^t|gz_{k,i+j}^{t-1})p(a_{k,i}^t|ga_{k,i+j}^{t-1})p(x_i^t|z_i^t, a_i^t, \mathfrak{z}^t) \tag{4.22}$$

and we have used the fact that

$$f(\cdot)_{k,i}^{t-1} = g(\cdot)_{k,i+f_{k,i}^t - g_{k,i}^t}^t. \tag{4.23}$$

We can *marginalize* f_k^t by approximating $p(f_{k,i}^t | g_{k,i}^t) = \mathcal{N}\left(f_{k,i}^t - g_{k,i}^t, \sigma_f^2\right)$ with a discrete finite impulse response filter, h_f. The distribution over z_k^t in Equation (4.16) is then proportional to

$$p(z^t | g^t, z^{t-1}, a^{t-1}, x^t, \mathfrak{z}^t) \propto Q_L(\ell_m^t) \prod_{i=1}^N \sum_j h_f(j) \mathcal{L}_{m,i}^t(j), \tag{4.24}$$

which is efficiently sampled using PGIMH described in Algorithm 3.4. Equation (4.24) is efficiently evaluated for h_f with small support. A detailed derivation is included in Appendix A.2. Changing the k^{th} layer accomplishes one of the following:

1. Grow and occlude the currently visible layer;
2. Grow behind the currently visible layer;
3. Shrink and disocclude the next layer;
4. Shrink behind the currently visible layer.

Each situations can be expressed with Equations (4.2)–(4.9). Furthermore, the Gaussian appearance dynamics and observation models allow for efficient marginalization of previously observed appearances. For disoccluded and revealed appearances, we draw a sample from the distributions in Equation (4.4)–(4.5). Similarly, f_k^t is approximately sampled from Equation (4.17) by multiplying Equation (4.24) with $\mathbb{1}\left[f_{k,i}^t = g_{k,i}^t + j\right]$.

We conclude a full iteration by sampling the layer ordering, $\mathfrak{z}^t$. Because of the uniform prior on orderings, Equation (4.18) is proportional to the observation likelihood, which can be computed using Equations (4.2)–(4.9). The videos we analyze typically have fewer than ten objects. As such, it is efficient to simply enumerate all possible total orders. When the number of layers is larger, a swap proposal in a Metropolis-Hastings framework can be used.

■ 4.4.3 Multiple Layer Sampler

The second step of the preceding algorithm samples the support of a single layer at once, which can exhibit slow convergence in certain situations. For example, if layers $k = 1, 2, 3$ all have support at pixel i, but the actual visible layer should be the background ($k = 3$), the sampler must move through an intermediate state before making the background visible. Consequently, this approach may converge to a local extrema when the intermediate state is unlikely. We now detail an alternative sampling algorithm that samples the layer support at a a single-pixel for all layers jointly. First, a visible

layer is sampled according to the following categorical distribution

$$\Pr(v_i^t = k) = p(x_i^t | v_i^t = k) \Pr[z_{k,i}^t = 1] \prod_{\{\ell; \mathfrak{z}_\ell^t < \mathfrak{z}_k^t\}} \Pr[z_{\ell,i}^t = 0], \tag{4.25}$$

where the conditioned variables have been omitted for convenience. If layer k is visible, all layers above k are explicitly precluded from having support at pixel i and all layers below are sampled from Equation (4.7). This process is repeated for all pixels in a random order. In practice, we sample from Equation (4.16) by running a full iteration of this "Multiple Layer Sampler" followed by a full iteration of the "Single Layer Sampler".

4.5 Experiments

Having described a model and associated inference procedure, we now compare the performance of the proposed method with results reported in the literature. As noted, we assume that objects of interest have been detected. In our experiments, simple user annotations followed by Lazy Snapping [78] are used to initiate tracking. Parameters are fixed across all sequences and we have verified that a wide range of parameters yield similar performance.

4.5.1 Implementation Details

Flow is an inferred latent variable and, theoretically, initialization does not impact convergence guarantees. However, as with any iterative procedure, convergence may be to a local mode. We have found that using a combination of the optical flow estimates of [13] and [82] as an *initialization* improves convergence empirically. For each frame, both flows are calculated, and the flow that minimizes the L_2 warped image difference is used for the initial value of $\overline{f}$.

Additionally, while a pixelwise appearance model is used, blurring effects caused by the image acquisition are not explicitly modeled. This causes boundary pixels between two regions to have a value that is the convex combination of the bordering regions. While this image acquisition phenomenon could potentially be incorporated in the model, we find that a simple procedure for fixing edges works well. We run each frame through three Sobel filter, one for each color channel. Then, we threshold each image according to the procedure described in [98]. We define an edge pixel as a pixel that is declared to be an edge in any of the color channels. The color of each edge pixel is then set to the color of the nearest non-edge pixel. An example of this process is shown in Figure 4.7. We find that preprocessing the frames with this simple edge-sharpening procedure slightly improves results.

4.5.2 Tracking

For each video frame, we draw 100 samples and consider pixels that appear in at least T of the sampled layer supports. Of this set, we take the largest connected component.

Figure 4.7: Visualization of the edge sharpening. Original image (left), edge detection (middle), edge sharpened image (right). Bottom row shows a detail of the image.

Table 4.1: Average number of incorrect pixels per frame on SegTrack.

	Human	Ours	[84]	[77]	[118]
birdfall	130	265	189	288	252
cheetah	308	570	806	905	1142
girl	762	841	1698	1785	1304
monkeydog	306	289	472	521	563
parachute	299	310	221	201	235
penguin	279	456	-	136285	1705
Mean Error	347	455	677*	740*	867

* indicates the exclusion *penguin*.

Results are shown for $T = 25\%$, but other confidence levels may be useful for other applications. Quantitative results on the SegTrack [118] dataset with the top three state-of-the-art algorithms are shown in Table 4.1. We note that [84] and [77] do not require the first frame to be segmented; however by depending on future data, these methods are a form of Bayesian smoothing instead of filtering. Additionally, we hand-label each video separately from the ground truth as a means to gauge human error. The proposed method achieves state-of-the-art results on most of the videos. Tracked frames and initial annotations for select SegTrack videos are shown in Figure 4.8. We note that there is ambiguity in the parachute sequence; our algorithm incorporates the human (as shown in the last frame) whereas the ground truth and other algorithms do not.

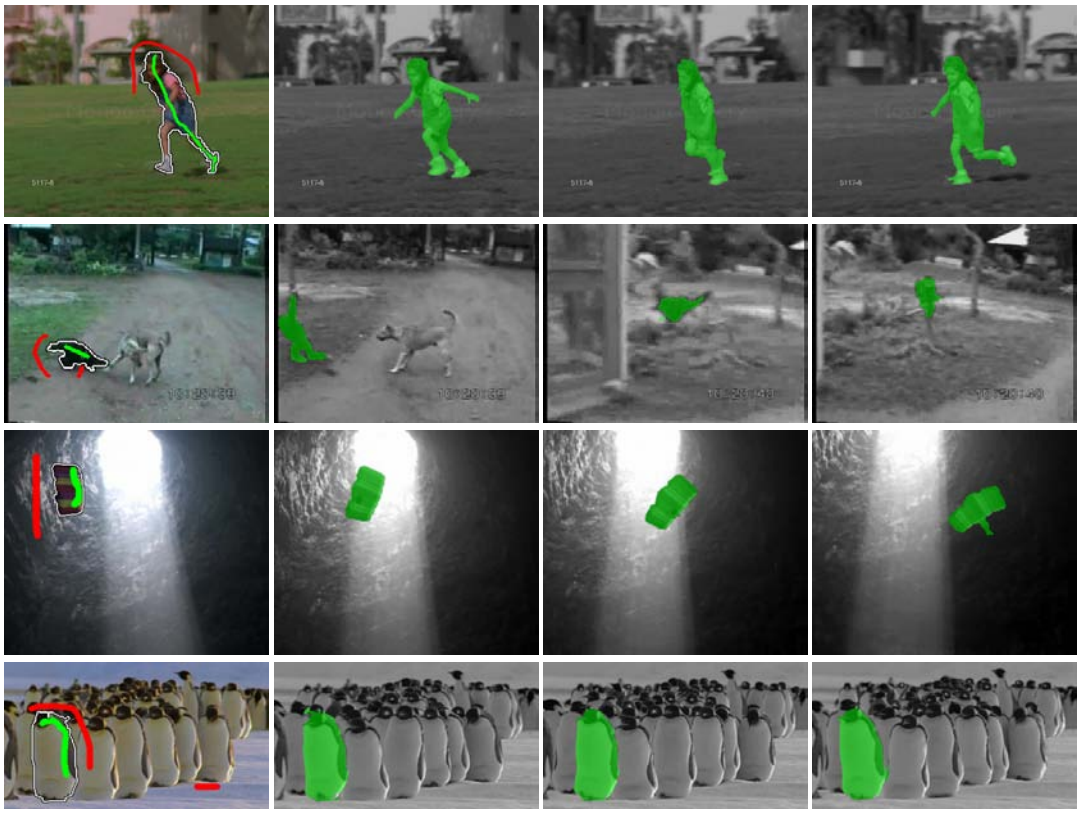


Figure 4.8: Four results on the SegTrack dataset [118]. First column shows user annotation and segmentation.

Since SegTrack only contains ground truth for single object tracking, we show additional results from the datasets of [49] and [82] in Figure 4.9. While good results are achieved for most videos, the first video of Figure 4.9 exhibits a failure of the proposed approach when a car in the background explodes. Such rapid appearance changes are not well represented by Gaussian evolution. As such, the flames in this video are erroneously labeled as foreground. On the other hand, the last video is over 500 frames long, and the proposed approach tracks the ice skater throughout the entire sequence.

■ 4.5.3 Inferring Layer Order

In this section, we show a visualization of the inferred layer order. While we do not impose temporal dependencies among layers, many videos have a static layer order for all frames. We can calculate the posterior distribution over layer orders for the entire video by treating each frame as an independent observation. We show this distribution for two videos in Figure 4.10. In the first video, the posterior distribution is only non-zero for the three orderings where the printer tray is in front of the printer. The uncertainty

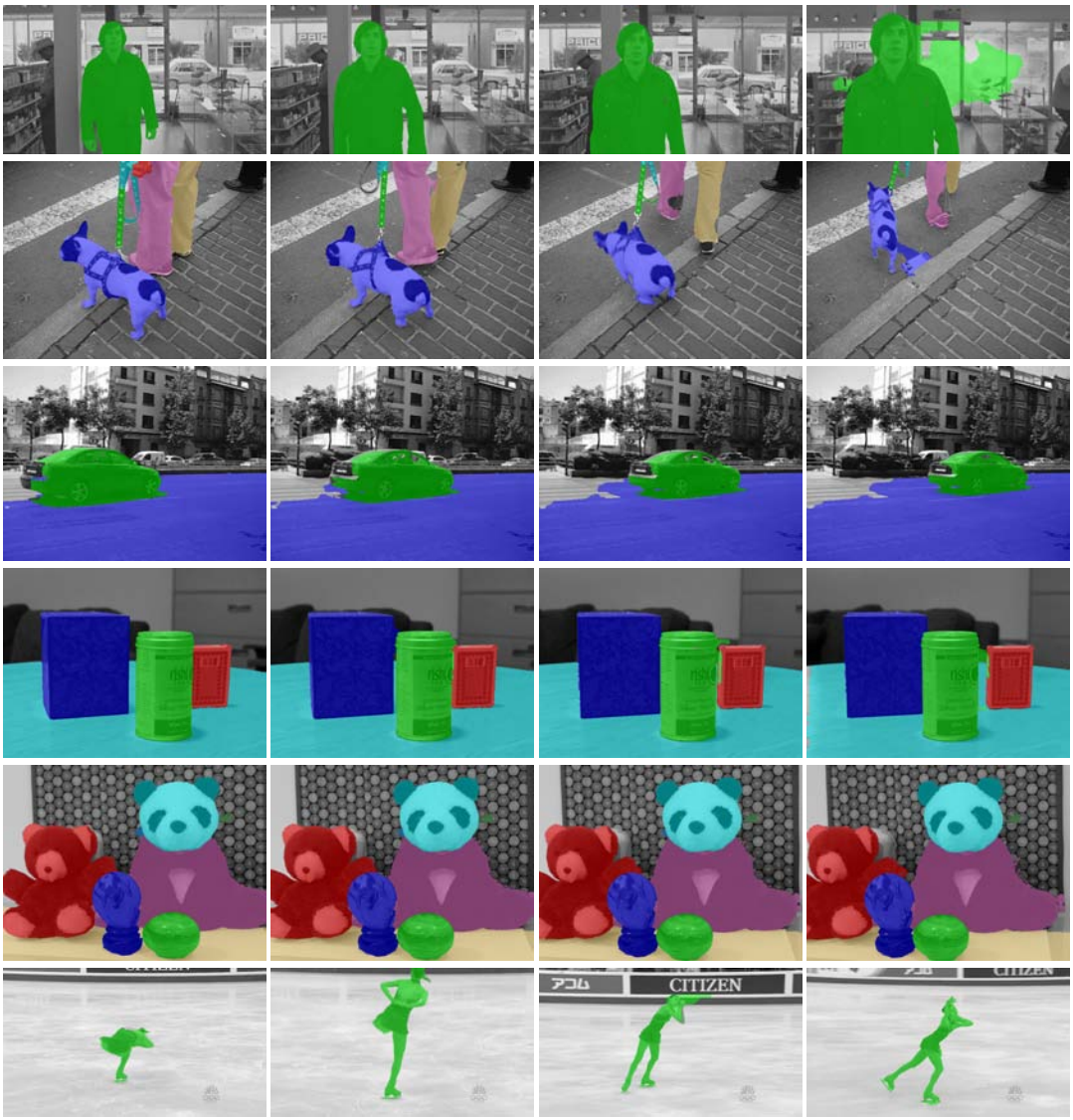


Figure 4.9: Results on the datasets of [49] and [82].

is expected because the phone and printer do not overlap during the sequence. In the second example, we consider the penguin video of SegTrack. Although the ground truth segmentation only tracks one penguin, we track three here. Because the penguins overlap, the posterior distribution is essentially a delta function at the correct layer order.

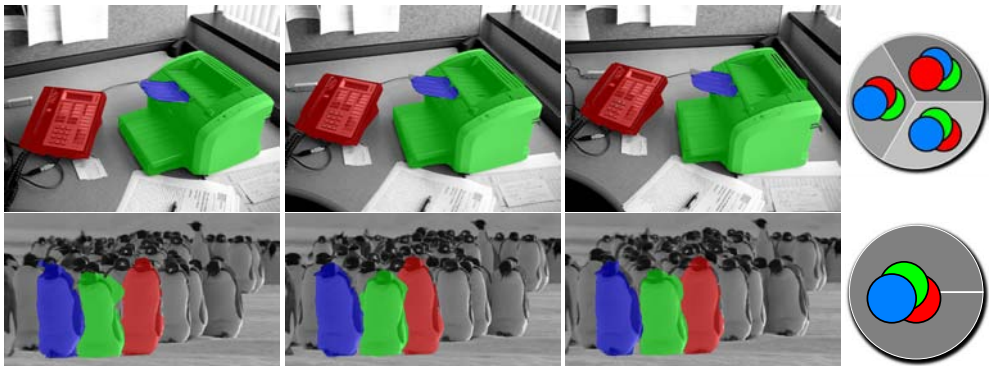


Figure 4.10: Frames and pie charts showing the posterior distribution over orderings.

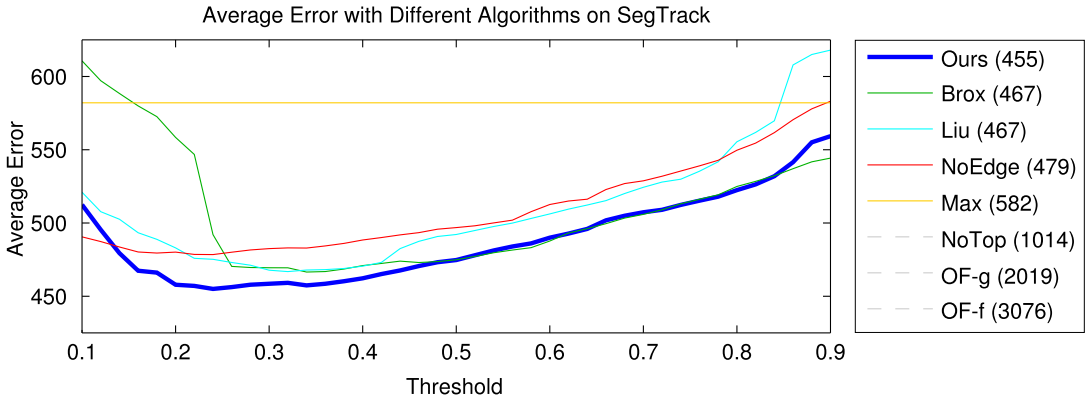


Figure 4.11: Average errors on SegTrack using different versions of our algorithm. Numbers in legend indicate the best achievable errors. Gray algorithms lie outside the limits.

■ 4.5.4 Independent Contributions

We analyze seven variations of our algorithm in order to test different aspects of the model. We consider using only one optical flow algorithm as an initialization (Brox [12] and Liu [82]), not using edge-sharpening (NoEdge), using an optimization scheme (Max), not enforcing topology constraints (NoTop), and treating optical flow as a measurement (OF-g and OF-f). For the optimization-based inference, we change all sampling steps to maximization steps. When using optical flow as a measurement (which is done in [49, 77, 84]), we can choose to either equate it to g (OF-g) or f (OF-f). The resulting average errors on the SegTrack dataset are shown for varying thresholds (T) in Figure 4.11. Removing topology constraints or plugging in flow without reinferring it perform so poorly that the curves do not fit in the plot. Using different optical flow initializations or removing the edge-sharpening does not change results significantly. While the optimization scheme only produces one segmentation and eliminates the trade-off

Table 4.2: Average endpoint error for training set of Middlebury dataset [3].

Video	Without Initialization	With Initialization
Dimetrodon	0.6864	0.3195
Grove2	1.0042	0.4917
Grove3	1.5564	1.0516
Hydrangea	0.5151	0.4302
RubberWhale	0.4103	0.4053
Urban2	6.1440	0.8631
Urban3	4.5784	0.8631
Venus	1.1397	0.6317

with thresholds, it still performs better than the current state-of-the-art algorithms.

■ 4.5.5 Optical Flow

While the presented approach infers a dense flow field, the purpose of the flow is not to be accurate on a sub-pixel level. Rather, the particular Gaussian process formulation was chosen for purposes of efficient inference in object tracking. Any realization of the flow will not be locally smooth because of the independence assumption in the composite flow field.

Regardless, for completeness, we show quantitative results on the Middlebury optical flow dataset [3] for the presented approach. We note that in the actual formulation, objects are assumed to have been detected. In the Middlebury dataset, because other algorithms do not have access to this segmentation data, we also do not use it. Rather, we treat the entire image as one layer and hope that the independent flow can capture the necessary discontinuities. Additionally, while our method is designed to track objects moving in natural scenes, the frames from [3] are all synthetically created or pictures in a laboratory setup. As such, their motion vectors are only a few pixels in magnitude as compared to the tens of pixels common in natural scenes. Estimated flow using our algorithm with the optical flow initialization of [82] and initialization with zero flow are evaluated quantitatively in Table 4.2 and shown in Figure 4.12. The flow field for each sequence is calculated as the mean flow over 100 samples, and each result is computed using the same set of parameters. We note that from Figure 4.12, it seems like the flow near the center of objects is estimated quite well while the flow near object boundaries is overly smoothed. This is a result of treating the entire scene as one layer. Additionally, we find that using optical flow as an initialization can greatly help the inference scheme for large regions of similar color.

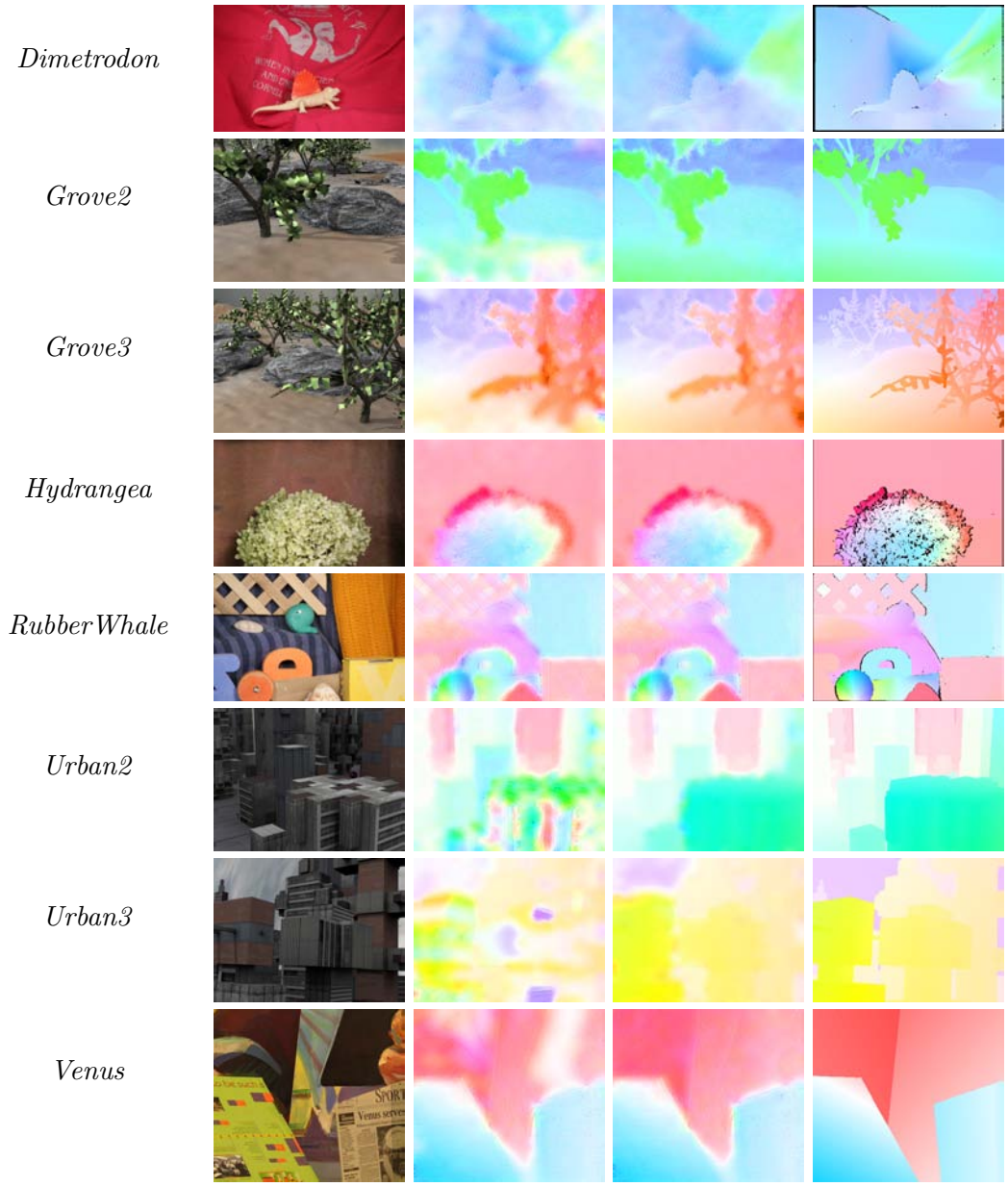


Figure 4.12: Inferred flow on the Middlebury dataset [3]. The first column shows the initial frame, the second column shows the inferred flow without any initialization, the third column shows the inferred flow with the optical flow of [82] as an initialization, and the fourth column is the ground truth flow. These results are obtained assuming the video is composed of a **single** layer, which would never be used in actual tracking.

4.6 Discussion

We presented a generative model for tracking objects in scenes that models occlusions, dynamic appearances, dynamic shapes, and enforces explicit topology constraints. Furthermore, we have shown that particle filtering where particles are propagated according to the posterior distribution does not require weight updates or particle resampling techniques. While this is typically difficult to do efficiently, the development of the PGIMH algorithm in Chapter 3 makes the particle propagation very efficient. The proposed approach has proven to outperform current tracking and video segmentation algorithms on the SegTrack dataset.

4.6.1 Future Work

We believe that the results presented in this chapter are quite encouraging. However, there is still much work to be done. MCMC sampling cannot typically be performed in real time because of the need to draw multiple samples for computing event statistics. In addition to the computation needed for multiple samples, the current framework still takes approximately 1 minute of processing time per sample per frame. While this is much faster than some previous probabilistic approaches (e.g., [113]), it is not nearly fast enough to process many videos of interest (such as surveillance videos with constant, streaming data). Multi-threaded approaches using a graphics processing unit (GPU) may be potential avenues to speed up the computation.

It is also not clear if the squared exponential (SE) kernel is the best class of covariance kernels for flow estimation. The "independent" flow, f, was introduced so that it could capture highly deformable and self-occluding objects that cannot be modeled by the smooth Gaussian process. Using other kernels (e.g., the Matérn class of kernels), or other parametric models of flow (e.g., piece-wise affine) may be more suitable.

Additionally, the current method requires objects to be already detected. As a proxy, results in this chapter have been obtained using user-specified strokes followed by Lazy Snapping [78]. It would be beneficial if the model capturing new objects entering the scene and had an explicit appearance model for them. A particle smoothing (as opposed to particle filtering) approach that considers both past and future data may aid such a model.

One possible approach to entertain new objects is to use a mixture model for the appearance model of each object, essentially treating each object as a bag of pixels. However, this would require an additional inference step in an already complicated model. Current probabilistic inference methods for mixture models with an unknown number of components are quite slow. In the following chapters, we develop alternative inference methods for Dirichlet process mixture models that improve convergence by orders of magnitude. We believe the combination of these approaches with the layered tracking presented in this chapter can be combined into a very interesting framework.

Chapter 5

Parallel Split-Merge MCMC for the DPMM

Computer vision problems are often difficult because of the sheer size of data to process. As such, developments in the probabilistic modeling community are often impeded in applications to computer vision. We believe one such development is in the Bayesian nonparametric models (e.g., Dirichlet process mixture models and their extensions), where inference that can handle the large number of observations in computer vision problems are still lacking. Motivated by this observation, we spend the next two chapters developing scalable inference algorithms for Dirichlet process mixture models (DPMMs) and their extensions. This work is a slight departure from the previous focus on computer vision and is a standalone contribution to the machine learning and probabilistic modeling community. In Chapter 7, we apply the developed inference algorithms to computer vision in intrinsic image decomposition.

Mixture models are a commonly used framework to model clusters of data in the machine learning community. In the recent decades, there has been considerable interest in extending classical finite mixture models to exploit non-parametric Bayesian statistics. Among other things, the elegant theory behind Dirichlet process mixture models (DPMM) has extended finite mixture models to include automatic model selection in clustering problems.

However, the rich representative power of the DP comes at a cost; unlike finite mixture models where the number of components is known *a priori* and can be fully instantiated, the infinite number of components cannot be represented. This has led to much work on developing methods of posterior inference, some of which are summarized in Section 2.9.2. One common approach is to perform Markov chain Monte Carlo sampling with Gibbs sampling, but this often leads to undesirable results because samplers that propose local changes exhibit poor convergence. Split and merge moves, first considered in DPs by [63], attempt to address these convergence issues, but currently cannot be parallelized and often waste precious computation on proposing a split or merge that is simply rejected by a Metropolis-Hastings ratio. Little work has been done in developing *scalable* split/merge moves for large datasets. Alternatively, approximate inference based on asymptotic limits such as [73] or variational approx-

Table 5.1: Capabilities of MCMC Sampling Algorithms in DPMMs

	CW	[60, 61]	[32, 96]	[27, 45, 63]	[64]	[83, 127]	Proposed Method
Exact Model	✓	·	✓	✓	✓	✓	✓
Splits & Merges	·	·	·	✓	✓	·	✓
Intra-cluster Parallel	·	·	·	·	·	✓	✓
Inter-cluster Parallel	·	✓	✓	·	·	·	✓
Non-conjugate Priors	✓	✓	✓	·	✓	·	✓

imations such as [10] can be used. Small-variance asymptotics perform inference on an altered model and variational algorithms do not have the limiting guarantees of MCMC methods. Both such methods may also suffer from similar convergence issues, but are particularly appealing for use in large datasets as they often lend themselves to parallelization.

In this chapter, we develop an MCMC sampling technique for mixture models that: (1) preserves limiting guarantees; (2) proposes splits and merges to improve convergence; (3) parallelizes and scales well for use in large datasets; and (4) is applicable to a conjugate and non-conjugate priors. To our knowledge, no current sampling algorithms satisfy all of these properties simultaneously. While this chapter mainly focuses on Dirichlet process mixture models, we note that similar methods can be applied for mixture models with other priors (finite Dirichlet distributions, Pitman-Yor Processes, etc.). An earlier version of this work was originally presented in [21].

■ 5.1 Related Work

We briefly review relevant related work in MCMC sampling for the Dirichlet process mixture model. The majority of DPMM samplers fit into one of two categories: collapsed-weight samplers that marginalize over the mixture weights or instantiated-weight samplers that explicitly represent them. Capabilities of current algorithms, which we now overview, are summarized in Table 5.1. A more detailed description of some related work is given in Section 2.9.2.

Collapsed-weight (CW) samplers using both conjugate (e.g., [16, 30, 85, 92, 126]) and non-conjugate (e.g., [86, 93]) priors sample the cluster labels iteratively one data point at a time without needing to approximate the infinite-length model. When a conjugate prior is used, one can also marginalize out cluster parameters. However, as noted by multiple authors (e.g., [27, 63, 79]), these methods often exhibit slow convergence. Additionally, due to the particular marginalization schemes, these samplers cannot be parallelized.

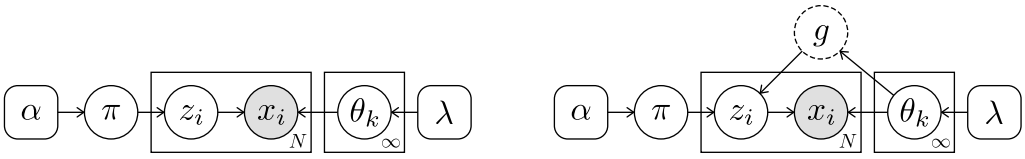
Instantiated-weight (IW) samplers explicitly represent cluster weights, typically using a finite approximation to the DP (e.g., [60, 61]). Recently, [32] and [96] have eliminated the need for this approximation; however, IW samplers still suffer from con-

vergence issues. If cluster parameters are marginalized, it can be very unlikely for a single point to start a new cluster. When cluster parameters are instantiated, samples of parameters from the prior are often a poor fit to the data. However, IW samplers are often useful because they can be parallelized across each data point conditioned on the weights and parameters. We refer to this type of algorithm as "inter-cluster parallelizable", since the cluster label for each point within a cluster can be sampled in parallel.

The recent works of [83] and [127] present an alternative parallelization scheme for CW samplers in DPMMs and HDPs. They observe that multiple clusters can be grouped into "super-clusters" and that each super-cluster can be sampled independently. We refer to this type of implementation as "intra-cluster parallelizable", since points in different super-clusters can be sampled in parallel, but points within a cluster cannot. This distinction is important as many problems of interest contain far more data points than clusters, and the greatest computational gain may come from inter-cluster parallelizable algorithms. Due to their particular construction, current algorithms group super-clusters solely based on the size of each super-cluster. We will show empirically that this can lead to slow convergence and demonstrate how data-dependent super-clusters improve upon these methods.

There has also been work on collapsed-weight sampling algorithms that consider larger moves to address convergence issues. Green and Richardson [45] present a reversible jump MCMC sampler that proposes splitting and merging components. While a general framework is presented, proposals are model-dependent and generic choices are not specified. Proposed splits are unlikely to fit the posterior since auxiliary variables governing the split cluster parameters and weights are proposed independent of the data. Jain and Neal [63, 64] construct a split by running multiple restricted Gibbs scans for a single cluster in conjugate and non-conjugate models. While each restricted scan improves the constructed split, it also increases the amount of computation needed. As such, it is not easy to determine how many restricted scans are needed. Dahl [27] proposes a split scheme for conjugate models by reassigning labels of a cluster sequentially. All current split samplers construct a proposed move to be used in a Metropolis-Hastings framework. If the split is rejected, considerable computation is wasted, and all information contained in learning the split is forgotten. In contrast, the proposed method of fitting sub-clusters iteratively learns likely split proposals with the auxiliary variables. Additionally, we show that split proposals can be computed in parallel, allowing for very efficient implementations.

Alternatives to classical sampling in DPMMs have been proposed. For example, Liang et al. [79] propose to sample from an augmented space of orderings and consistent partitions. While their algorithm works well when many components exist, they still find that using split-merge algorithms (e.g., [63, 27]) improve convergence speeds.



(a) DPMM Graphical Model (b) Augmented Super-Cluster

Figure 5.1: Graphical models for the DPMM and augmented super-cluster space. Auxiliary variables are dotted.

■ 5.2 Exact and Parallel Instantiated-Weight Samplers

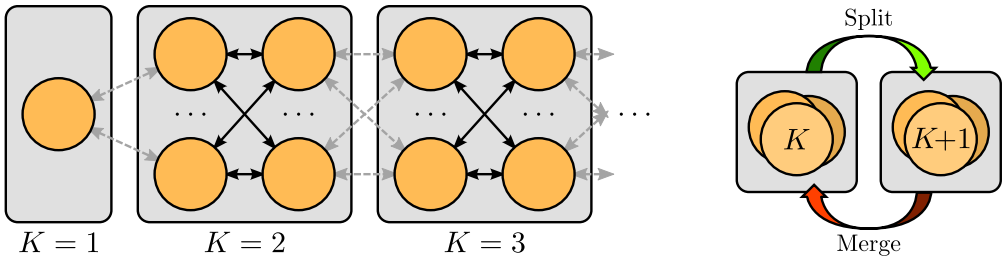
The Dirichlet process mixture model is summarized with previous sampling methods in Section 2.9.2. We reproduce the graphical model corresponding to a DPMM in Figure 5.1a for convenience. As stated previously, sampling from the DPMM is complicated by the infinite length mixture weights, π, and cluster parameters, θ, and often requires using an approximate finite model (such as the finite symmetric Dirichlet or truncated stick-breaking approximations).

We now present an alternative to the instantiated-weight samplers that does not require any finite model approximations. The *detailed balance* property of Definition 2.5.1 underlies most MCMC sampling algorithms. If one desires to sample from a target distribution, satisfying detailed balance for an *ergodic* Markov chain (Definition 2.5.8) guarantees that simulations of the chain will uniquely converge to the target distribution of interest. We now consider the atypical case of simulating from a *non-ergodic* chain with a transition distribution that satisfies detailed balance.

We define a *restricted* sampling algorithm as one that satisfies detailed balance (e.g., using the Metropolis-Hastings or Gibbs sampling algorithms) but that does not result in an ergodic chain. We note that without ergodicity, detailed balance does not imply uniqueness in, or convergence to the stationary distribution. However, multiple restricted samplers can be combined to form an ergodic chain, ensuring the uniqueness of the stationary distribution. In particular, we consider a sampler that is restricted to only sample labels belonging to non-empty clusters. Such a sampler is not ergodic because it cannot create new clusters. However, when mixed with a sampler that proposes splits, the resulting chain is ergodic and yields a valid sampler. A visualization of the state space is shown in Figure 5.2. We now consider a restricted *Gibbs* sampler. The coupled samplers that split or merge clusters is discussed in Sections 5.3-5.4.

■ 5.2.1 Restricted DPMM Gibbs Sampler with Super-Clusters

A property stemming from the definition of Dirichlet processes is that the measure for every finite partitioning of the measurable space is distributed according to a Dirichlet distribution [33]. While the DP places an infinite length prior on the labels, denoted with z, any realization of z will belong to a finite number of clusters. Supposing $z_i \in \{1, \cdots, K\}$, $\forall i$, we showed in Section 2.9.2 that the posterior distribution on the



(a) State Diagram of Restricted Sampler (b) State Diagram with Split/Merge

Figure 5.2: Visualizations of the state diagrams for the restricted sampler and the restricted sampler with split/merge moves. Each orange circle represents some configuration of z with K unique values. Dotted arrows correspond to the transitions in the state diagram that are not allowed due to the restricted sampling. This results in isolated islands of states, which the split and merge moves connect.

Dirichlet process follows the Dirichlet distribution of Equation (2.122). This corresponds directly to the following posterior distribution of mixture weights, π:

$$(\pi_1, \cdots, \pi_K, \pi_{K+1}) \sim \text{Dir}\left(\pi_1, \ldots, \pi_K, \tilde{\pi}_{K+1}; N_1, \ldots, N_K, \alpha\right), \tag{5.1}$$

where $N_k = \sum_i \mathbb{1}[z_i = k]$ is the number of points in cluster k, and $\pi_{K+1} = \sum_{k=K+1}^{\infty} \pi_k$ is the sum of all the weights associated with empty clusters. This relationship has previously been noted in the literature (cf. [116]). This leads to the following iterated restricted Gibbs sampler:

$$(\pi_1, \ldots, \pi_K, \tilde{\pi}_{K+1}) \sim \text{Dir}(N_1, \ldots, N_K, \alpha), \tag{5.2}$$

$$\theta_k \overset{\propto}{\sim} f_x(x_{\mathcal{I}_k}; \theta_k) f_\theta(\theta_k; \lambda), \qquad \forall k \in \{1, \ldots, K\}, \tag{5.3}$$

$$z_i \overset{\propto}{\sim} \sum_{k=1}^{K} \pi_k f_x(x_i; \theta_k) \mathbb{1}[z_i = k], \quad \forall i \in \{1, \ldots, N\}, \tag{5.4}$$

where $\overset{\propto}{\sim}$ denotes drawing a sample from a distribution proportional to the equation on the right and the subscript $\mathcal{I}_k \triangleq \{i; z_i = k\}$ denotes the set of indices with label $z_i = k$. The astute reader may realize that these distributions are quite similar to posterior inference in finite mixture models discussed in Section 2.8 or the finite approximations to the DP discussed in Section 2.9.2. The main difference is that in this formulation, the concentration parameter still corresponds to the probability of creating a new cluster, but z_i is never actually allowed to create a new cluster. These steps preserve detailed balance on the exact model, unlike the finite approximations. Additionally, we note that each of these steps can be parallelized and that this procedure applies to conjugate and non-conjugate priors because the mixture parameters are explicitly represented. When non-conjugate priors are used, any proposal that leaves the stationary distribution invariant can be used (cf. [93]).

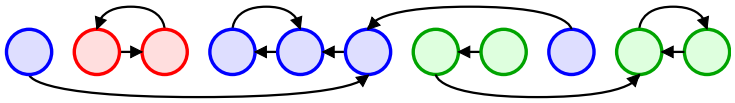


Figure 5.3: An illustration of the super-cluster grouping. Nodes represent clusters, arrows point to neighbors, and colors represent the implied super-clusters.

Deleted Clusters via Restricted Sampling

While restricted sampling explicitly disallows the creation of new clusters, it is possible for all the data points in a non-empty cluster to slowly move to another cluster. This is difficult to explicitly restrict since the label assignments are done in parallel. However, using similar arguments from Section 2.9.2 about the posterior on π, we can create a partitioning of the space to find the posterior on the cluster weights. Suppose that cluster e no longer has any data points associated with it (i.e., $N_e = 0$). The partitioning described by A in Equation (2.121), reproduced here as

$$(A_1, \ldots, A_e, \ldots, A_K, A_{K+1}) = (\delta_{\theta_1}, \ldots, \delta_{\theta_e}, \ldots, \delta_{\theta_K}, \Omega \setminus \{\cup_{k=1}^{K} \delta_{\theta_k}\}), \tag{5.5}$$

then has a corresponding posterior distribution that follows

$$(G(A_1), \ldots, G(A_e), \ldots, G(A_K), G(A_{K+1})) \sim \mathrm{Dir}(N_1, \ldots, N_e, \ldots, N_K, \alpha). \tag{5.6}$$

Because $N_e = 0$, there is no posterior measure on the resulting partition, $G(A_e)$, and we know that it must be assigned zero probability. Consequently, π_e will be zero, and the next step of restricted sampling will not assign any data points to cluster e. A cluster that is empty after an iteration of restricted sampling can therefore just be deleted, since no data points will ever be assigned to it again.

Data-Dependent Super-Clusters

Similar to previous super-cluster methods, we can also restrict each cluster to only consider moving to a subset of other clusters. The super-clusters of [83] and [127] are formed using a size-biased sampler. This can lead to slower convergence since clusters with similar data may not be in the same super-cluster. Because *any* restricted Gibbs sampler satisfies detailed balance, any algorithm that assigns finite probability to all super-cluster grouping will still satisfy detailed balance.

We therefore can augment the sample space with super-cluster groups, g, that group similar clusters together. The resulting graphical model is depicted in Figure 5.1b. Conditioned on g, Equation (5.4) is altered to only consider labels within the super-cluster that the data point currently belongs to. The super-cluster sampling procedure is described in Algorithm 5.1. Here, D denotes an arbitrary distance measure between probability distributions. In our experiments, we use the symmetric version of KL-divergence (J-divergence). When the J-divergence is difficult to calculate, any distance

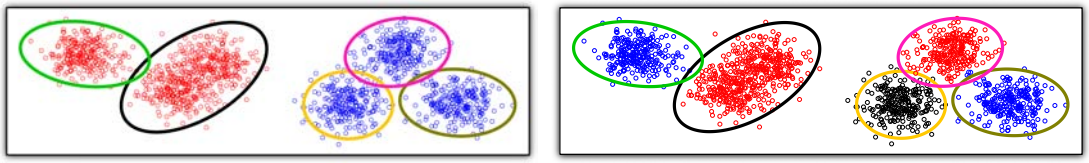


Figure 5.4: An illustration of the difference between data-dependent and data-independent super-clusters. Ellipses indicate cluster means and covariances. Color of data points indicate super-cluster membership. (left) super-clusters from the presented the algorithm. (right) super-clusters from [83].

measure can be substituted. For example, in the case of multinomial distributions, we use the J-divergence for the categorical distribution as a proxy. An illustration of the implied super-cluster grouping from the algorithm is shown in Figure 5.3 and a visualization of an actual super-cluster grouping is shown in Figure 5.4. Notice that the super-cluster groupings using [83] are essentially random while the data-dependent super-clusters from Algorithm 5.1 are grouped by similar data.

Algorithm 5.1 Sampling Super-clusters with Similar Cluster

1. Form the adjacency matrix, A, where $A_{k,m} = \exp[-D(f_x(\circ;\theta_k), f_x(\circ;\theta_m))]$.
2. For each cluster, k, sample a random neighbor k', according to

$$k' \overset{\propto}{\sim} \sum_m A_{k,m} \mathbb{I}[k' = m].$$

3. Form the groups of super-clusters, g, by finding the separate connected graphs.

■ 5.3 Randomized Split/Merge Moves

The preceding section showed that an exact MCMC sampling algorithm can be constructed by alternating between a restricted Gibbs sampler and split moves. In this section, we present a pair of data-independent split and merge proposals. Split moves that are constructed without knowledge of the data will typically be nonsensical, and we should not expect these split moves to perform well. Data-independent merge moves, in contrast to splits, can produce sensible proposals because the only way to merge two clusters is to simply put all the data into one new cluster. As we shall see in Section 5.4, these merge moves will have an important role in sampling from the true model.

Similar "randomized" moves were first considered in [63], where a proposed splitting of a cluster was generated by randomly assigning each data point to one of two new clusters with probability 0.5. We will use the symbols $\natural$, $\flat$, and $\sharp$ to denote cluster indices that will be involved in splits and merges, where $\natural, \flat, \sharp \in \{1, \ldots, K\}$. A proposed merge move will merge clusters $\flat$ and $\sharp$ into cluster $\natural$. This notation is motivated by musical

theory, where a flat accidental ($\flat$) combined with a sharp accidental ($\sharp$) results in a natural note ($\natural$).

We now discuss the randomized splits of [63]. A proposed split of cluster $\natural$ into clusters $\flat$ and $\sharp$, denoted $Q^K_{\text{rsplit-}\natural}$, is first selected with probability $q(Q^K_{\text{rsplit-}\natural})$ and then constructed according to

$$\hat{z}_i \sim q(\hat{z}_i|z, Q^K_{\text{rsplit-}\natural}) = \begin{cases} 1, & \hat{z}_i = z_i, z_i \neq \natural, \\ 0.5, & \hat{z}_i = \flat, z_i = \natural, \\ 0.5, & \hat{z}_i = \sharp, z_i = \natural. \end{cases} \tag{5.7}$$

The corresponding merge move, $Q^{K+1}_{\text{rmerge-}\flat\sharp}$, is selected with probability $q(Q^{K+1}_{\text{rmerge-}\flat\sharp})$ and simply puts all of the data associated with clusters $\flat$ and $\sharp$ into cluster $\natural$. We note that this procedure is slightly different from the original work of Jain and Neal [63], but will suffice for our purposes. The Hastings ratio for proposing a merge in this framework is

$$H^{\text{J.N.}}_{\text{merge-}\flat\sharp} = \frac{p(\hat{z})p(x|\hat{z})}{p(z)p(x|z)}\frac{q(z|\hat{z})}{q(\hat{z}|z)} = \frac{\Gamma(N_\flat + N_\sharp)}{\alpha\Gamma(N_\flat)\Gamma(N_\sharp)}\frac{p(x|\hat{z})}{p(x|z)}\frac{0.5^{N_\flat+N_\sharp}}{1}\frac{q(Q^{K-1}_{\text{rsplit-}\natural})}{q(Q^K_{\text{rmerge-}\flat\sharp})} \tag{5.8}$$

This randomized procedure has one minor flaw; corresponding partitions from a split will have high probability of being similar in size. This is in direct contrast to the Dirichlet process prior on partitions which explicitly favors larger clusters getting larger. We highlight the discrepancy between the proposal and the prior with an example. Consider the proposal of merging two clusters, m and n, where $N_m = 90$ and $N_n = 10$, with $\alpha = 1$. It would be preferable if the acceptance of the proposed merge was based on the data. However, this proposal has a corresponding Hastings ratio of

$$H^{\text{J.N.}}_{\text{merge-}\flat\sharp} = \frac{p(x|\hat{z})}{p(x|z)}\frac{q(Q^{K-1}_{\text{rsplit-}\natural})}{q(Q^K_{\text{rmerge-}\flat\sharp})}e^{-36.635},$$

which extremely favors rejecting the proposal, regardless of the data.

We present a slight improvement over this randomized split proposal that is still data-independent. As stated in Section 2.5.1, the closer the proposal distribution is to the target distribution, the better the convergence. We therefore propose a split by generating from the related Dirichlet-Categorical distribution

$$\hat{z}_{\mathcal{I}_\natural} \sim \text{DirCat}(\hat{z}_{\mathcal{I}_\natural}; \tfrac{\alpha}{2}, \tfrac{\alpha}{2}), \tag{5.9}$$

where $\mathcal{I}_\natural \triangleq \{i; z_i = \natural\}$ denotes the subset of indices that have label $\natural$. All other values of z_i remain unchanged. Again, the corresponding merge move simply combines clusters $\flat$ and $\sharp$ into one new cluster. This set of split/merge proposals results in the following

Hastings ratio for a proposed merge

$$H^{\text{rand}}_{\text{merge-}\flat\sharp} = \frac{\Gamma(N_\flat + N_\sharp)}{\alpha\Gamma(N_\flat)\Gamma(N_\sharp)} \frac{p(x|\hat{z})}{p(x|z)} \frac{\frac{\Gamma(\alpha)}{\Gamma(\alpha+N_\flat+N_\sharp)} \frac{\Gamma(\frac{\alpha}{2}+N_\flat)\Gamma(\frac{\alpha}{2}+N_\sharp)}{\Gamma(\frac{\alpha}{2})\Gamma(\frac{\alpha}{2})}}{1} \frac{q(Q^{K-1}_{\text{rsplit-}\natural})}{q(Q^{K}_{\text{rmerge-}\flat\sharp})}. \tag{5.10}$$

For the example considered above, this Hastings ratio equates to

$$H^{\text{rand}}_{\text{merge-}\flat\sharp} = \frac{p(x|\hat{z})}{p(x|z)} \frac{q(Q^{K-1}_{\text{rsplit-}\natural})}{q(Q^{K}_{\text{rmerge-}\flat\sharp})} e^{-2.363}.$$

This value essentially results in the proposed sampling being accepted if the data likelihood favors the merge and is a consequence of the Dirichlet-Categorical distribution better fitting the Dirichlet process prior.

Furthermore, generating a sample from Equation (5.9) can be parallelized by first sampling an auxiliary Dirichlet random variable, $\tilde{\pi}$ from

$$\tilde{\pi} \sim \text{Dir}(\tilde{\pi}; \tfrac{\alpha}{2}, \tfrac{\alpha}{2}), \tag{5.11}$$

followed by sampling each z_i in parallel according to

$$\hat{z}_i \sim \text{Cat}(z_i; \tilde{\pi}), \quad \forall i \in \{i; z_i = \natural\}. \tag{5.12}$$

The Hastings ratio for a random merge proposal can additionally be calculated efficiently from summary statistics (e.g., Equation (2.30)) since the two current clusters, $\flat$ and $\sharp$, are already instantiated. Thus, a merge can be proposed in constant time. The Hastings ratio for a random split proposal depends on the resulting split cluster assignments, $\hat{z}$. Consequently, a random split proposal requires linear time in the size of the cluster. We therefore choose $q(Q^{K}_{\text{rsplit-}\natural}) = 0.01 \times q(Q^{K+1}_{\text{rmerge-}\flat\sharp})$ so that the random split proposals do not take too much computation. We note that any value besides 0.01 could be used without much difference.

We reiterate that the previous restricted Gibbs sampling algorithm can be paired with *any* split/merge framework to produce an exact sampling algorithm for infinite mixture models. While the randomized split proposals described above do not typically fit the data (and are therefore likely to be rejected), the merge proposals work quite well in practice. In the following section, we describe a more sophisticated framework that excels at proposing likely splits.

■ 5.4 Parallel Split/Merge Moves via Sub-Clusters

We now develop efficient data-dependent split moves that are compatible with conjugate and non-conjugate priors and that can be parallelized. The general approach will be to augment the space with auxiliary variables that learn two *sub-clusters* within each cluster of the mixture model. These sub-clusters will contain a likely partition of the

data and will be used to propose splits. We note that in any augmented model, samples of the non-auxiliary variables can be obtained by drawing samples from the joint space and simply discarding any auxiliary values.

■ 5.4.1 Augmenting the Space with Auxiliary Variables

Each regular cluster is augmented with two explicit sub-clusters, herein referred to as the "left" and "right" sub-clusters. Each data point is then attributed with a sub-cluster label, $\overline{z}_i \in \{\ell, r\}$, indicating whether it is associated with the left or right sub-cluster. Additionally, each sub-cluster has an associated pair of weights, $\overline{\pi}_k = \{\overline{\pi}_{k\ell}, \overline{\pi}_{kr}\}$, and parameters, $\overline{\theta}_k = \{\overline{\theta}_{k\ell}, \overline{\theta}_{kr}\}$. These auxiliary variables are named in a similar fashion to their regular-cluster counterparts because of the similarities between sub-clusters and regular-clusters. One naïve choice for auxiliary parameter distributions is

$$p(\overline{\pi}_k) = \mathrm{Dir}(\overline{\pi}_{k\ell}, \overline{\pi}_{kr}; \tfrac{\alpha}{2}, \tfrac{\alpha}{2}), \tag{5.13}$$

$$p(\overline{\theta}_k) = f_\theta(\overline{\theta}_{k\ell}; \lambda) f_\theta(\overline{\theta}_{kr}; \lambda), \tag{5.14}$$

$$p(\overline{z}|\overline{\pi}, \overline{\theta}, x, z) = \prod_{k=1}^{K} \prod_{i \in \mathcal{I}_k} \sum_{h \in \{\ell, r\}} \frac{\overline{\pi}_{kh} f_x(x_i; \overline{\theta}_{kh})}{Z_i(x, z, \overline{\pi}_k, \overline{\theta}_k)} \mathbb{I}[\overline{z}_i = h], \tag{5.15}$$

where the normalization term $Z_i(x, z, \overline{\pi}_k, \overline{\theta}_k)$ is defined to be

$$Z_i(x, z, \overline{\pi}_k, \overline{\theta}_k) \triangleq \overline{\pi}_{k\ell} f_x(x_i; \overline{\theta}_{k\ell}) + \overline{\pi}_{kr} f_x(x_i; \overline{\theta}_{kr}), \tag{5.16}$$

and is constant with respect to $\overline{z}$. The corresponding graphical model is shown in Figure 5.5a. It would be advantageous if the form of the posterior for the auxiliary variables matched those of the regular-clusters in Equation (5.2)–(5.4). Unfortunately, because the normalization, Z_i, depends on $\overline{\pi}$ and $\overline{\theta}$, this choice of auxiliary distributions results in the following posterior distributions for $\overline{\pi}$ and $\overline{\theta}$

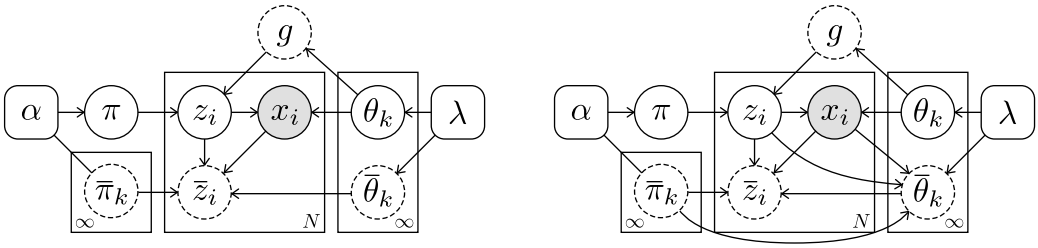
$$p(\overline{\pi}_k|\bullet) \propto \mathrm{Dir}(\overline{\pi}_{k\ell}, \overline{\pi}_{kr}; \tfrac{\alpha}{2}, \tfrac{\alpha}{2}) \prod_{i \in \mathcal{I}_k} Z_i(x, z, \overline{\pi}_k, \overline{\theta}_k)^{-1}, \tag{5.17}$$

$$p(\overline{\theta}_k|\bullet) \propto \prod_{h=\{\ell, r\}} f_\theta(\overline{\theta}_{kh}; \lambda) f_x(x_{\mathcal{I}_{kh}}; \overline{\theta}_{kh}) \prod_{i \in \mathcal{I}_k} Z_i(x, z, \overline{\pi}_k, \overline{\theta}_k)^{-1}, \tag{5.18}$$

where conditioning on $\bullet$ denotes conditioning on all other variables, and $\mathcal{I}_{kh} \triangleq \{i; z_i = k, \overline{z}_i = h\}$. These posterior distributions are quite different from the regular-cluster posterior distributions, and it is not clear how to sample from them efficiently. We note that this problem only arises in the auxiliary space where x generates the auxiliary label $\overline{z}$ (in contrast to the regular space, where z generates x).

Consequently, we alter the distribution over sub-cluster parameters to be

$$p(\overline{\theta}_k|x, z, \overline{\pi}) \propto f_\theta(\overline{\theta}_{k\ell}; \lambda) f_\theta(\overline{\theta}_{kr}; \lambda). \prod_{i \in \mathcal{I}_k} Z_i(x, z, \overline{\pi}_k, \overline{\theta}_k). \tag{5.19}$$



(a) Naïve Augmented Sub-Cluster Model (b) Correct Augmented Sub-Cluster Model

Figure 5.5: Graphical models for the augmented DPMMs. Auxiliary variables are dotted.

It is easily verified that this results in the following conditional posterior distributions

$$p(\overline{\pi}_k|\bullet) = \mathrm{Dir}(N_{k\ell} + \alpha/2, N_{kr} + \alpha/2), \qquad \forall k \in \{1, \dots, K\}, \tag{5.20}$$

$$p(\overline{\theta}_{kh}|\bullet) \propto f_x(x_{\mathcal{I}_{kh}}; \overline{\theta}_{kh}) f_\theta(\overline{\theta}_{kh}; \lambda), \qquad \forall k \in \{1, \dots, K\}, \forall h \in \{\ell, r\}, \tag{5.21}$$

$$p(\overline{z}_i|\bullet) \propto \sum\nolimits_{h \in \{\ell, r\}} \overline{\pi}_{z_i h} f_x(x_i; \overline{\theta}_{z_i h}) \mathbb{1}[\overline{z}_i = h], \quad \forall i \in \{1, \dots, N\}, \tag{5.22}$$

which essentially match the distributions for regular-cluster parameters in Equation (5.2)–(5.4). We note that the joint distribution over the augmented space cannot be expressed analytically as a result of only specifying Equation (5.19) up to a proportionality constant that depends on $\overline{\pi}$, x, and z. The corresponding graphical model is shown in Figure 5.5b. Additional details and derivations for this section can be found in Appendix B.

■ 5.4.2 Restricted Gibbs Sampling in Augmented Space

Restricted sampling in the augmented space can be performed in a similar fashion as before. One can draw a sample from the space of K regular clusters by sampling all the regular- and sub-cluster parameters conditioned on labels and data from Equations (5.2), (5.3), (5.20), and (5.21). Conditioned on these parameters, one can sample a regular-cluster label followed by a sub-cluster label for each data point from Equations (5.4) and (5.22). All of these steps can be computed in parallel. The procedure is summarized in Algorithm 5.2.

Algorithm 5.2 Restricted Sampling with Sub-Clusters

1. Sample π and $\overline{\pi}$ from Equations (5.2) and (5.20).
2. For each cluster k, sample θ_k and $\overline{\theta}_k$ from Equations (5.3) and (5.21).
3. For each index i, sample z_i and $\overline{z}_i$ from Equation (5.4) and (5.22).

The resulting inference from the restricted Gibbs sampling algorithm for a synthetic

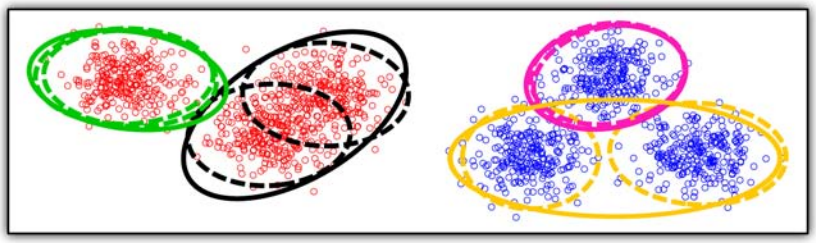


Figure 5.6: A visualization of the inferred sub- and super-clusters of the algorithm. Solid ellipses indicate regular-cluster means and covariances and dotted ellipsses indicate sub-cluster means and covariances. Color of data points indicate super-cluster membership.

Gaussian mixture model is shown in Figure 5.6. We have initialized the inference to have 4 clusters. Inferred regular-cluster parameters are illustrated with a solid ellipse and inferred sub-cluster parameters are illustrated with dotted ellipses of the same color. Because split moves have not been incorporated into the procedure yet, the result is not a valid sample from the posterior. This is indicated by the black and yellow clusters, which each contain two true clusters. However, the dotted ellipses show that the inferred sub-clusters correctly capture the information of interest in representing a likely split.

■ 5.4.3 Sub-Cluster Split Moves

We now exploit these auxiliary variables to propose likely splits. Similar to previous split/merge algorithms, we use a Metropolis-Hastings (MH) MCMC [51] method for proposed splits. A new set of random variables, $\{\hat{\pi}, \hat{\theta}, \hat{z}, \hat{\overline{\pi}}, \hat{\overline{\theta}}, \hat{\overline{z}}\}$ are proposed via some proposal distribution, q, and accepted with probability

$$\min[1, H] = \min\left[1, \frac{p(\hat{\pi}, \hat{z}, \hat{\theta}, x)p(\hat{\overline{\pi}}, \hat{\overline{\theta}}, \hat{\overline{z}}|x, \hat{z})}{p(\pi, z, \theta, x)p(\overline{\pi}, \overline{\theta}, \overline{z}|x, z)} \cdot \frac{q(\pi, z, \theta, \overline{\pi}, \overline{\theta}, \overline{z}|\hat{\pi}, \hat{z}, \hat{\theta}, \hat{\overline{\pi}}, \hat{\overline{\theta}}, \hat{\overline{z}})}{q(\hat{\pi}, \hat{z}, \hat{\theta}, \hat{\overline{\pi}}, \hat{\overline{\theta}}, \hat{\overline{z}}|\pi, z, \theta, \overline{\pi}, \overline{\theta}, \overline{z})}\right]. \quad (5.23)$$

Unfortunately, because the joint likelihood for the augmented space cannot be expressed analytically, the Hastings ratio for an arbitrary proposal distribution cannot be computed. We now discuss a very specific proposal distribution which results in a tractable Hastings ratio. A split or merge move, denoted by Q, is first selected at random. As we detail shortly, all possible splits and a large subset of all possible merges are considered at every iteration. A randomized proposal can be used instead when the number of clusters is large.

Conditioned on $Q = Q^K_{\text{split-}\natural}$, which splits cluster $\natural$ into clusters $\flat$ and $\sharp$, or $Q = Q^K_{\text{merge-}\flat\sharp}$, which merges clusters $\flat$ and $\sharp$ into cluster $\natural$, a new set of variables are sampled

with the following

$$
\begin{aligned}
Q &= Q^K_{\text{split-}\natural} & Q &= Q^K_{\text{merge-}\flat\sharp} \\
\hat{z} &= \text{split-}\natural(z, \overline{z}), & \hat{z} &= \text{merge-}\flat\sharp(z), && (5.24)\\
(\hat{\pi}_\flat, \hat{\pi}_\sharp) &= \pi_\natural \cdot (u_\flat, u_\sharp), \quad (u_\flat, u_\sharp) \sim \text{Dir}(\hat{N}_\flat, \hat{N}_\sharp), & \hat{\pi}_\natural &= \hat{\pi}_\flat + \hat{\pi}_\sharp, && (5.25)\\
(\hat{\theta}_\flat, \hat{\theta}_\sharp) &\sim q(\hat{\theta}_\flat, \hat{\theta}_\sharp | x, \hat{z}, \hat{\overline{z}}), & \hat{\theta}_\natural &\sim q(\hat{\theta}_\natural | x, \hat{z}, \hat{\overline{z}}), && (5.26)\\
\hat{\overline{v}}_\flat, \hat{\overline{v}}_\sharp &\sim p(\hat{\overline{v}}_\flat, \hat{\overline{v}}_\sharp | x, \hat{z}), & \hat{\overline{v}}_\natural &\sim p(\hat{\overline{v}}_\natural | x, \hat{z}). && (5.27)
\end{aligned}
$$

Here, $\overline{v}_k = \{\overline{\pi}_k, \overline{\theta}_k, \overline{z}_{\mathcal{I}_k}\}$ denotes the set of auxiliary variables for cluster k, the function split-$\natural(\circ)$ splits the labels of cluster $\natural$ *deterministically* based on the sub-cluster labels according to

$$
\hat{z}_i = \text{split-}\natural(z_i, \overline{z}_i) = \begin{cases} z_i, & z_i \neq \natural \\ \flat, & z_i = \natural, \overline{z}_i = \ell \\ \sharp, & z_i = \natural, \overline{z}_i = r \end{cases}. \tag{5.28}
$$

and merge-$\flat\sharp(\circ)$ merges the labels of clusters $\flat$ and $\sharp$ according to

$$
\hat{z}_i = \text{merge-}\flat\sharp(z_i) = \begin{cases} z_i, & z_i \neq \flat, z_i \neq \sharp \\ \natural & z_i = \flat \text{ or } z_i = \sharp \end{cases}. \tag{5.29}
$$

The proposal of cluster parameters in Equation (5.26) is written in a general form for compatibility with non-conjugate priors. If conjugate priors are used, Equation (5.26) should be sampled directly from the posterior distribution. Sampling auxiliary variables from Equation (5.27) will be discussed shortly. Assuming that this can be performed, we show in Appendix B that the resulting Hastings ratio for a split is

$$
\begin{aligned}
H^{\text{det}}_{\text{split-}\natural} &= \frac{q(Q^{K+1}_{\text{merge-}\flat\sharp})}{q(Q^K_{\text{split-}\natural})} \frac{\alpha q(\theta_\natural | x, z, \hat{z})}{\Gamma(N_\natural) f_\theta(\theta_\natural; \lambda) f_x(x_{\mathcal{I}_\natural}; \theta_\natural)} \prod_{k \in \{\flat, \sharp\}} \frac{\Gamma(\hat{N}_k) f_\theta(\hat{\theta}_k; \lambda) f_x(x_{\mathcal{I}_k}; \hat{\theta}_k)}{q(\hat{\theta}_k | x, z, \hat{z})} \\
&= \frac{q(Q^{K+1}_{\text{merge-}\flat\sharp})}{q(Q^K_{\text{split-}\natural})} \frac{\alpha}{\Gamma(N_\natural) f_x(x_{\mathcal{I}_\natural}; \lambda)} \prod_{k \in \{\flat, \sharp\}} \Gamma(\hat{N}_k) f_x(x_{\mathcal{I}_k}; \lambda).
\end{aligned} \tag{5.30}
$$

The first expression can be used for non-conjugate models, and the second expression can be used in conjugate models where new cluster parameters are sampled directly from the posterior distribution. We note that these expressions do not have any residual normalization terms and can be computed exactly, even though the joint distribution of the augmented space can not be expressed analytically.

As noted previously, we consider every possible split at each iteration, resulting in $q(Q^K_{\text{split-}\natural}) = 1$. When proposing merge moves, we construct $\lfloor K/2 \rfloor$ possible pairs by first generating a random permutation of the integers in $[1, K]$, and proposing to merge disjoint neighbors. For example, if the random permutation for $K = 7$ is $\{\overline{3\,1}\ \overline{7\,4}\ \overline{2\,6}\ 5\}$,

we will propose to merge clusters 3 and 1, clusters 7 and 4, and clusters 2 and 6. It is easily verified that the probability of proposing any specific merge is then $\frac{2\lfloor K/2 \rfloor}{K(K-1)}$. The probability of selecting any specific merge is approximately $\frac{2}{K}$, meaning that it is uncommon to select a pair of clusters that should actually be merged. We therefore use this proposal K times, resulting in $q(Q^K_{\text{merge-}\flat\sharp}) = \frac{2\lfloor K/2 \rfloor}{K-1}$.

Unfortunately, the Hastings ratio for a merge move is slightly more complicated. We discuss these complications following the explanation of sampling the auxiliary variables in the next section.

■ 5.4.4 Deferred Metropolis-Hastings Sampling

The preceding section showed that sampling a split according to Equations (5.24)–(5.27) results in an accurate MH framework. However, sampling the auxiliary variables from Equation (5.27) is not straightforward. This step is equivalent to sampling cluster parameters and labels for a 2-component mixture model, which is known to be difficult. One typically samples from this space using an MCMC procedure. In fact, that is precisely what the restricted Gibbs sampler is doing. We therefore sample from Equation (5.27) by running a restricted Gibbs sampler for each newly proposed sub-cluster until they have burned-in. We monitor the data-likelihood for cluster k, $\overline{\mathcal{L}}_k = f_x(x_{\mathcal{I}_{k\ell}}; \overline{\theta}_{k,\ell}) \cdot f_x(x_{\mathcal{I}_{kr}}; \overline{\theta}_{k,r})$ and declare burn-in once $\overline{\mathcal{L}}_k$ begins to oscillate.

Furthermore, due to the implicit marginalization of auxiliary variables, the restricted Gibbs sampler and split moves that act on clusters that were not recently split do not depend on the proposed auxiliary variables. As such, these proposals can be computed before the auxiliary variables are even proposed. The sampling of auxiliary variables of a recently split cluster are *deferred* to the restricted Gibbs sampler while the other sampling steps are run concurrently. Once a set of proposed sub-clusters have burned-in, the corresponding clusters can be proposed to split again.

■ 5.4.5 Merge Moves with Random Splits

The Hastings ratio for a merge depends on the proposed auxiliary variables for the reverse split. Since proposed splits are deterministic conditioned on the sub-cluster labels, the Hastings ratio will be zero if the proposed sub-cluster labels for a merge do not match those of the current clusters. We show in Appendix B.3 that as the number of data points grows, the acceptance ratio for a merge move quickly decays. With only 256 data points, the acceptance ratio for a merge proposal for 1000 trials in a 1D Gaussian mixture model did not exceed 10^{-16}. We therefore approximate all merges with an automatic rejection. Unfortunately, this can lead to slow convergence when too many clusters exist at the current iteration.

The sub-cluster split and merge moves excel at proposing good split moves but are poor at accepting merge moves. We remind the reader that the pair of randomized split and merge proposals presented in Section 5.3 is essentially the opposite; randomized splits are typically poor but the corresponding randomized merges often perform

well. Therefore, we mix the sub-cluster split sampler with the randomized split/merge sampler to achieve good splits *and* merges.

■ 5.5 Non-Deterministic Sub-Cluster Split Proposals

The preceding section presented a sampling algorithm that samples from the exact model without any approximations. While we have empirically noticed that this sampling algorithm works very well in practice, it is a bit unnerving that the sub-cluster merges are always rejected and that the randomized splits and merges are needed to achieve good convergence. In this section, we propose an alternative set of split/merge moves, where both the split and the merge are likely to be accepted. We note that this algorithm will require a slight approximation whereas the previous framework did not.

Instead of deterministically copying the sub-topic labels, we modify the proposal to *sample* a split. The sub-cluster statistics are used to propose a new cluster assignment by first constructing temporary parameters, $\{\tilde{\pi}_\flat, \tilde{\pi}_\sharp, \tilde{\theta}_\flat, \tilde{\theta}_\sharp\}$

$$(\tilde{\pi}_\flat, \tilde{\pi}_\sharp) = \pi_\natural \cdot (\overline{\pi}_{\natural\ell}, \overline{\pi}_{\natural r}), \quad (\tilde{\theta}_\flat, \tilde{\theta}_\sharp) = (\overline{\theta}_{\natural\ell}, \overline{\theta}_{\natural r}). \tag{5.31}$$

Conditioned on these temporary cluster parameters, new cluster assignments for topic $\natural$ are drawn from

$$q(\hat{z}|v, \overline{v}, Q^K_{\text{split-}\natural}) = \prod_{i \in \mathcal{I}_\natural} \sum_{k \in \{\flat, \sharp\}} \frac{\tilde{\pi}_k f_x(x_i; \tilde{\theta}_k) \mathbb{I}[\hat{z}_i = k]}{\tilde{\pi}_\flat f_x(x_i; \tilde{\theta}_\flat) + \tilde{\pi}_\sharp f_x(x_i; \tilde{\theta}_\sharp)}. \tag{5.32}$$

We note that a sample from this distribution is already drawn from the restricted Gibbs sampler described in Equation (5.22). Therefore, no additional computation is needed to sample from this distribution. If the split is rejected, the $\hat{z}$ is used as the next sample of the auxiliary $\overline{z}$ for cluster $\natural$.

The corresponding merge move combines topics $\flat$ and $\sharp$ into topic $\natural$ by deterministically performing

$$q(\hat{z}_i|v, Q^K_{\text{merge-}\flat\sharp}) = \mathbb{I}[\hat{z}_i = \natural], \quad \forall i \in \mathcal{I}_\flat \cup \mathcal{I}_\sharp. \tag{5.33}$$

Split and merge proposals for $\hat{\pi}$, $\hat{\theta}$, and $\hat{\overline{v}}$ follow the previous distributions of Equations (5.25)–(5.27) conditioned on $\hat{z}$.

The resulting Hastings ratio for this non-deterministic split only differs from Equation (5.30) by including the additional term from Equation (5.32) and can be expressed as

$$H^{\text{non-det}}_{\text{split-}\natural} = H^{\text{det}}_{\text{split-}\natural} \frac{1}{q(\hat{z}|v, \overline{v}, Q^K_{\text{split-}\natural})} \tag{5.34}$$

We record the probability $q(\hat{z}|v, \overline{v})$ when generating the auxiliary variables, and all remaining terms can be computed efficiently as before without iterating through the data.

The Hastings ratio for a merge is essentially the reciprocal of Equation (5.34). However, calculating the terms for a merge is slightly more problematic since the probability of the reverse split after a merge is proposed, $q(z|\hat{v}, \hat{\bar{v}}, Q^{K-1}_{\text{split-}\natural})$, depends on the inferred sub-cluster parameters, $\hat{\bar{v}} = \{\hat{\bar{\pi}}, \hat{\bar{\theta}}, \hat{\bar{z}}\}$. These proposed sub-topic parameters are not readily available due to the Deferred Metropolis-Hastings. Instead, we calculate the Hastings ratio by approximating the inferred sub-clusters with the two original clusters that are merging. In the limit, as the Markov chain has reached its stationary distribution, this assumption is quite accurate because of the similarity between regular-clusters and sub-clusters.

With this approximation, generating the reverse move that splits cluster l into m and n can be expressed as

$$q(z|\hat{v}, \hat{\bar{v}}, Q^{K-1}_{\text{split-}\natural}) \approx \prod_{i \in \mathcal{I}_\flat \cup \mathcal{I}_\sharp} \frac{\pi_{z_i} f_x(x_i; \theta_{z_i})}{\sum_{k \in \{\flat, \sharp\}} \pi_k f_x(x_i; \theta_k)}. \tag{5.35}$$

All the terms in this ratio are already calculated in the restricted Gibbs steps of Equation (5.4) in Algorithm 5.2. When aggregated properly, any merge can be proposed in constant time. We maintain a $K \times K$ matrix $\mathcal{L}$, where each element aggregates the following

$$\mathcal{L}_{kk} = \prod_{i \in \mathcal{I}_k} \pi_k f_x(x_i; \theta_k), \tag{5.36}$$

$$\mathcal{L}_{kl} = \prod_{i \in \mathcal{I}_k} \sum_{\kappa \in \{k,l\}} \pi_\kappa f_x(x_i; \theta_\kappa). \tag{5.37}$$

The reverse split move can then be approximated with

$$q(z|\hat{v}, \hat{\bar{v}}, Q^{K-1}_{\text{split-}\natural}) \approx \frac{\mathcal{L}_{\flat\flat} \mathcal{L}_{\sharp\sharp}}{\mathcal{L}_{\flat\sharp} \mathcal{L}_{\sharp\flat}}. \tag{5.38}$$

This concludes the discussion of non-deterministic split and merge proposals in DPMMs. Because merge proposals are now accepted with non-zero probability, incorporating randomized split/merge proposals is no longer needed.

■ 5.6 Experimental Results

In this section, we analyze the proposed sampling method. We compare the different split/merge proposals described in the preceding sections and compare the proposed methods to other popular MCMC sampling methods.

■ 5.6.1 Split/Merge Proposal Comparison

We begin by comparing the different split/merge proposals described in the preceding sections. We consider four algorithms: using deterministic and randomized proposals

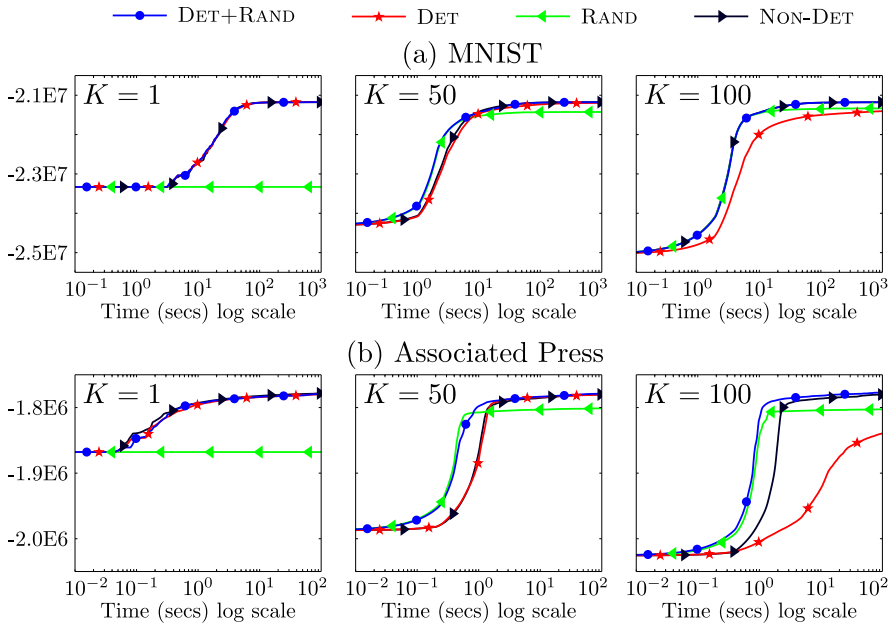


Figure 5.7: Log likelihood vs. computation time for various split/merge proposals on two datasets. Three different initializations with varying number of initial clusters are used for each algorithm.

(DET+RAND), using only deterministic proposals (DET), using only randomized proposals (RAND), and using the non-deterministic proposals (NON-DET). We test the methods on a Gaussian model with a Normal Inverse-Wishart prior on the MNIST dataset [76] by first running PCA on the 70,000 training and test images to 50 dimensions. We additionally test the algorithm on multinomial data with a Dirichlet prior on the Associated Press [9] (2,246 documents and 10,473 dimension dictionary). Results are shown in Figure 5.7 for each dataset with 1, 50, and 100 initial clusters. Each plot shows the average log likelihood for multiple sample paths obtained using 16 cores.

The plots show a few important points. The RAND algorithm does not typically propose useful splits, causing the log-likelihood to stay constant when initialized to a single cluster. Conversely, the DET algorithm does not accept merges, causing the log-likelihood to prematurely plateau when initialized to too many clusters. The DET+RAND and NON-DET methods both do not suffer from these issues since likely splits are proposed, and merges are accepted with non-negligible probability. Additionally, the NON-DET method seems to take slightly longer than DET+RAND. This increase in time is due to the extra computation required to aggregate statistics in $\mathcal{L}$ for the non-deterministic moves. For this reason, the remainder of this chapter will use the DET+RAND split proposals. However, as we shall see in Chapter 6, the non-deterministic proposals will still play an important role for extensions to hierarchical models.

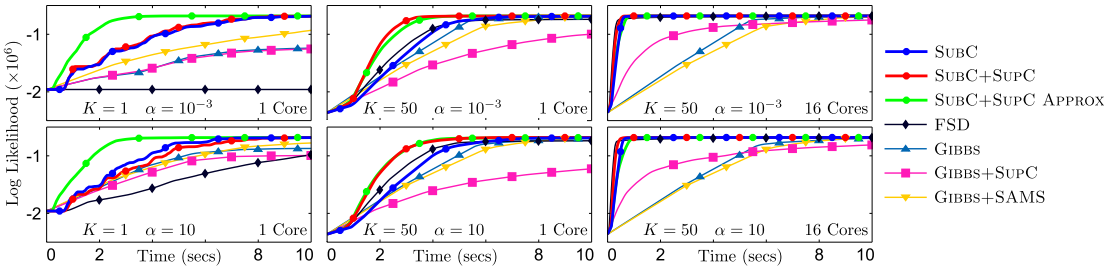


Figure 5.8: Results on synthetic data for various initial clusters K, concentration parameters α, and cores.

■ 5.6.2 Parallelizability and Sensitivity to Hyper-Parameters

Next, we compare the proposed algorithm with other MCMC sampling algorithms on synthetic data. We consider three different versions of the proposed algorithm: using sub-clusters with and without super-clusters (SubC and SubC+SupC) and an approximate method that does not wait for the convergence of sub-clusters to split (SubC+SupC Approx). We note that while we do not expect this last version to converge to the correct distribution, empirical results show that it is similar in average performance. We compare the proposed methods against four other methods: the finite symmetric Dirichlet approximate model (FSD) with 100 components, a Rao-Blackwellized Gibbs sampler (Gibbs), a Rao-Blackwellized version of the original super-cluster work of [83] (Gibbs+SupC), and the current state-of-the-art split/merge sampler [27] (Gibbs+SAMS). In our implementations, the concentration parameter is not resampled, though one could easily use a slice-sampling algorithm if desired.

We compare these algorithms on synthetic Gaussian data with a Normal Inverse-Wishart prior. 100,000 data points are simulated from ten 2D Gaussian clusters. The average log likelihood for multiple sample paths obtained using the algorithms without parallelization for different numbers of initial clusters K and concentration parameters α are shown in the first two columns of Figure 5.8. In this high data regime, α should have little effect on the resulting clusters. However, we find that the samplers without split/merge proposals (FSD, Gibbs, Gibbs+SC) perform very poorly when the initial number of clusters and the concentration parameter is small. We also find that the super-cluster method, Gibbs+SC, performs even worse than regular Gibbs sampling. This is likely due to the super-clusters not being grouped by similar data, which hinders convergence because data points cannot move between different super-clusters. In contrast, the proposed super-cluster method does not suffer from the same convergence problems, but is comparable to SubC because there are a small number of clusters. Finally, the approximate sub-cluster method has significant gains when only one initial cluster is used, but performs approximately the same with more initial clusters.

Next we consider parallelizing the algorithms using 16 cores in the last column of Figure 5.8. The four inter-cluster parallelizable algorithms, SubC, SubC+SupC,

SubC+SupC Approx, and FSD exhibit an order of magnitude speedup, while the the intra-cluster parallelizable algorithm Gibbs+SupC only has minor gains. As expected, parallelization does not aid the convergence of algorithms, only the speed at which they converge.

5.6.3 Real-World Datasets

We now show results on real data. We test a Gaussian model with a Normal Inverse-Wishart prior on the MNIST dataset [76] by first running PCA on the 70,000 training and test images to 50 dimensions. Results on the MNIST dataset are shown in Figure 5.9a. We additionally test the algorithm on multinomial data with a Dirichlet prior on the following datasets: Associated Press [9] (2,246 documents and 10,473 dimension dictionary), Enron Emails [2] (39,861 documents and 28,102 dimension dictionary), New York Times articles [2] (300,000 documents and 102,660 dimension dictionary), and PubMed abstracts [2] (8,200,000 documents and 141,043 dimension dictionary). Results are shown in Figure 5.9b-e. In contrast to HDP models, each document is treated as a *single* draw from a multinomial distribution. We note that on the PubMed dataset, we had to increase the approximation of FSD to 500 components after observing that SubC inferred approximately 400 clusters. On real data, it is clearly evident that the other algorithms have issues with convergence. In fact, in the allotted time, no algorithms besides the proposed methods converge to the same log likelihood with the two different initializations on the larger datasets. The presented sub-cluster methods converge faster and to a better configuration as compared to the other algorithms.

On the small, Associated Press dataset, the proposed methods actually perform slightly worse than the Gibbs methods. Approximately 20 clusters are inferred for this dataset, resulting in approximately 100 observations for each cluster in a 10,473-dimensional space. In these small data regimes, it is important to marginalize over as many variables as possible. We believe that because the Gibbs methods marginalize over the cluster parameters and weights, they achieve better performance as compared to the sub-cluster methods and FSD which explicitly instantiate them. This is not an issue with larger datasets.

5.7 Discussion

This chapter develops a new MCMC sampling algorithm for Dirichlet process mixture models called the DP Sub-Cluster algorithm. The DP Sub-Cluster algorithm is easily parallelized to scale to large datasets, and exhibits better convergence through the proposed sub-cluster split and merge moves. In fact, the algorithm converges to a better configuration than other algorithms, and does so approximately 10–10^3 times faster. While the focus of discussion in this chapter was on DPMMs, we believe that the framework used in developing the DP Sub-Cluster algorithm is widely applicable to the general finite or infinite mixture model.

The deterministic split move and randomized split/merge proposals are paired with

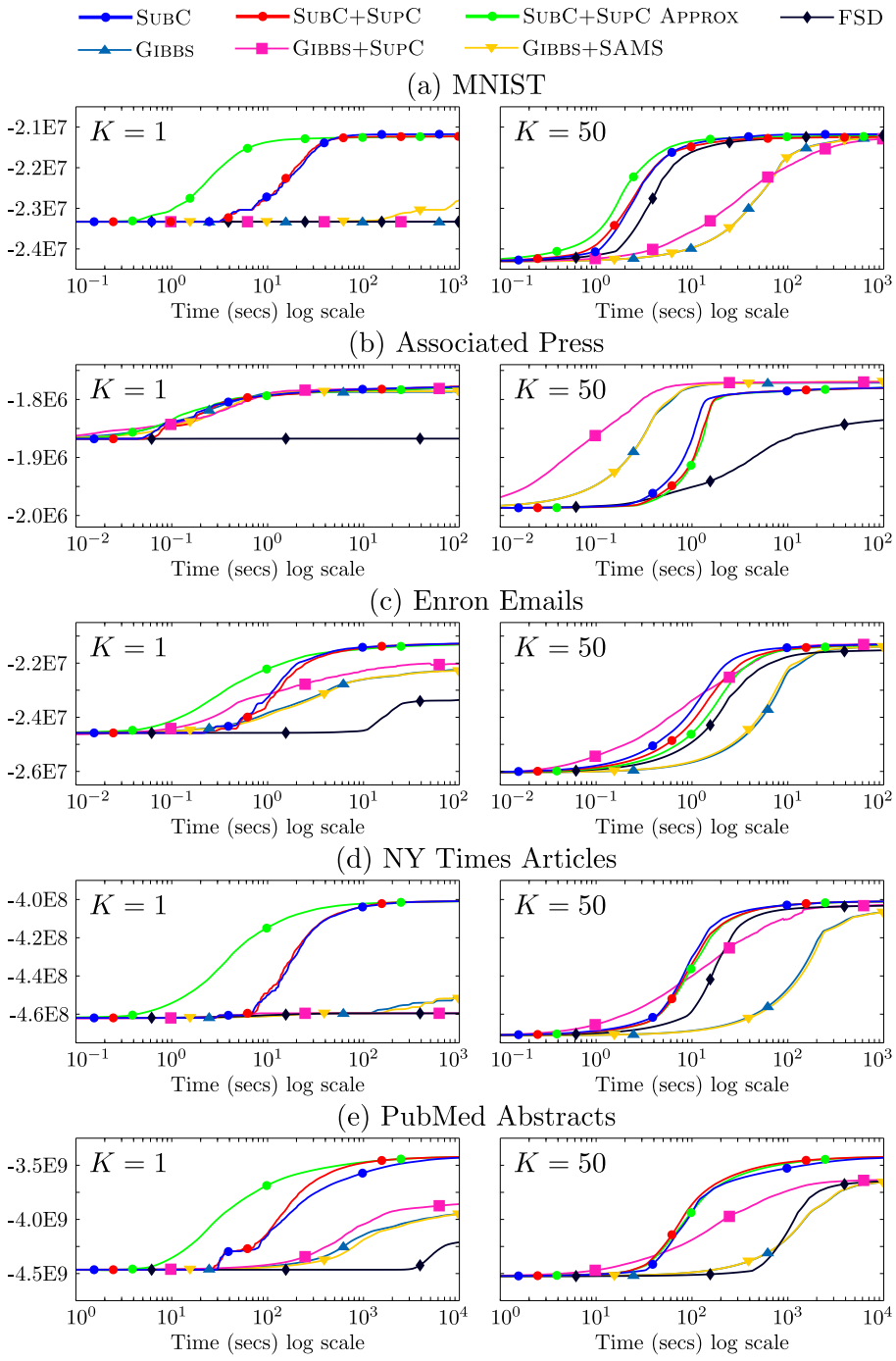


Figure 5.9: Results on real-world Gaussian and Multinomial data. Each figure plots log likelihood vs. computation time. All parallel algorithms use 16 cores. The first column is initialized to one cluster and the second column is initialized to 50 clusters.

a restricted Gibbs sampling algorithm to produce an MCMC algorithm that does not require any approximations. However, the non-deterministic split and merge moves required a slight approximation due to the deferred Metropolis-Hastings. In particular, because the sub-cluster assignments are not readily available after a proposed split or merge, the sub-clusters were estimated using the original clusters. While this approximation was argued to be fairly accurate in the limit of reaching the stationary distribution, the theory would benefit from a more rigorous analysis.

Additionally, we note that small clusters, which often appear from the collapsed Gibbs sampling algorithms, tend to occur less frequently in the DP Sub-Cluster algorithm. This is likely due to the implicit bias towards proposing large split moves from the instantiated sub-clusters paired with the restricted Gibbs sampling algorithm that does not allow the creation of new clusters. This is in stark contrast to the collapsed Gibbs sampling algorithms, which can start a cluster with a single data point at any iteration. As shown in [91], these small clusters are in the typical set of DPMMs, and should be expected.

We also believe the method of deferred Metropolis-Hastings, where the auxiliary variables are deferred to the restricted Gibbs sampling algorithm, is an interesting idea that other auxiliary variable methods may benefit from. This type of proposal could also benefit from a more detailed derivation.

While we have done our best to compare many of the current MCMC sampling algorithms, one additional interesting experiment that could be performed is comparing the DP Sub-Cluster algorithm to variational inference methods. In particular, variational methods often benefit from being highly parallelizable. Moreover, recent variational methods (e.g., [57]) also exploit split/merge moves, but are simpler because they do not need to satisfy detailed balance.

We believe one promising direction is the extension of the DP Sub-Cluster algorithm to models that use Dirichlet processes as a component in a larger model. For example, the Hierarchical Dirichlet Process [116], the Hierarchical Dirichlet Process Hidden Markov Model [34, 116], and the Dependent Dirichlet Process [81] could all greatly benefit from scalable MCMC inference methods. In the next chapter, we demonstrate how the DP Sub-Cluster algorithm can extend to the HDP, with the intent to show that many models can take advantage of the work presented in this chapter.

Chapter 6

Parallel Split-Merge MCMC for the HDP

The previous chapter developed an efficient sampling scheme for Dirichlet process mixture models based on sub-clusters. In this chapter, we show how the sub-cluster method can be extended to hierarchical models, like the Hierarchical Dirichlet process (HDP) [116]. The HDP models groups of data with shared cluster statistics and has been used in many statistical learning applications including document analysis [116], object categorization [112], and as a prior for hidden Markov models [34].

Similar to DPMMs, the proposed method for HPDs is extremely parallelizable and does not require any finite model approximations. As will be shown, considerable work is needed for this extension because of additional latent variables and overlapping distributions that are difficult to disambiguate.

This chapter will follow in a similar fashion as Chapter 5. We first consider a restricted Gibbs sampling algorithm that is exact and can be easily parallelized. We then introduce the extension of the auxiliary sub-clusters in the HPDs, and show how they can be incorporated in the restricted sampler. The sub-clusters are used to propose large split and merge moves that change the labels of all observations. Finally, we test the algorithm on synthetic and real-world data and observe improved convergence properties as compared to other sampling methods. As a by-product of our experiments, we have found that cross-validation metrics do not accurately gauge convergence of MCMC procedures in HDPs. We propose an alternative metric that empirically shows that current sampling methods converge extremely slowly.

6.1 Related Work

The seminal work of Teh et al. [116] developed the HDP along with the Chinese Restaurant Franchise and the Direct Assignment sampling algorithms. These methods are reviewed in Section 2.9.3. Since then, there has been a considerable effort on alternative inference methods for HDPs with a large emphasis on scalable, variational methods.

Unfortunately, unlike Dirichlet process mixture models, there has not been much work in developing split/merge MCMC algorithms for HDP models. To our knowledge,

the only work that attempts to use splits and merges in an MCMC sampling framework is [122], which extends the Sequentially Allocated Merge-Split (SAMS) algorithm of [27] designed for DPMMs. Their preliminary results show that the addition of splits and merges has little impact on cross-validation metrics, sometimes even degrading performance. While this may result from a misfit between the HDP model and the data, our empirical results suggest that it is likely due to properties of the specific sampler instead. Additionally, the SAMS algorithm cannot be parallelized, and is therefore only tested on a corpus with up to 263K words. In contrast, we scale our algorithm to test on 100M words.

Alternatively, approximate inference methods such as the variational algorithms of [10, 15, 74, 117] can be used. Variational algorithms do not have the limiting guarantees of MCMC methods and may also suffer from similar convergence issues, but are particularly appealing for use in large datasets as they often lend themselves to parallelization. Recent work in Bayesian non-parametrics (e.g., [15, 58, 57]) has addressed convergence issues in variational approximations with split/merge moves being applied to HDPs, DPs, and beta process hidden Markov models. Consequently, the results of the HDP-SAMS sampler should not be discouraging; we expect splits and merges to help convergence in MCMC sampling for HDPs much like they have in variational inference and in DPMMs.

There has also been work on parallel sampling algorithms for HDPs. Fox et al. [34] approximate the highest level DP with a finite symmetric Dirichlet distribution of order L. This method generalizes the work of Ishwaran and Zarepour [61] on DPs to HDPs. While iterations of this finite approximation can be easily parallelized, setting a hard model order ruins the non-parametric nature of the HDP since the model no longer grows with the data. Furthermore, at each iteration, the sampler must consider each data point as being associated with each of the L clusters. Assuming N data points, each iteration then takes $O(NL)$ complexity. Since the approximation improves as the L increases, this method slows for good approximations (large L).

The parallelization scheme of Williamson et al. [127] for DPMMs was also extended to HDPs. However, Gal and Ghahramani [38] have since shown that the parallelization does not scale well. Furthermore, as evident from the experiments in Chapter 5, this super-cluster method often converges slower that Gibbs sampling because of the restriction on sampling, even with the added benefit of parallelization.

Lastly, Newman et al. [94] present another parallel approximation scheme that explicitly partitions and assigns the data across multiple processors. Each processor then independently runs a Gibbs sampler on its assigned data. When complete, the processors resynchronize sufficient statistics of the clustered data and the process is repeated. While this method has shown to perform well on cross-validation techniques, no limiting guarantees or bounds can be given.

The presented approach differs from previous methods by achieving a linear parallelization speed-up with the number of processors while also proposing large split and merge moves in parallel. We show that when the observed mixture model has am-

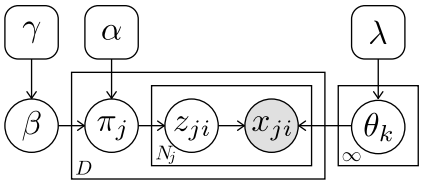


Figure 6.1: The Hierarchical Dirichlet process graphical model.

biguous, overlapping distributions (as is the case in topic modeling), simply splitting a cluster into two or merging two clusters into one may not greatly affect convergence. However, by incorporating larger, "global" moves, one can improve inference.

6.2 Hierarchical Dirichlet Processes

A summary of the HDP model is presented in Section 2.9.3. We reproduce the graphical model in Figure 6.1 and the notation here for convenience. Because of their prolific use in topic modeling, we will refer to the model variables with their topic modeling names: β are the corpus-level, global topic proportions, π_j are the topic proportions for document j, z_{ji} is the topic assignment for the i^{th} word in document j, x_{ji} is i^{th} word in document j, and θ_k are the parameters for the word distribution of topic k.

A sample from the HDP model can be drawn according to:

$$\beta \sim \text{GEM}(1, \gamma), \tag{6.1}$$

$$\pi_j \sim \text{DP}(\alpha, \beta), \qquad \forall j \in \{1, \dots, D\}, \tag{6.2}$$

$$\theta_k \sim f_\theta(\theta_k; \lambda), \qquad \forall k \in \{1, 2, \dots\}, \tag{6.3}$$

$$z_{ji} \sim \text{Cat}(\pi_j), \qquad \forall j \in \{1, \dots, D\},\ \forall i \in \{1, \dots, N_j\}, \tag{6.4}$$

$$x_{ji} \sim f_x(x_{ji}; \theta_{z_{ji}}), \qquad \forall j \in \{1, \dots, D\},\ \forall i \in \{1, \dots, N_j\}, \tag{6.5}$$

where f_x and f_θ denote some specific form of likelihood and prior distributions.

Alternatively, one can generate a sample from the HDP using the Chinese Restaurant Franchise (CRF). In the CRF, each document is a restaurant, each word is a customer, and cluster parameters are dishes served to tables. A customer enters a restaurant and sits at a table with probability proportional to the number of customers already at a table, or sits at a new table with probability α. New tables are then assigned a particular dish with probability proportional to the number of tables already serving that dish, or a new dish with probability γ. It can be shown that simulating a CRF is equivalent to sampling from Equations (6.1)–(6.5).

We adopt the notation of [116] for the CRF. The number of tables in restaurant j serving dish k is denoted m_{jk}, and the number of customers in restaurant j at table t eating dish k is n_{jtk}. Marginal counts are represented with dots. For example, $n_{j\cdot\cdot}$ and $m_{j\cdot}$ represent the number of customers and dishes, respectively, in restaurant j.

6.3 Restricted Parallel Sampling in HDPs

Similar to the work on DPMMs, the restricted Gibbs sampling algorithms acts on the fully instantiated model, but is restricted to the current non-empty clusters. More precisely, the restricted sampler is not allowed to create new clusters. This allows sampling without needing to instantiate an infinite number of β, π, or θ. Following the observation in Chapter 5 that super-clusters do not significantly affect results, we omit their inclusion in the augmented HDP model for simplicity.

The main difference between the HDP restricted Gibbs sampling algorithm and the DPMM counterpart is the inclusion of β. While the posterior $p(\beta|z)$ is not known in closed form, the posterior $p(\beta|m)$ is known. Antoniak [1] showed that the posterior of m_{jk} has the following distribution

$$p(m_{jk}|\beta, z) = \frac{\Gamma(\alpha\beta_k)}{\Gamma(\alpha\beta_k + n_{j\cdot k})} s(n_{j\cdot k}, m_{jk})(\alpha\beta_k)^{m_{jk}}, \tag{6.6}$$

where $s(n, m)$ are unsigned Stirling numbers of the first kind. Assuming that the current topic assignments fall into K topics (i.e., $z_{ji} \in \{1, \ldots, K\}$), it was shown in [34, 116] that the posterior distribution on the global-level and document-level proportions can then be expressed as

$$p(\beta|m) = \mathrm{Dir}(\beta_1, \ldots, \beta_{K+1}; m_{\cdot 1}, \ldots, m_{\cdot K}, \gamma), \tag{6.7}$$

$$p(\pi_j|\beta, z) = \mathrm{Dir}(\pi_{j1}, \ldots \pi_{j(K+1)}; \alpha\beta_1 + n_{j\cdot 1}, \ldots, \alpha\beta_K + n_{j\cdot K}, \alpha\beta_{K+1}). \tag{6.8}$$

We note that β and π have slightly changed from Equations (6.1)–(6.2), and are now defined over explicit partitions of the space. Also, instead of being infinite in length, they are $(K+1)$-length vectors where the last components, β_{K+1} and $\pi_{j(K+1)}$, aggregate the weight over all empty topics.

Topic parameters can be sampled according to

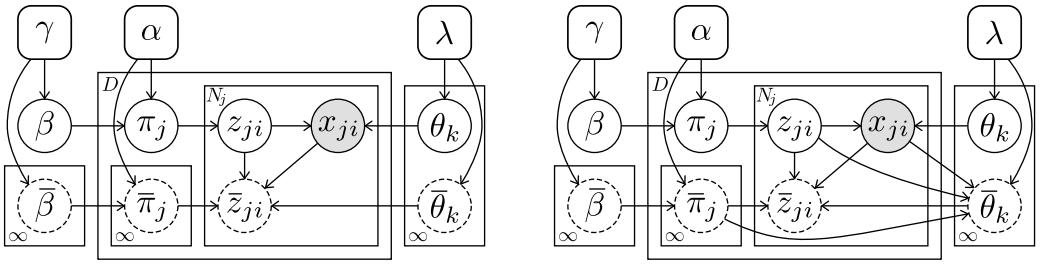
$$p(\theta_k|x, z) \propto f_x(x_{\mathcal{I}_k}; \theta_k) f_\theta(\theta_k; \lambda). \tag{6.9}$$

Similar to the DPMM case, $\mathcal{I}_k \triangleq \{ji; z_{ji} = k\}$ denotes the subset of data assigned to cluster k across all documents. If conjugate priors are used, Equation (6.9) stays in the same family of parametric distributions as $f_\theta(\theta; \lambda)$ (see Section 2.2).

The last step in the restricted sampler is to sample the topic assignments, z. This posterior distribution is a categorical distribution, *restricted* to the current non-empty clusters.

$$p(z_{ji}|x, \pi_j, \theta) \propto \sum_{k=1}^{K} \pi_{jk} f_x(x_{ji}; \theta_k) \mathbb{1}[z_{ji} = k]. \tag{6.10}$$

This concludes the extension of the DPMM Sub-cluster algorithm to HDPs. We note that sampling from any of Equations (6.6)–(6.10) can be done in parallel. Furthermore, similar to the DPMM algorithm, these distributions are quite similar to the finite



(a) Naïve Augmented Sub-Cluster Model (b) Correct Augmented Sub-Cluster Model

Figure 6.2: Augmented sub-topic HDP graphical models. Hyper-parameters are omitted and auxiliary variables are dotted.

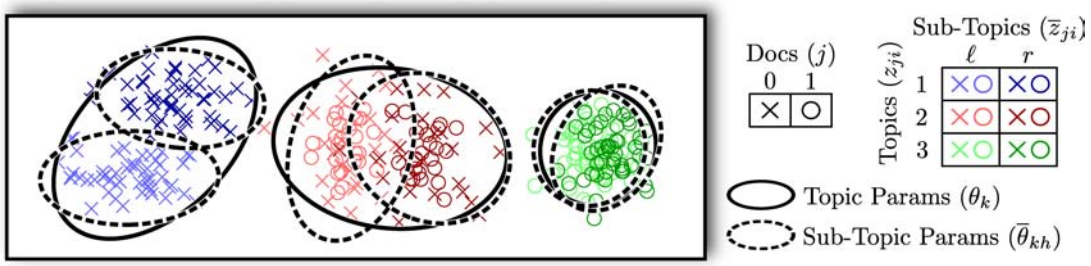


Figure 6.3: Visualization of augmented sample space.

approximations of HDPs used in [34]. The main differences are in the distribution of β since no approximation is used, and the sampling of z, which is explicitly restricted to non-empty clusters. Unlike the finite approximation, this sampler is guaranteed to converge to the correct target distribution without approximations when combined with any split proposal.

■ 6.4 Sub-Topic Fitting

In this section we extend the augmented sub-cluster model in DPMMs to a sub-topic model for HDPs. We reiterate that the goal of fitting sub-topics is to find a two-component mixture model that potentially corresponds to a likely split of the data.

For each topic, k, we fit two sub-topics, $k\ell$ and kr, referred to as the "left" and "right" sub-topics. Each topic is augmented with auxiliary variables for global sub-topic proportions $\overline{\beta}_k = \{\overline{\beta}_{k\ell}, \overline{\beta}_{kr}\}$, document-level sub-topic proportions $\overline{\pi}_{jk} = \{\overline{\pi}_{jk\ell}, \overline{\pi}_{jkr}\}$, and sub-topic parameters $\overline{\theta}_k = \{\overline{\theta}_{k\ell}, \overline{\theta}_{kr}\}$. Furthermore, a sub-topic assignment, $\overline{z}_{ji} \in \{\ell, r\}$ is associated with each word, x_{ji}. The augmented latent space is summarized in Figure 6.2a. Figure 6.3 visualizes these latent variables. Similar to the DP Sub-Cluster

method, we adopt the following auxiliary distributions

$$p(\overline{\beta}_k) = \mathrm{Dir}(\overline{\beta}_{k\ell}, \overline{\beta}_{kr}; \gamma, \gamma), \tag{6.11}$$

$$p(\overline{\pi}_{jk}|\overline{\beta}_k) = \mathrm{Dir}(\overline{\pi}_{jk\ell}, \overline{\pi}_{jkr}; \alpha\overline{\beta}_{k\ell}, \alpha\overline{\beta}_{kr}), \tag{6.12}$$

$$p(\overline{\theta}_k|\overline{\pi}, z, x) = \prod_{h\in\{\ell,r\}} f_\theta(\overline{\theta}_{kh}; \lambda) \prod_{ji\in\mathcal{I}_k} Z_{ji}(\overline{\pi}, \overline{\theta}, z, x), \tag{6.13}$$

$$p(\overline{z}|\overline{\pi}, \overline{\theta}, z, x) = \prod_{k=1}^{K} \prod_{ji\in\mathcal{I}_k} \frac{\overline{\pi}_{jk\overline{z}_{ji}} f_x(x_{ji}; \overline{\theta}_{k\overline{z}_{ji}})}{Z_{ji}(\overline{\pi}, \overline{\theta}, z, x)}, \tag{6.14}$$

$$Z_{ji}(\overline{\pi}, \overline{\theta}, z, x) \triangleq \sum_{h\in\{\ell,r\}} \overline{\pi}_{jz_{ji}s} f_x(x_{ji}; \overline{\theta}_{z_{ji}h}). \tag{6.15}$$

To be correct, Figure 6.2a should have edges pointing from $\overline{\pi}$, z, and x to $\overline{\theta}$ as shown in Figure 6.2b. However, we find Figure 6.2a easier to interpret. It can be shown that the auxiliary distributions of Equations (6.11)–(6.15) result in the following posteriors:

$$p(\overline{m}_{jkh}|\bullet) = \frac{\Gamma(\alpha\overline{\beta}_{kh})}{\Gamma(\alpha\overline{\beta}_{kh} + \overline{n}_{j\cdot kh})} s(\overline{n}_{j\cdot kh}, \overline{m}_{jkh})(\alpha\overline{\beta}_{kh})^{\overline{m}_{jkh}}, \tag{6.16}$$

$$p(\overline{\beta}_k|\bullet) = \mathrm{Dir}(\gamma + \overline{m}_{\cdot k\ell}, \gamma + \overline{m}_{\cdot kr}), \tag{6.17}$$

$$p(\overline{\pi}_{jk}|\bullet) = \mathrm{Dir}(\alpha\overline{\beta}_{k\ell} + \overline{n}_{j\cdot k\ell}, \alpha\overline{\beta}_{kr} + \overline{n}_{j\cdot kr}), \tag{6.18}$$

$$p(\overline{\theta}_{kh}|\bullet) \propto f_x(x_{\mathcal{I}_{kh}}; \overline{\theta}_{kh}) f_\theta(\overline{\theta}_{kh}; \lambda), \tag{6.19}$$

$$p(\overline{z}_{ji}|\bullet) \propto \overline{\pi}_{jz_{ji}\overline{z}_{ji}} f_x(x_{ji}; \overline{\theta}_{z_{ji}\overline{z}_{ji}}), \tag{6.20}$$

where $\bullet$ denotes all other variables and $\mathcal{I}_{kh} \triangleq \{ji; z_{ji} = k, \overline{z}_{ji} = h\}$. Notice the similarity between these equations and Equations (6.6)–(6.10). Since the auxiliary variables are generated conditioned on the main variables, posterior inference can be performed by interleaving the sampling of Equations (6.6)–(6.10) and Equations (6.16)–(6.20). Furthermore, all of these steps can all be computed in parallel because the model is fully instantiated.

■ 6.5 Sub-Topic Split/Merge Moves

The proposed split/merge sampling algorithm for HDPs is also quite similar to DPMMs; we use a Metropolis-Hastings algorithm that proposes splits or merges and accepts them with some probability. The steps of the framework are summarized in Algorithm 6.1. In the following sections, we present two different methods for the proposal of new topic assignments in Step 2.

Most of these steps are identical to the DPMM algorithm. For example, $\hat{\pi}$ can be sampled from its posterior expressed in Equation (6.8), and $\hat{\theta}$ can be sampled directly from its posterior assuming conjugate priors. Similarly, sampling the auxiliary variables,

Algorithm 6.1 HDP Split-Merge Framework

1. Propose a cluster to split or a pair of clusters to merge.
2. Propose new topic assignment from the inferred sub-topics: $\hat{z} \sim q(\hat{z}|v, \overline{v})$.
3. Propose new global topic proportions: $\hat{\beta} \sim q(\hat{\beta}|\hat{z})$.
4. Propose new document topic proportions: $\hat{\pi} \sim q(\hat{\pi}|\hat{\beta}, \hat{z})$.
5. Propose new topic parameters: $\hat{\theta}_k \sim q(\hat{\theta}_k|x, z)$.
6. Defer the proposal of auxiliary variables.
7. Accept/reject the proposal with the Hastings ratio.

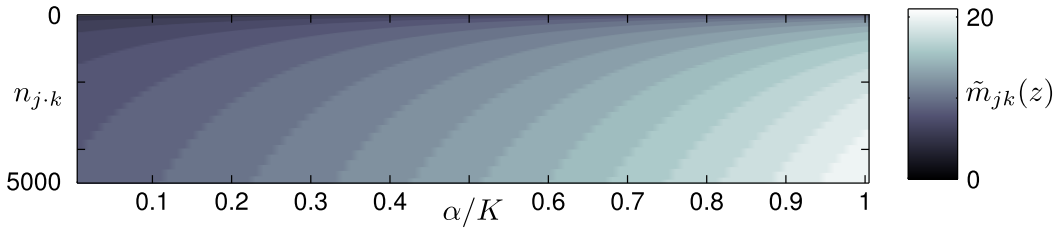


Figure 6.4: A visualization of how $\tilde{m}_{jk}(z)$ is determined.

$\hat{\hat{v}} \triangleq \{\hat{\hat{m}}, \hat{\hat{\beta}}, \hat{\hat{\pi}}, \hat{\hat{\theta}}, \hat{\hat{z}}\}$, is deferred to the restricted sampler like what was done in the DPMM case. The main difference is that a new set of global topic proportions, β, must be proposed conditioned on the labels, z.

As stated previously, the closer the proposal distribution, $q(\cdot)$, is to the target distribution, $p(\cdot)$, the better the convergence. Thus, it would be ideal to propose a new β from $p(\beta|z)$. Unfortunately, this conditional distribution cannot be expressed analytically unless one additionally conditions on the dish counts, $m_{\cdot k}$ (cf. Equation (6.7)). Since the distribution of dish counts depends on β itself, we approximate its value with

$$\tilde{m}_{jk}(z) = \arg\max_m p(m|\beta\tfrac{\alpha}{K}, z) = \arg\max_m \frac{\Gamma(\frac{\alpha}{K})}{\Gamma(\frac{\alpha}{K} + n_{j\cdot k})} s(n_{j\cdot k}, m)(\tfrac{\alpha}{K})^m, \tag{6.21}$$

where the global topic proportions have essentially been substituted with $\frac{1}{K}$. A visualization of the $\tilde{m}_{jk}(z)$ is shown in Figure 6.4.

We note that the dependence on z is implicit through the counts, n. We can then define a proposal for the global topics proportions as

$$q(\hat{\beta}|\hat{z}) = p(\hat{\beta}|\hat{\tilde{m}}) = \text{Dir}\left(\hat{\beta}_1, \ldots, \hat{\beta}_K, \hat{\beta}_{K+1}\ ;\ \hat{\tilde{m}}_{\cdot 1}, \cdots, \hat{\tilde{m}}_{\cdot K}, \gamma\right), \tag{6.22}$$

where we denote $\hat{\tilde{m}} \triangleq \tilde{m}(\hat{z})$. We emphasize that the approximate $\hat{\tilde{m}}$ is only used to create a proposal distribution similar to the prior and the resulting chain will still satisfy detailed balance.

The global topic weights β not only complicate the proposal distributions, but also the computation of the Hastings ratio. When $\hat{\pi}$ and $\hat{\theta}$ are directly sampled from their respective posterior distributions and $\hat{\overline{v}}$ is deferred, the resulting Hastings ratio can be expressed as

$$H = \frac{p(\hat{\beta}, \hat{z})p(x|\hat{z})}{p(\beta, z)p(x|z)} \cdot \frac{q(z|\hat{v}, \hat{\overline{v}})q(\beta|z)}{q(\hat{z}|v, \overline{v})q(\hat{\beta}|\hat{z})} \tag{6.23}$$

With a considerable amount of algebra, we show in Appendix C that the joint prior $p(\beta, z)$ can be expressed as

$$p(\beta, z) = \gamma \beta_{K+1}^{\gamma-1} \prod_{k=1}^{K} \beta_k^{-1} \left[\prod_{j=1}^{D} \frac{\Gamma(\alpha)}{\Gamma(\alpha + n_{j\cdot\cdot})} \prod_{k=1}^{K} \frac{\Gamma(\alpha\beta_k + n_{j\cdot k})}{\Gamma(\alpha\beta_k)} \right]. \tag{6.24}$$

We now present two methods of proposing new labels that fit within this split/merge framework.

■ 6.5.1 Local Splits and Merges

In the DP Sub-Cluster algorithm presented in Chapter 5, splits were constructed by altering the labels of a *single* cluster. We define this type of split move as a "local" split since only assignments within one cluster or topic are changed. This is in contrast to a "global" split (discussed shortly), which changes all topic assignments.

A local split of topic $\natural$ into topics $\flat$ and $\sharp$ in HDPs is proposed in a similar fashion as in DPMMs. The sub-topic statistics are used to propose a new topic assignment by first constructing temporary parameters, $\{\tilde{\pi}_{1\flat}, \tilde{\pi}_{1\sharp}, \ldots, \tilde{\pi}_{D\flat}, \tilde{\pi}_{D\sharp}, \tilde{\theta}_{j\flat}, \tilde{\theta}_{j\sharp}\}$

$$(\tilde{\pi}_{j\flat}, \tilde{\pi}_{j\sharp}) = \pi_{j\natural} \cdot (\overline{\pi}_{\natural\ell}, \overline{\pi}_{\natural r}), \quad \forall j \in \{1, \ldots, D\} \tag{6.25}$$

$$(\tilde{\theta}_{\flat}, \tilde{\theta}_{\sharp}) = (\overline{\theta}_{\natural\ell}, \overline{\theta}_{\natural r}). \tag{6.26}$$

Conditioned on these temporary topic parameters, new topic assignments for data associated with topic $\natural$ are drawn from

$$q(\hat{z}_{ji}|v, \overline{v}, Q_{\text{split-}\natural}) = \sum_{k \in \{\flat, \sharp\}} \frac{\tilde{\pi}_{jk} f_x(x_{ji}; \tilde{\theta}_k) \mathbb{I}[\hat{z}_{ji} = k]}{\tilde{\pi}_{j\flat} f_x(x_{ji}; \tilde{\theta}_{\flat}) + \tilde{\pi}_{j\sharp} f_x(x_{ji}; \tilde{\theta}_{\sharp})}, \quad \forall \{j, i\} \in \mathcal{I}_{\natural}. \tag{6.27}$$

A new $\hat{\beta}$ can be drawn by splitting $\beta_{\natural}$, into two parts, $\hat{\beta}_{\flat}$ and $\hat{\beta}_{\sharp}$. The proportions of the new weights are generated in a local version of Equation (6.22) as follows

$$q(\hat{\beta}_{\flat}, \hat{\beta}_{\sharp}|\hat{z}, \beta_{\natural}) = \text{Dir}(\hat{\beta}_{\flat}/\beta_{\natural}, \hat{\beta}_{\sharp}/\beta_{\natural}; \hat{\overline{m}}_{\cdot\flat}, \hat{\overline{m}}_{\cdot\sharp}). \tag{6.28}$$

The corresponding merge move combines topics $\flat$ and $\sharp$ into topic $\natural$ by deterministically

performing

$$q(\hat{z}_{ji}|v, Q_{\text{merge-}\flat\sharp}) = \mathbb{I}[\hat{z}_{ji} = \natural], \quad \forall\{j,i\} \in \mathcal{I}_\flat \cup \mathcal{I}_\sharp, \tag{6.29}$$

$$q(\hat{\beta}_\natural|v) = \delta(\hat{\beta}_\natural - (\beta_\flat + \beta_\sharp)). \tag{6.30}$$

We show in Appendix C that this results in a Hastings ratio for a local split of

$$H^{\text{local}}_{\text{split-}\natural} = \frac{\gamma\Gamma(\hat{\tilde{m}}_{\cdot\flat})\Gamma(\hat{\tilde{m}}_{\cdot\sharp})}{\Gamma(\hat{\tilde{m}}_{\cdot\flat} + \hat{\tilde{m}}_{\cdot\sharp})} \frac{\beta_\natural^{\hat{\tilde{m}}_{\cdot\flat}+\hat{\tilde{m}}_{\cdot\sharp}}}{\beta_\flat^{\hat{\tilde{m}}_{\cdot\flat}}\beta_\sharp^{\hat{\tilde{m}}_{\cdot\sharp}}} \frac{p(x|\hat{z})}{p(x|z)} \frac{1}{q(\hat{z}|v,\overline{v},Q_{\text{split-}\natural})} \times \frac{Q^{K+1}_{\text{merge-}\flat\sharp}}{Q^K_{\text{split-}\natural}} \prod_{j=1}^{D} \frac{\Gamma(\alpha\beta_\natural)}{\Gamma(\alpha\beta_\natural + n_{j\cdot\natural})} \prod_{k\in\{\flat,\sharp\}} \frac{\Gamma(\alpha\hat{\beta}_k + \hat{n}_{j\cdot k})}{\Gamma(\alpha\hat{\beta}_k)}, \tag{6.31}$$

where the dependence of $\tilde{m}$ on $\hat{z}$ is omitted. All of these terms can be computed efficiently for the same reasons as in the DPMM case. Similarly, the Hastings ratio for a local merge is

$$H^{\text{local}}_{\text{merge-}\flat\sharp} = \frac{\Gamma(\tilde{m}_{\cdot\flat} + \tilde{m}_{\cdot\sharp})}{\gamma\Gamma(\tilde{m}_{\cdot\flat})\Gamma(\tilde{m}_{\cdot\sharp})} \frac{\beta_\flat^{\tilde{m}_{\cdot\flat}}\beta_\sharp^{\tilde{m}_{\cdot\sharp}}}{\hat{\beta}_\natural^{\tilde{m}_{\cdot\flat}+\tilde{m}_{\cdot\sharp}}} \frac{p(x|\hat{z})}{p(x|z)} \frac{q(z|\hat{v},\hat{\overline{v}},Q_{\text{split-}\natural})}{1} \times \frac{Q^{K-1}_{\text{split-}\natural}}{Q^K_{\text{merge-}\flat\sharp}} \prod_{j=1}^{D} \frac{\Gamma(\alpha\hat{\beta}_\natural + \hat{n}_{j\cdot\natural})}{\Gamma(\alpha\hat{\beta}_\natural)} \prod_{k\in\{\flat,\sharp\}} \frac{\Gamma(\alpha\beta_k)}{\Gamma(\alpha\beta_k + n_{j\cdot k})}. \tag{6.32}$$

■ 6.5.2 Global Split/Merge Proposals

Since the clusters in topic modeling are defined over a discrete dictionary, word distributions for different topics tend to have significant overlap. Consequently, local splits may be rejected because the joint likelihood cannot increase enough unless points move to and from the other topics as well. In this section, we consider generating global split/merge moves by constructing a split or merge followed by reassigning *all* topic assignments.

A global split proposal is constructed by first forming temporary topic proportions and parameters with

$$(\tilde{\pi}_{j\flat}, \tilde{\pi}_{j\sharp}) = \pi_{j\natural} \cdot (\overline{\pi}_{\natural\ell}, \overline{\pi}_{\natural r}), \quad \tilde{\pi}_{jk} = \pi_{jk}, \quad \forall k \neq \natural, \quad \forall j \in \{1,\ldots,D\} \tag{6.33}$$

$$(\tilde{\theta}_\flat, \tilde{\theta}_\sharp) = (\overline{\theta}_{\natural\ell}, \overline{\theta}_{\natural r}), \qquad \tilde{\theta}_k = \theta_k, \quad \forall k \neq \natural. \tag{6.34}$$

New topic assignments are then sampled for all words:

$$q(\hat{z}_{ji}|v,\overline{v}) = \sum_k \frac{\tilde{\pi}_{jk} f_x(x_{ji};\tilde{\theta}_k)\mathbb{I}[z_{ji}=k]}{\sum_l \tilde{\pi}_{jl} f_x(x_{ji};\tilde{\theta}_l)}, \quad \forall\{j,i\}. \tag{6.35}$$

Note that the summations in this categorical distribution index over all K previous topics and topic $\flat$ and $\sharp$, but excludes the index $\natural$. We then sample $\hat{\beta} \sim q(\hat{\beta}|\hat{z})$ via Equation (6.22). The remaining split proposal follows directly from Algorithm 6.1.

The corresponding merge first samples a temporary parameter, $\tilde{\theta}_\natural$,

$$\tilde{\theta}_\natural \sim q(\tilde{\theta}_\natural|x,z) \propto f_x(x_{\mathcal{I}_\flat \cup \mathcal{I}_\sharp};\tilde{\theta}_\natural) f_\theta(\tilde{\theta}_\natural;\lambda), \tag{6.36}$$

and constructs the remaining temporary variables with

$$\tilde{\pi}_{j\natural} = \pi_{j\flat} + \pi_{j\sharp}, \quad \tilde{\pi}_{jk} = \pi_{jk}, \quad \forall k \neq \flat,\sharp, \quad \forall j \in \{1,\ldots,D\}, \tag{6.37}$$

$$\tilde{\theta}_k \ = \theta_k, \quad \forall k \neq \flat,\sharp. \tag{6.38}$$

$\hat{z}$ is then sampled via Equation (6.35), followed by sampling $\hat{\beta}$ via Equation (6.22).

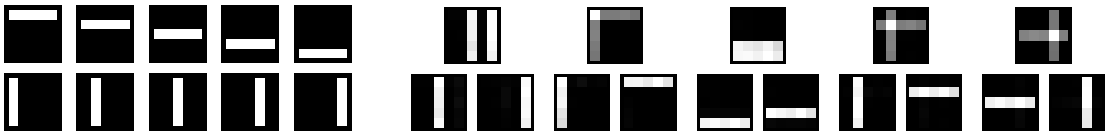
We show in Appendix C that the Hastings ratios for the global split and merge are

$$\begin{aligned} H^{\text{global}}_{\text{split-}\natural} = & \frac{\gamma\Gamma\left(\gamma+\sum_{k=1}^{K}\tilde{m}_{\cdot k}\right)}{\Gamma\left(\gamma+\sum_{k=1}^{K+1}\hat{\tilde{m}}_{\cdot k}\right)} \frac{p(x|\hat{z})}{p(x|z)} \frac{q(z|\hat{v},\hat{\overline{v}})}{q(\hat{z}|v,\overline{v})} \frac{q(\tilde{\theta}_\natural|x,z)}{1} \frac{Q^{K+1}_{\text{merge-}\flat\sharp}}{Q^{K}_{\text{split-}\natural}} \\ & \times \prod_{k=1}^{K} \frac{\beta_k^{\tilde{m}_{\cdot k}}}{\Gamma(\tilde{m}_{\cdot k})} \prod_{k=1}^{K+1} \frac{\Gamma(\hat{\tilde{m}}_{\cdot k})}{\hat{\beta}_k^{\hat{\tilde{m}}_{\cdot k}}} \times \prod_{j=1}^{D}\prod_{k=1}^{K} \frac{\Gamma(\alpha\beta_k)}{\Gamma(\alpha\beta_k+n_{j\cdot k})} \prod_{k=1}^{K+1} \frac{\Gamma(\alpha\hat{\beta}_k+\hat{n}_{j\cdot k})}{\Gamma(\alpha\hat{\beta}_k)}. \end{aligned} \tag{6.39}$$

$$\begin{aligned} H^{\text{global}}_{\text{merge-}\flat\sharp} = & \frac{\Gamma\left(\gamma+\sum_{k=1}^{K}\tilde{m}_{\cdot k}\right)}{\gamma\Gamma\left(\gamma+\sum_{k=1}^{K-1}\hat{\tilde{m}}_{\cdot k}\right)} \frac{p(x|\hat{z})}{p(x|z)} \frac{q(z|\hat{v},\hat{\overline{v}})}{q(\hat{z}|v,\overline{v})} \frac{1}{q(\tilde{\theta}_\natural|x,\hat{z})} \frac{Q^{K-1}_{\text{split-}\natural}}{Q^{K}_{\text{merge-}\flat\sharp}} \\ & \times \prod_{k=1}^{K} \frac{\beta_k^{\tilde{m}_{\cdot k}}}{\Gamma(\tilde{m}_{\cdot k})} \prod_{k=1}^{K-1} \frac{\Gamma(\hat{\tilde{m}}_{\cdot k})}{\hat{\beta}_k^{\hat{\tilde{m}}_{\cdot k}}} \times \prod_{j=1}^{D}\prod_{k=1}^{K} \frac{\Gamma(\alpha\beta_k)}{\Gamma(\alpha\beta_k+n_{j\cdot k})} \prod_{k=1}^{K-1} \frac{\Gamma(\alpha\hat{\beta}_k+\hat{n}_{j\cdot k})}{\Gamma(\alpha\hat{\beta}_k)}. \end{aligned} \tag{6.40}$$

The Hastings ratio for a merge move again depends on the resulting sub-topics parameters. Similar to the argument made for local splits and merges, we approximate the resulting sub-topic parameters with the main-topic parameters prior to the proposed merge.

Unfortunately, unlike the local split/merge proposals, generating a new $\hat{z}$ requires significant computation by looping through all data points. As such, we randomly propose only one split and merge at every iteration. Thus, $Q^K_{\text{split-}\natural} = 1/K$ and $Q^K_{\text{merge-}\flat\sharp} = 2/(K(K-1))$. Even with this added computation, we show in the Section 6.6 that convergence rates improve dramatically. Additionally, we note that the presented global split is very different from all previous local split/merge algorithms. We believe it is for this reason that the exhibited performance of [122] is poor.



(a) True Topics (b) Visualizing Sub-Topics

Figure 6.5: Visualizing sub-topics on the synthetic "bars" example of [47]. (a) Visualization of the 10 true topics. (b) Visualization of the word distributions for each topic without splits/merges for $K = 5$. Top row: regular-topics, $f_x(\circ;\theta)$. Bottom row: sub-topics, $f_x(\circ;\overline{\theta})$.

6.6 Experimental Results

In this section, we test the proposed method on topic modeling. The sampler is summarized in Algorithm 6.2. Hyper-parameters are fixed in our implementation, although resampling techniques [116] can be easily incorporated. All results were obtained by averaging 10 sample paths.

Algorithm 6.2 Sub-Cluster HDP Sampler

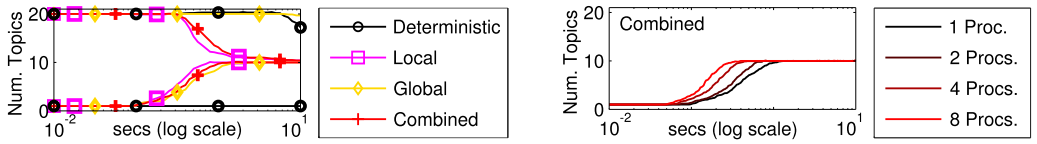
1. Initialize β and z randomly.
2. Sample π, θ, $\overline{\pi}$, and $\overline{\theta}$ via Equations (6.8), (6.9), (6.18), and (6.19).
3. Sample z and $\overline{z}$ via Equations (6.10) and (6.20).
4. Propose $\lfloor \frac{K}{2} \rfloor$ local merges followed by K local splits.
5. Propose a global merge followed by a global split.
6. Sample m and $\overline{m}$ via Equations (6.6) and (6.16).
7. Sample β and $\overline{\beta}$ via Equations (6.7) and (6.17).
8. Repeat from Step 2 until convergence.

6.6.1 Visualizing Sub-Topics

We generated 200 documents from the synthetic "bars" example of [47] with a dictionary of 25 words that can be arranged in a 5x5 grid. Figure 6.5a shows the 10 topics used to generate documents, each of which is a horizontal or vertical bar. To visualize the auxiliary variables, we initialize to 5 topics and **do not** propose any splits or merges. The resulting regular-topic and sub-topic word distributions are shown in Figure 6.5b.

6.6.2 Parallelizability & Convergence

Figure 6.6a considers the different split/merge proposals. The local and combined moves both converge whereas the deterministic moves are always rejected. While there is no benefit of global moves in such a well-separated dataset, we have observed that the combination of the local and global moves outperforms any single type in real-world



(a) Split/Merge Proposals (b) Parallelization

Figure 6.6: (a) Different split/merge proposals. (b) Different levels of parallelization.

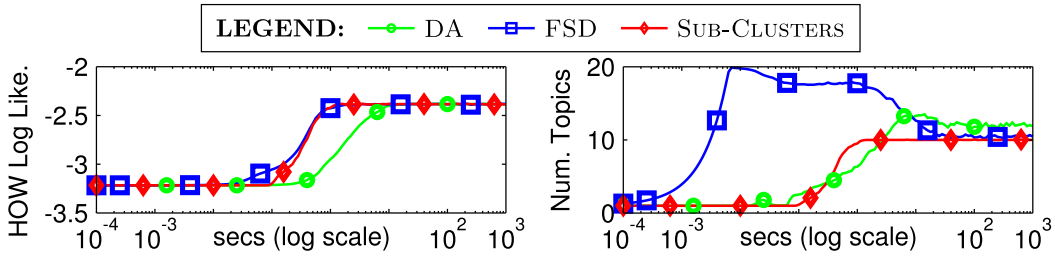


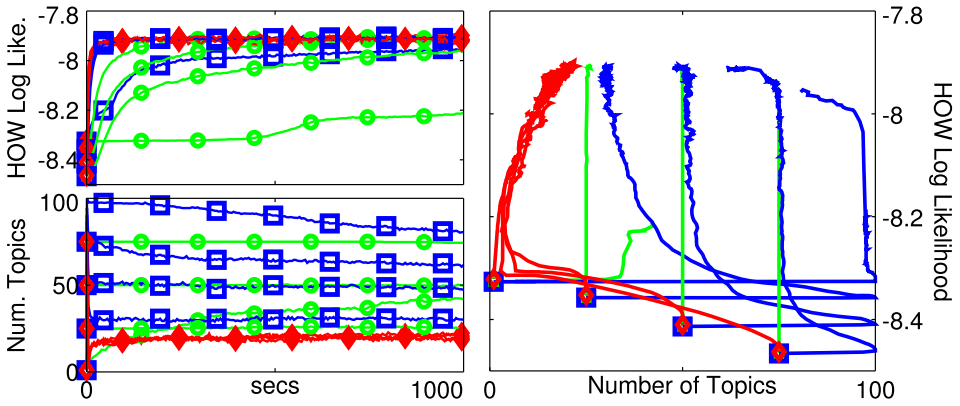
Figure 6.7: Results on the "bars" example. All algorithms use a single core and are initialized to one topic.

datasets. Furthermore, every step of Algorithm 6.2 can be parallelized. As evident from Figure 6.6b, we achieve a linear speedup in the number of processors.

We compare our algorithm (Sub-Clusters) with the Direct Assignment (DA) sampler and the Finite Symmetric Dirichlet with $L = 20$ (FSD). We emphasize the importance here is to analyze convergence *speed*, since all algorithms should sample from the same model. We monitor the number of inferred topics and perform cross-validation by holding out one word (HOW) from each document. Results are shown *without* parallelization in Figure 6.7. Notice that while FSD seems to have converged according to the HOW likelihood after 1 second, it is clear from the number of topics that it does not converge until 100 seconds. While cross-validation metrics evaluate model fit, they alone cannot determine convergence of the MCMC procedure. It is also interesting to note that the FSD sampler tends to first create all L topics and slowly prune them away. Even larger speedups occur when parallelization is exploited.

■ 6.6.3 Associated Press Dataset

Next, we consider articles from the Associated Press (AP) [9] containing 2K documents and 436K words. We manually increased L to 100 for the FSD algorithm. Results for each algorithm using 16 cores with 1, 25, 50, and 75 initial topics are shown in Figure 6.8. We note that DA cannot be parallelized. Each sampler should converge to the same distribution regardless of the initialization. However, Figure 6.8a shows that while HOW likelihood converges for 3/4 FSD algorithms, the number of topics indicates that none of the sample paths of DA or FSD have converged. In contrast, all initializations of the Sub-Clusters method have converged to approximately 20



(a) AP Statistics (b) AP Sample Paths

Figure 6.8: Results on the Associated Press dataset for 1, 25, 50, and 75 initial topics.

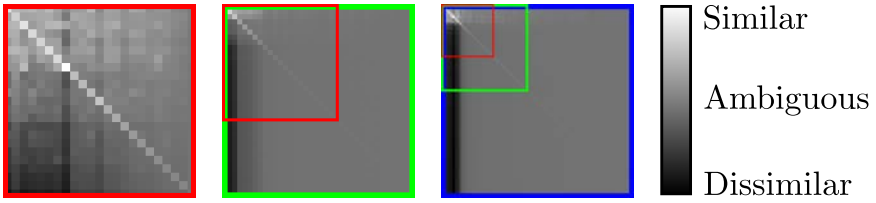


Figure 6.9: Confusion matrices on the Associated Press dataset for Sub-Clusters, DA, and FSD (left to right). Outlines are overlaid to compare size.

topics. We visualize the *path* each sampler takes in the joint HOW likelihood / number of topics space in Figure 6.8b. This figure shows the stark difference in how the different samplers approach the problem. FSD explodes the number of topics and slowly prunes them. The path of FSD indicates that it is approaching the same stationary point, just at a very slow rate. DA has difficulty because of the small local changes and the lack of parallelization. Sub-Clusters efficiently make these changes from the splits and merges.

Figure 6.9 visualizes the confusion matrices, C, of the inferred topics. Each element of C is defined as:

$$C_{r,c} = \sum_x f_x(x;\theta_r) \log f_x(x;\theta_c), \tag{6.41}$$

and approximately measures how likely a word from one topic is under another topic. We find that DA and FSD both infer many topics that are easily confused, whereas Sub-Clusters finds more distinguishable topics.

While DA and FSD have clearly not converged, we cannot be certain that Sub-Clusters has converged. Concrete statements about convergence to the stationary distribution are quite difficult. Instead, we consider running each sample path for a total of 2,000 seconds. After 1,000 seconds, we switch DA and FSD to Sub-Clusters,

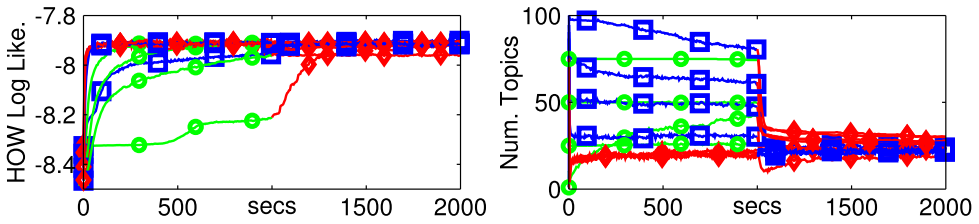
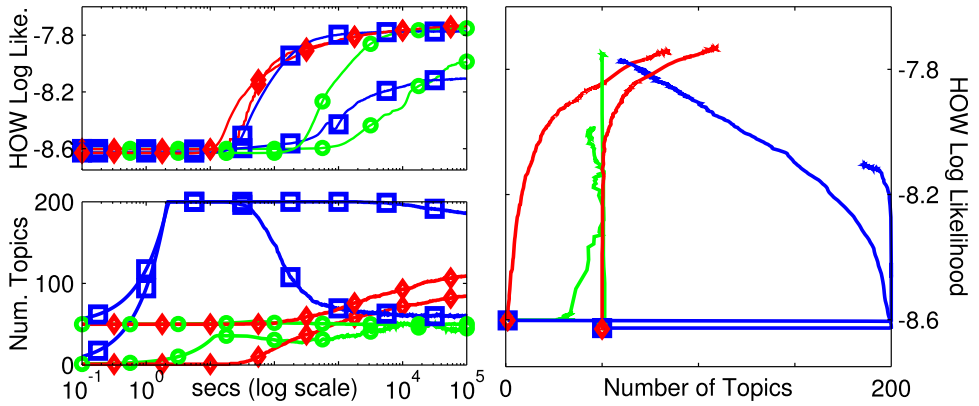


Figure 6.10: Results on the Associated Press dataset after switching algorithms at 1000 secs.



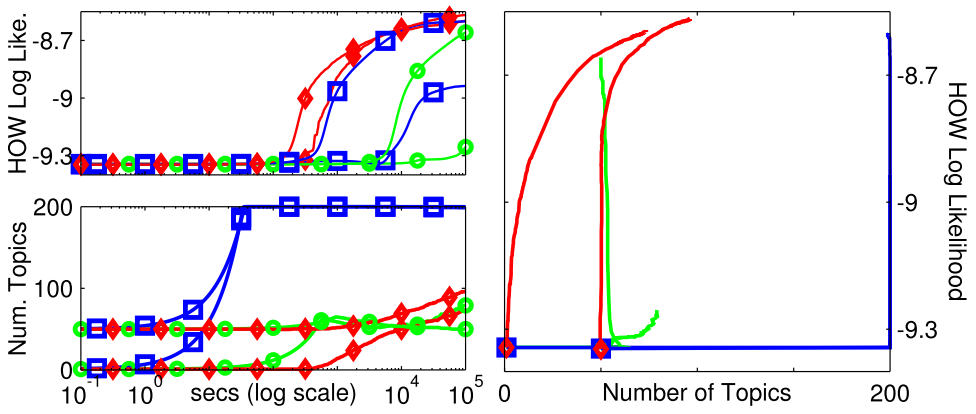
(a) Enron Statistics (b) Enron Sample paths

Figure 6.11: Results on the dataset of Enron emails for 1 and 50 initial topics.

and switch Sub-Clusters to FSD. We expect to see all of these algorithms converge to the same point. As shown in Figure 6.10, switching from Sub-Clusters to FSD did not change the results, but switching from DA and FSD to Sub-Clusters immediately helps the algorithms reach the topics that were inferred by the Sub-Clusters method. We note that the number of topics after the switch is slightly higher than the original. We believe this is because it is difficult for the Sub-Clusters method to create topics with few words since, by construction, the splits make large moves. In contrast, DA and FSD often create single word topics.

■ 6.6.4 Large Datasets

Finally, we consider two large datasets from [2]: Enron Emails containing 6M words in 40K documents and New York Times (NYTimes) Articles containing 100M words in 300K documents. We note that the NYTimes dataset is 3 orders of magnitude larger than the datasets considered in the previous HDP split/merge work of [122]. Again, we manually increased L to 200 for the FSD algorithm. Results are shown in Figures 6.11–6.12 initialized to 1 and 50 topics. In such large datasets, it is difficult to know how long convergence will take. After 28 hours, it seems as though none of the algorithms



(a) NYTimes Statistics (b) NYTimes Sample Paths

Figure 6.12: Results on the dataset of New York Times articles for 1 and 50 initial topics.

have converged. However, Sub-Clusters seems to be approaching a solution, whereas FSD has yet to prune topics and DA has yet to to achieve a good cross-validation score. Thirty inferred topics using Sub-Clusters on the NYTimes dataset are shown in Figure 6.13.

■ 6.7 Discussion

We have extended the DP Sub-Cluster algorithm to Hierarchical Dirichlet Processes. In addition to demonstrating the extensibility of the DP Sub-Cluster algorithm, we have also discovered an important observation about HDPs; current sampling algorithms can take a very long time to converge. Moreover, while commonly-used cross-validation metrics may measure how well the current model fits the data, our experiments have shown that they do not accurately capture convergence of Markov chain Monte Carlo methods to their stationary distributions.

We would like to reiterate that the only previous HDP MCMC sampling algorithm that incorporated splits and merges was the method of Wang and Blei [122], which found that their extension of [27] to HDPs only slightly helped convergence. Our results indicate that global split and merge moves can drastically improve results due to the highly overlapping distributions in topic models.

One interesting direction of research to extend this work is to consider its application to Hierarchical Dirichlet process Hidden Markov models (HDP-HMMs) [116]. HDP-HMMs essentially model an HMM with a nonparametric prior on state transition and emission distributions. An example graphical model for an HMM is shown in Figure 6.14. In the HDP-HMM, a realization is drawn according to

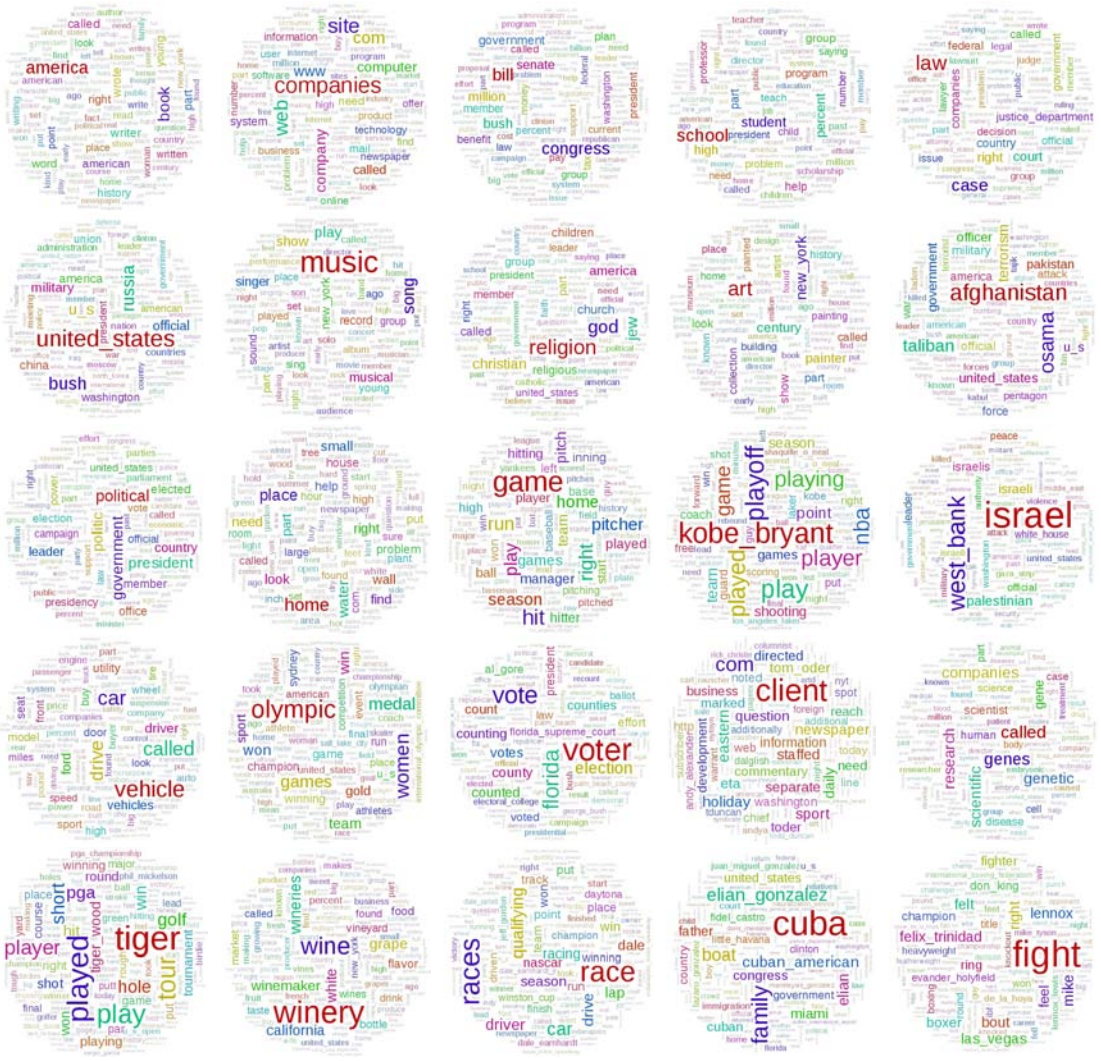


Figure 6.13: Subset of learned topic distributions from the New York Times dataset.

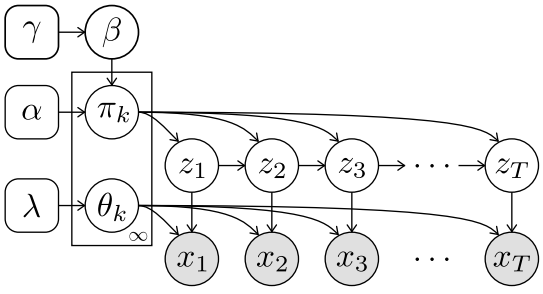


Figure 6.14: A graphical model for the HDP-HMM.

$$\beta \sim \text{GEM}(1, \gamma), \tag{6.42}$$
$$\pi_k \sim \text{DP}(\alpha, \beta), \qquad \forall k \in \{1, 2, \dots\}, \tag{6.43}$$
$$\theta_k \sim f_\theta(\theta_k; \lambda), \qquad \forall k \in \{1, 2, \dots\}, \tag{6.44}$$
$$z_t \sim \pi_{z_{t-1}}, \qquad \forall t \in \{2, \dots, T\}, \tag{6.45}$$
$$x_t \sim f_x(x_t; \theta_{z_t}), \qquad \forall t \in \{1, \dots, T\}, \tag{6.46}$$

where π_k denotes an infinitely-sized matrix containing the transition distribution between the infinite number of possible states. The Direct Assignment (DA) sampling algorithm discussed in Section 2.9.3 can be applied to HDP-HMM models [116] to perform inference without needing model approximations. However, as was explored in [103] for finite HMMs, inference that only considers changing one state assignment at a time typically exhibits poor convergence. As an alternative to the DA algorithm, Fox [34] considers approximating the infinite state space with some truncation L combined with the forward-backward algorithm [99] to make larger moves.

The Sub-Cluster algorithm should easily adapt to temporal sequences and the forward-backward algorithm because the resulting "sub-sequences" (similar to the sub-clusters) will already be instantiated. For this reason, we believe that the use of the HDP Sub-Cluster algorithm can significantly improve results in the HDP-HMM, without requiring a finite-model approximation.

Chapter 7

Intrinsic Image Decomposition via the SV-DPGMM

In this chapter, we consider applying the DPMM Sub-Cluster inference algorithm to the intrinsic image decomposition problem in computer vision. Intrinsic image analysis, first introduced in [5], is the problem of decomposing an image into various, typically generative, scene characteristics. The intrinsic reflectance and shading images are of particular interest. In this specific decomposition, the reflectance image contains the albedo of the underlying object surface and the shading image captures the amount of reflected light from the surface. Under the assumption of a Lambertian reflectance model, where the perceived illumination is constant from all angles of incidence, the observed image decomposes into the product of the shading and reflectance image. An example of this decomposition obtained using the proposed algorithm is shown in Figure 7.1.

While interesting in its own right, intrinsic image analysis is also important for other fields of computer vision. For example, the shading image can be exploited in shape-from-shading algorithms to reveal the underlying 3D structure of an object. Other quantities of interest describing the scene illumination, such as the number, location, and color of the light sources, can also be inferred from the shading image. Use of the reflectance image improves many segmentation algorithms, where shading effects often introduce artifacts.

We consider the problem of intrinsic reflectance and shading decomposition from a

Figure 7.1: An example of the intrinsic image problem. From left to right: original masked image, inferred shading image, inferred reflectance image. Result were obtained using the proposed method

single observation. The Retinex algorithm [7, 54, 75], one of the first proposed methods for this problem, estimates the intrinsic images by detecting edges in the observed image and solving for a reflectance image that has matching gradients at the detected edges. Surprisingly, existing methods still require these gradient-matching terms to achieve good results except for the recent algorithm of [4].

The proposed method, which does not incorporate the gradient-matching terms, combines a Dirichlet process Gaussian mixture model (DPGMM) for the reflectance image with a Gaussian process for the shading image. While aspects of the approach are certainly related to previous methods, the presented formulation differs by: (1) treating the observed image as an observation from a generative, stochastic process; (2) using more expressive smoothness terms for the shading image; and (3) developing inference techniques that are extremely robust to initialization.

■ 7.1 Related Work

There have been many algorithms developed to decompose images into their intrinsic components. Some algorithms use multiple images to disambiguate the decomposition (e.g., [125]), while others use data-driven, patch-based algorithms (e.g., [36]). We review the most related work here.

The original Retinex algorithm [75], which many algorithms build upon (e.g., [7, 37, 39, 54, 89, 107, 115]), still performs well decades after its original inception. In fact, results on the MIT Intrinsic Image Dataset [48] show that the original Retinex algorithm formulated in 1971 outperforms all other algorithms prior to 2009. The different flavors of the Retinex algorithm all include two underlying concepts: sharp edges should occur in the reflectance image, and the shading image should be smooth. Edges in the image are first detected, typically by thresholding intensity and/or chromaticity gradients. The gradients of the reflectance image are then favored to match the gradients in the observed image at the detected edges. This type of interaction is often referred to as the "Retinex term". Because this term only biases solutions at edges, a smoothness assumption in the shading image is then used to propagate the information of the Retinex term globally.

Many authors have observed that a small set of distinct colors can often be used to model the reflectance image (e.g., [4, 39, 107]). In particular, Shen et al. [107] groups reflectance values based on a local texture patch. They develop a "match weight" for each pairwise match that is used as a heuristic to weight reflectance differences in their energy functional. The approach of Gehler et al. [39] explicitly partitions the pixels based on their reflectance colors into K clusters. However, it is unclear how to set K *a priori*, since one would expect this value to be dependent on the particular image. In contrast, we model the reflectance image as being drawn from a Dirichlet process mixture model that does not predefine a model order.

Smoothness in the shading image is most commonly enforced by using a Markov random field (MRF) with an L_1 or L_2 penalty on the difference of neighboring shading

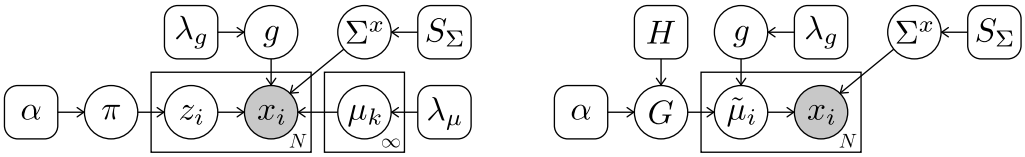
pixels. As mentioned in Chapter 4, an L_2 penalty is equivalent to using an improper Gaussian MRF (GMRF) prior [87]. These types of model are also used in [39], [107], and every method in the survey paper of [48]. In this work, we place a similar prior on the shading image. However, instead of restricting the smoothness to be a 4-connected GMRF as was done previously, we allow for a much broader class of smooth functions by placing a Gaussian process (GP) prior on the shading image. Certain properties of the GP, which we will exploit, lend themselves to efficient inference (such as stationary covariance kernels). Stationary GMRFs are approximately finite realizations of GPs with stationary covariance kernels. However, as we shall see, framing the model using a GP allows us to exploit two advantages: (1) inference is simplified when framing the model with GPs; and (2) changing the prior smoothness is a matter of altering the covariance kernel without having to explicitly adapt to a different graphical MRF structure.

The two current state-of-the-art intrinsic image algorithms take quite different approaches. The recent work of Barron and Malik [4], called SIRFS, is the current best-performing algorithm on the MIT Intrinsic Image Dataset [48]. SIRFS differs from many other methods by incorporating a model of the 3D structure of an object. In this framework, the shading image is seen as a by-product of the lighting conditions and 3D geometry. One might draw the conclusion from the results obtained using SIRFS that modeling the 3D structure is essential to good performance; however, as we will show, that is not necessarily the case. Furthermore, training and inference in the SIRFS model is complex because of the numerous parameters, and the current SIRFS framework only allows inference from a single spherical harmonic illumination. Multiple point light sources, which often occur in real-world scenes, cannot be modeled without modifying the formulation and developing a new inference procedure. As such, we address the problem from the more traditional approach of explicitly inferring the shading image.

The second best-performing algorithm is the approach of Gehler et al. [39], which we have already mentioned above. Our model, called the spatially-varying Dirichlet process Gaussian mixture model (SV-DPGMM), is actually quite similar to the model of [39]. As such, we explicitly summarize the differences in Table 7.1. While [39] has shown that the Retinex term improves results, it is difficult to incorporate such a term in a *generative* model. Moreover, our experiments show that by using a more expressive shading model and improved performance, the Retinex term is unnecessary to achieve state-of-the-art results. Additionally, we note one final distinction between the work presented here and all previous works. The SV-DPGMM is a generative model that explicitly captures the noise characteristics of the observed image. As we shall see, the generative model benefits from incorporating Bayesian priors that adapt to different noise variances and object complexities. Moreover, as we briefly discuss in Chapter 8, the particular generative model can be modularized into a larger framework. Studying the problem when applied to noise-free images certainly provides insight. However, as we shall see in Section 7.6, noise significantly degrades results of algorithms that do not

Table 7.1: Differences in Algorithms for Intrinsic Image Decomposition

	Gehler et al. [39]	**SV-DPGMM**
Shading Smoothness	4-connected GMRF	Gaussian Process
Reflectance Prior	Uniform over fixed K clusters	Dirichlet Process
Observations	Noiseless	Log-Normal Noise
Probabilistic Model	Discriminative	Generative
Retinex Term	Yes	No
Inference	Iterative Optimization	Robust & MCMC



(a) Model with Explicit Labels (b) Model with Atoms

Figure 7.2: The generative graphical model with two equivalent representations. See text for description. $\lambda_\mu = \{\theta, \Sigma^\mu\}$ and $\lambda_g = \{\kappa, \sigma_g^2, \nu, l\}$ denote sets of hyper-parameters in the labeled model. λ_μ is encoded in the base measure, H, in the atom model.

explicitly model it.

■ 7.2 Generative Model

As is common in intrinsic image analysis, we assume a Lambertian surface model, where an image decomposes into the product of a shading a reflectance image. We now present a generative model that explicitly builds upon this decomposition.

Multiplicative models become additive in the log domain, which simplifies noise characteristics and interactions between Gaussian terms. Consequently, for the remainder of this chapter, we will work in the log domain, where the the log of the observed image, x, is assumed to be generated from the sum of the log shading and the log reflectance image. We note that this additive model in the log domain implicitly defines an equivalent multiplicative model in the regular image domain.

We now detail the specific generative process. The related graphical model capturing the dependency structure is depicted in Figure 7.2. The log reflectance image is generated from a standard Dirichlet process Gaussian mixture model (DPGMM) as follows: (1) an infinite-length vector or mixture weights, π, is drawn from a stick-breaking process [106], (2) the 3D mean color for each cluster, μ_k, is drawn from a multivariate Gaussian prior, (3) the cluster assignment for each pixel, i, denoted z_i, is drawn from a categorical distribution with parameters π. The following expressions summarize this

process:

$$p(\pi) = \mathrm{GEM}(\pi\ ;\ 1, \alpha), \tag{7.1}$$

$$p(\mu) = \prod_{k=1}^{\infty} p(\mu_k) = \prod_{k=1}^{\infty} \mathcal{N}(\mu_k\ ;\ \theta, \Sigma^{\mu})\,, \tag{7.2}$$

$$p(z|\pi) = \prod_{i=1}^{N} p(z_i|\pi) = \prod_{i=1}^{N} \mathrm{Cat}(z_i\ ;\ \pi). \tag{7.3}$$

The hyper-parameters, α, θ, and Σ^{μ}, are chosen such that the resulting priors are broad and uninformative. We use μ without subscripts to denote the entire vector of means. Assuming 3 color channels and K clusters in an actual realization, μ will have dimensions $3K \times 1$. The log reflectance image, denoted μ_z, is then formed by setting each pixel to the corresponding cluster mean. That is, each pixel i of μ_z, denoted $[\mu_z]_i$ is assigned the following 3×1 vector-valued color

$$[\mu_z]_i = \mu_{z_i}. \tag{7.4}$$

The reflectance image, μ_z, is then represented by a $3N \times 1$ vector for an image with N pixels.

The log shading image, denoted g, is generated from a zero-mean Gaussian process (GP) with a stationary covariance kernel, κ. Shading images of interest (e.g., in the MIT Intrinsic Image Dataset [48]) are often generated from white-colored incident light. However, we find that allowing colored shading images generally results in better convergence. As such, we model g as a 3D Gaussian process with a covariance kernel that is a function of location and color. Denoting $\{i, m\}$ as color channel m of pixel i, and $\{j, n\}$ as color channel n of pixel j, the 3D *stationary* covariance kernel can then be expressed as a function of the different in location and color channels:

$$\kappa(\{i, m\}, \{j, n\}) = \kappa(i - j, m - n). \tag{7.5}$$

Furthermore, we are only interested in the values of the GP at the fixed grid locations. Since any subset of variables in a GP is jointly Gaussian, we can express the GP as

$$p(g) = \mathrm{GP}(g\ ;\ \kappa) = \mathcal{N}(g\ ;\ 0, \Sigma^{g})\,, \tag{7.6}$$

where Σ^{g} denotes the finite-dimensional covariance matrix obtained by evaluating the kernel, κ, at the grid points. The specific covariance kernel and parameters govern the smoothness properties of g and are inferred as part of the training phase.

Finally, we assume that the observed pixels in the the log image are drawn inde-

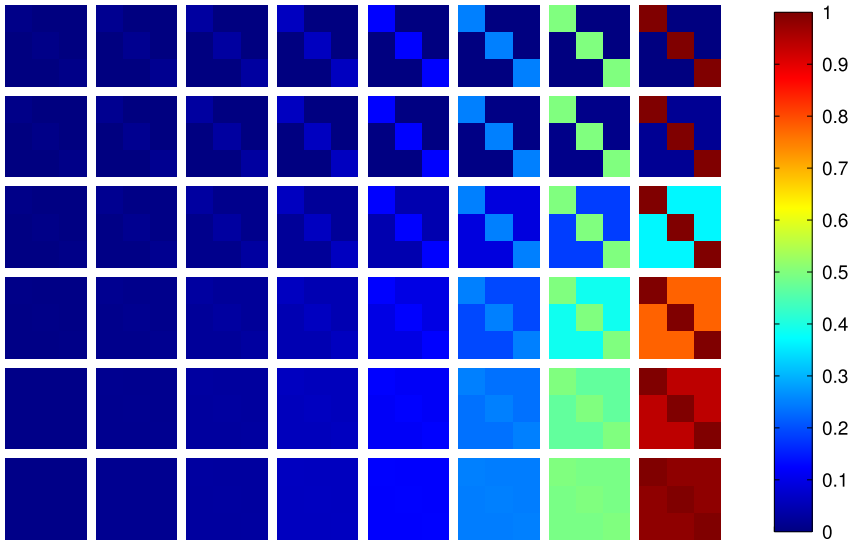


Figure 7.3: A visualization of the set of covariances, S_Σ. Note that the covariance matrices on the left have small values that do not display well.

pendently from the following Gaussian distribution:

$$p(x|\mu, z, g, \Sigma^x) = \prod_{i=1}^{N} p(x_i|\mu, z_i, g_i, \Sigma^x) = \prod_{i=1}^{N} \mathcal{N}(x_i; \mu_{z_i} + g_i, \Sigma^x)\,. \tag{7.7}$$

While one could assume a fixed observation covariance, Σ^x, we have found that this is quite difficult to set *a priori*, and instead treat the covariance as a latent variable. We note that one naïve solution is to use a cluster-specific covariance instead of a global covariance (e.g., via a Normal Inverse-Wishart distribution). However, as described in Section 7.3, a global observation covariance that is also Toeplitz lends itself to efficient inference of g. As we are unaware of conjugate priors on positive definite Toeplitz matrices, we generate Σ^x uniformly from a *discrete* set of covariances, S_Σ:

$$\Sigma^x = S_\Sigma(u), \quad u \sim \text{Uniform}(|S_\Sigma|). \tag{7.8}$$

The elements of S_Σ are chosen to be 3×3 matrices with color correlations logarithmically spaced in $[2^{-10}, 2^0]$ and covariances logarithmically spaced in $[2^{-7}, 2^0]$. This choice does not affect results significantly as long as the range is sufficiently broad. A visualization of the set of covariances is shown in Figure 7.3.

■ 7.2.1 Relation to DPGMMs

A typical DPGMM treats each pixel as being drawn from one of the infinite Gaussians with mean μ_k. This is equivalent to having mixture components that are the same re-

gardless of where the actual pixel is located. Following the discussion in the background (Section 2.9.2), the DPMM can be expressed in the explicit atom formulation, where the Dirichlet process, G, is expressed as

$$G(\mu) = \sum_{k=1}^{\infty} \pi_k \delta_{\mu_k}. \tag{7.9}$$

Here, G is a discrete distribution with atoms located at the μ_k locations, each with height π_k. A particular atom is then drawn for each of the N data points according to

$$\tilde{\mu}_i \sim G(\tilde{\mu}_i), \quad \forall i \in \{1, \ldots, N\}. \tag{7.10}$$

Finally, each data point is drawn according to

$$x_i \sim \mathcal{N}(x_i; \tilde{\mu}_i, \Sigma^x), \quad \forall i \in \{1, \ldots, N\}. \tag{7.11}$$

The model presented in Equation (7.1)–(7.8) departs from this traditional interpretation by allowing the parameters to change *jointly* in space according to the Gaussian process, g. More precisely, the generative process in the presented model draws a particular atom *conditioned* on g according to

$$\tilde{\mu}_i \sim G(\tilde{\mu}_i - g_i), \quad \forall i \in \{1, \ldots, N\}. \tag{7.12}$$

Each data point is then drawn similarly to the traditional DPMM from Equation (7.11). One can therefore view each pixel as being drawn from an (infinite) mixture of Gaussian distributions whose means are shifted jointly (and smoothly) over space by a Gaussian process. As such, we refer to this type of model as the spatially-varying Dirichlet process Gaussian mixture model (SV-DPGMM).

Related models that change parameters in some independent dimension have been proposed in the nonparametric Baysian statistics literature before. For example, Lin et al. [81] constructs Dependent Dirichlet Processes where, in addition to the birth and death of atoms, atom locations are allowed to vary with time. We note that the main difference between the SV-DPGMM and previous DPMM extensions like [81] is that in the SV-DPGMM, the parameters evolve *jointly* instead of *independently* for each mixture component. This joint evolution is exactly the quantity of interest. Although inference is slightly more complicated, we now develop a method that efficiently marginalizes over multiple quantities.

■ 7.3 Posterior Inference

One motivation for generative models is that computation of marginal event probabilities are generally more robust to noise as compared to point estimates such as the maximum *a posteriori* (MAP) estimate. Consequently, we resort to methods that reason over the full distribution described by the SV-DPGMM rather than using optimization

approaches. MCMC methods, such as Gibbs sampling or the Metropolis-Hastings algorithm, are commonly used in complex probabilistic models. We now develop similar MCMC inference techniques for the SV-DPGMM.

We first introduce some useful notation. All covariance matrices are denoted by Σ, possibly superscripted by an associated random variable. The corresponding precision matrices are denoted by $\Lambda \triangleq \Sigma^{-1}$. Additionally, we use $i, j \in \{1, \ldots, N\}$ to denote pixel indices, $k, l \in \{1, \ldots, K\}$ to denote cluster indices, and $m, n \in \{1, 2, 3\}$ to denote color channel indices. While a DP prior describes an infinite number of mixture elements, there are only a finite number instantiated for any realization of z. We denote the number of instantiated components by K. The following sections derive relevant posterior distributions for each of the latent variables. As the expressions are somewhat complex, we break up the explanation into three sections leading to the final inference algorithm.

■ 7.3.1 Iterative Posteriors Inference without Marginalization

The SV-DPGMM simplifies to a traditional DPGMMM conditioned on the Gaussian process, g. In this paper, we exploit the DP Sub-Cluster sampling method presented in Chapter 5. It is a trivial extension to show that the restricted sampling in the DP Sub-Cluster algorithm results in alternating between the following steps

$$\pi \sim \mathrm{Dir}(\pi \,;\, N_1, \ldots, N_K, \alpha), \tag{7.13}$$

$$\mu_k \sim \mathcal{N}\big(\mu_k \,;\, \theta^*(x_{\{k\}}), \Sigma^{\mu*}(x_{\{k\}})\big)\,, \quad \forall k \in \{1, \ldots, K\} \tag{7.14}$$

$$\Sigma^x \overset{\propto}{\sim} \sum_{u=1}^{|S_\Sigma|} p(x|\mu, z, g, \Sigma^x = S_\Sigma(u)), \tag{7.15}$$

$$z_i \overset{\propto}{\sim} \sum_{k=1}^{K} \mathbb{1}[z_i = k]\, \pi_k \mathcal{N}(x_i \,;\, \mu_k + g_i, \Sigma^x)\,, \quad \forall i \in \{1, \ldots, N\} \tag{7.16}$$

where N_k counts the number of pixels assigned to cluster k, and θ^* and $\Sigma^{\mu*}$ denote posterior hyper-parameters that are functions of the data through the conjugate prior. The use of a global covariance, Σ^x, constitutes a minor departure from the usual formulation of DPGMMs. Due to the uniform prior over a discrete set (see Equation (7.8)), the posterior distribution is equivalent to weighting each possible value with the likelihood, as reflected in Equation (7.15).

Conditioned on the cluster assignments, z, and cluster parameters, μ, the posterior on g is known to be Gaussian with the following distribution (cf. [100]):

$$p(g|\mu, z, \Sigma^x, x) = \mathcal{N}\big(g \,;\, \Sigma^g \Lambda^{g+x}(x - \mu_z), \Sigma^g - \Sigma^g \Lambda^{g+x} \Sigma^g\big)\,, \tag{7.17}$$

where $(\Lambda^{g+x})^{-1} = \Sigma^{g+x} \triangleq (\Sigma^g + \Sigma^x \otimes \mathrm{I}_N)$ is defined for convenience, $\otimes$ denotes the Kronecker product, and I_N denotes an $N \times N$ identity matrix. We note that $\Sigma^x \otimes \mathrm{I}_N$ is a $3N \times 3N$ block diagonal matrix where each 3×3 block represents the observation

covariance for a 3-channel, colored pixel. When a stationary covariance kernel is used for the Gaussian process, Sampling from Equation (7.17) is well approximated using equivalent kernel methods [110]. One expects the approximation to suffer near image boundaries, but we have observed that padding the boundaries with the nearest values *prior* to filtering mitigates this potential issue (details in the Section 2.9.1).

Equations (7.13)–(7.17) analytically describe the conditional posterior distributions of all latent variables. One possible method to perform posterior inference then alternate between sampling these expressions. Such a procedure is outlined in Algorithm 7.1 and is very closely related to the optimization method presented in [39]. The main difference is that one may solve Equation (7.17) analytically, while the authors of [39] employ a significant number of conjugate gradient iterations. Regardless, this approach tends to converge to local extrema and is very sensitive to the initialization. While the algorithm of [39] chooses the best initialization from multiple restarts, it is preferable to have a procedure that does not rely heavily on initial values.

Algorithm 7.1 SV-DPGMM Iterative Inference via MCMC

1. Initialize z and g to be all 0.
2. Sample $(z, \mu, \Sigma^x | g, x)$ using the DP Sub-Cluster algorithm of Chapter 5.
3. Sample $(g | \mu, \Sigma^x, z, x)$ from Equation (7.17) using equivalent kernel [110] techniques.
4. Repeat from Step 2 until convergence.

■ 7.3.2 Marginalized Posterior Inference

Both the reflectance, μ, and shading, g, contribute additively (in the log domain). Consequently, large errors in one can be incorrectly explained by the other. In Bayesian inference such problems are addressed by treating one variable as a nuisance parameter for the other and then marginalizing out the nuisance parameter. While this is often computationally burdensome, there is an analytic expression for the marginalization of the shading image in the SV-DPGMM. Consider the joint model-likelihood conditioned on cluster assignments, $p(x, \mu, g | z, \Sigma^x)$. Because each conditional probability in the generative model is Gaussian, the joint distribution must be jointly Gaussian. As such, any marginal or conditional distribution must also be Gaussian. With some cumbersome algebra, we show in Appendix D that $p(\mu | z, \Sigma_x, x)$ can be analytically expressed as

$$p(\mu | z, \Sigma^x, x) = \mathcal{N}(\mu \; ; \; \theta^*, \Sigma^*), \tag{7.18}$$

where each element of the mean and precision ($\Lambda^* = (\Sigma^*)^{-1}$) is defined as

$$\Lambda^*_{km,ln} = \Lambda^{\mu}_{m,n} + \sum_{i\in\mathcal{I}_k}\sum_{j\in\mathcal{I}_l}\Lambda^{g+x}_{im,jn}, \qquad \forall k = l, \tag{7.19}$$

$$\Lambda^*_{km,ln} = \sum_{i\in\mathcal{I}_k}\sum_{j\in\mathcal{I}_l}\Lambda^{g+x}_{im,jn}, \qquad \forall k \neq l, \tag{7.20}$$

$$[\Lambda^*\theta^*]_{km} = [\Lambda^{\mu}\theta]_m + \sum_{i\in\mathcal{I}_k}\sum_{j}\sum_{n=1}^{3} x_{jn}\Lambda^{g+x}_{im,jn}, \tag{7.21}$$

where $\mathcal{I}_k \triangleq \{i; z_i = k\}$ denotes the set of pixels that have label k.

We note that Equations (7.18)–(7.21) define a system of $3K$ linear equations that express the posterior on reflectance colors, and differ from Equation (7.14) by *marginalizing* over the shading image. This modification to the inference procedure avoids dependence on (possibly erroneous) estimates of g. In its current form, the inference procedure requires the inversion of Σ^{g+x}, a large $3N \times 3N$ matrix, which represents a substantial computational burden. Because we enforce the Gaussian process to use a stationary covariance kernel on a square grid, the covariance matrix, Σ^{g+x}, will be Toeplitz and we can exploit equivalent kernel methods [110]. In the limit as the domain of observations extends to infinity, the inverse covariance will also be Toeplitz, but in finite data regimes, approximating the precision as Toeplitz degrades near boundaries. If such an approximation was used, Equations (7.19)–(7.21) can be expressed as convolutions and fast computations can be performed in the Fourier domain. In practice, we find that this approximation does not work well. Consequently, we consider an alternative.

We note that the system of equations in Equations (7.19)–(7.21) only contain $4.5(K^2 + K)$ variables estimated from approximately N^2 variables. We remind the reader that K is the number of clusters (typically less than 10) and N is the number of pixels (typically more than 50,000). As such, there are many more observations than are necessary to reliably categorize θ^* and Λ^*. We therefore approximate the posterior on μ from a *subset* of the data. The subset of data is chosen to ensure that each cluster has at least 10 pixels and there are a total of at least 1,000 pixels.

Denoting the subset of pixel indices as $\mathcal{S}$, we then define a new realization of the GP on the subset of indices as $g_{\mathcal{S}}$, which has the following distribution

$$p(g_{\mathcal{S}}) = \mathcal{N}(g_{\mathcal{S}}\ ;\ 0, \Sigma^{g_{\mathcal{S}}}). \tag{7.22}$$

Following the same formulation as above, we approximate the posterior on the mean colors as

$$p(\mu|z, \Sigma^x, x) \approx p(\mu|z_{\mathcal{S}}, \Sigma^x, x_{\mathcal{S}}) = \mathcal{N}(\mu\ ;\ \hat{\theta}^*, \hat{\Sigma}^*), \tag{7.23}$$

where the approximate mean and precision are defined as

$$\hat{\Lambda}^*_{km,ln} = \Lambda^{\mu}_{m,n} + \sum_{i \in \mathcal{I}_k \cap \mathcal{S}} \sum_{j \in \mathcal{I}_l \cap \mathcal{S}} \Lambda^{g_{\mathcal{S}}+x}_{im,jn}, \qquad \forall k = l, \tag{7.24}$$

$$\hat{\Lambda}^*_{km,ln} = \sum_{i \in \mathcal{I}_k \cap \mathcal{S}} \sum_{j \in \mathcal{I}_l \cap \mathcal{S}} \Lambda^{g_{\mathcal{S}}+x}_{im,jn}, \qquad \forall k \neq l, \tag{7.25}$$

$$[\hat{\Lambda}^* \hat{\theta}^*]_{km} = [\Lambda^{\mu}\theta]_m + \sum_{i \in \mathcal{I}_k \cap \mathcal{S}} \sum_{j \in \mathcal{S}} \sum_{n} x_{j,n} \Lambda^{g_{\mathcal{S}}+x}_{im,jn}. \tag{7.26}$$

Due to the subsampling process, $\Lambda^{g_{\mathcal{S}}+x} = (\Sigma^{g_{\mathcal{S}}} + \Sigma^x \otimes \mathrm{I}_{|\mathcal{S}|})^{-1}$ can be computed efficiently. We note that this approximation performs well in practice. The resulting inference procedure is detailed in Algorithm 7.2.

Algorithm 7.2 SV-DPGMM Marginalized Inference via MCMC

1. Initialize z and g to be all 0.
2. Sample $(z, \mu, \Sigma^x | g, x)$ using the DP Sub-Cluster algorithm.
3. Sample $(\mu | \Sigma^x, z, x)$, marginalizing out g, from Equation (7.23).
4. Sample $(g | \mu, \Sigma^x, z, x)$ from Equation (7.17) using equivalent kernel [110] techniques.
5. Repeat from Step 2 until convergence.

■ 7.3.3 Marginalized Split/Merge Posterior Inference

The previous sections detail relevant posterior distributions for the latent variables and presented two potential ways of performing inference. The latter of these methods samples μ while marginalizing over g. In this section, we describe an improved procedure that samples z while marginalizing out both μ and g. As mentioned previously, we exploit the DP Sub-Cluster algorithm presented in Chapter 5 to sample from the posterior of z. In particular, the split and merge proposals formed by the sub-clusters can be performed with the desired marginalization of μ and g. We can think of the marginalized split move as determining whether there are two distinct colors in a given subset of pixels, and the marginalized merge move as determining whether two regions should actually be represented with a single color, but that do so without specifying particular colors. These proposed moves make large, global decisions since each plausible labeling integrates over all possible shading images and reflectance colors.

In a similar fashion to the marginalization of the shading image g, we show in Appendix D that a similar derivation can be used to express $p(x|z, \Sigma^x)$ as

$$\begin{aligned} p(x|z, \Sigma^x) &= \frac{|\Lambda^{g+x}|^{\frac{1}{2}} |\Lambda^{\mu}|^{\frac{K}{2}}}{(2\pi)^{\frac{3N}{2}} |\Lambda^*|^{\frac{1}{2}}} \exp\left[\frac{1}{2}\left(\theta^{*\top}\Lambda^*\theta^* - K\theta^{\top}\Lambda^{\mu}\theta - x^{\top}\Lambda^{g+x}x\right)\right] \\ &\propto |\Lambda^*|^{-\frac{1}{2}} \exp\left[\frac{1}{2}\theta^{*\top}\Lambda^*\theta^*\right], \end{aligned} \tag{7.27}$$

where the dependence on z and Σ^x are implied in the definitions of θ^* and Λ^* in Equations (7.19)–(7.21). We note that one must be careful when computing this expression due to the sub-sampling procedure in estimating θ^* and Λ^*. In practice, we simply scale the terms so that the likelihood of the subset of indices has the same dimensionality as the full set of data. One can then accept a split of cluster $\natural$ into clusters $\flat$ and $\sharp$ with

$$H^{\text{SV-DPGMM}}_{\text{split-}\natural} = \frac{\alpha\Gamma(N_\flat)\Gamma(N_\sharp)}{\Gamma(N_\flat + N_\sharp)} \cdot \frac{p(x|\hat{z}, \Sigma^x)}{p(x|z, \Sigma^x)} \prod_{i \in \mathcal{I}_\natural} \frac{\tilde{\pi}_\flat \mathcal{N}(x_i\,;\,\tilde{\mu}_\flat, \Sigma^x) + \tilde{\pi}_\sharp \mathcal{N}(x_i\,;\,\tilde{\mu}_\sharp, \Sigma^x)}{\tilde{\pi}_{\hat{z}_i} \mathcal{N}(x_i\,;\,\tilde{\mu}_{\hat{z}_i}, \Sigma^x)}, \tag{7.28}$$

where $\hat{z}$ is the newly split cluster assignments, and $\tilde{\pi}$ and $\tilde{\mu}$ are the temporary cluster parameters constructed from the sub-clusters as defined in Section 5.5. Note that the likelihoods, $p(x|z, g, \Sigma^x)$, integrate out the mean parameter.

A similar marginalization scheme can be used when proposing merge moves. The resulting Hastings ratio for a proposed merge of clusters $\flat$ and $\sharp$ into cluster $\natural$ can then be expressed as

$$H^{\text{SV-DPGMM}}_{\text{merge-}\flat\sharp} = \frac{\Gamma(N_\flat + N_\sharp)}{\alpha\Gamma(N_\flat)\Gamma(N_\sharp)} \cdot \frac{p(x|\hat{z}, \Sigma^x)}{p(x|z, \Sigma^x)} \prod_{i \in \mathcal{I}_\flat \cup \mathcal{I}_\sharp} \frac{\tilde{\pi}_{z_i} \mathcal{N}(x_i\,;\,\tilde{\mu}_{z_i}, \Sigma^x)}{\tilde{\pi}_\flat \mathcal{N}(x_i\,;\,\tilde{\mu}_\flat, \Sigma^x) + \tilde{\pi}_\sharp \mathcal{N}(x_i\,;\,\tilde{\mu}_\sharp, \Sigma^x)}. \tag{7.29}$$

The steps in this sampling procedure, which proposes marginalized splits and merges, are summarized in Algorithm 7.3.

Algorithm 7.3 SV-DPGMM Marginalized Split/Merge Inference via MCMC

1. Initialize z and g to be all 0.
2. Run the restricted sampling algorithm of DP Sub-Cluster to find likely splits conditioned on g (Σ^x is concurrently sampled within DP Sub-Clusters).
3. Sample $(z|\Sigma^x, x)$ by proposing all splits or merges and accept with the Hastings ratios in Equations (7.28) and (7.29).
4. Sample $(\mu|\Sigma^x, z, x)$ marginalizing over g from Equation (7.23).
5. Sample $(g|\mu, \Sigma^x, z, x)$ from Equation (7.17) using equivalent kernel [110] techniques.
6. Repeat from Step 2 until convergence.

■ 7.4 Parameter Learning

We now present two methods for learning parameters of the model. The first is a supervised approach that uses training data to find the set of parameters that works best across all training examples. The second is an unsupervised approach that places hyper-priors on the parameters.

We remind the reader that the only parameters to set in the model are those of

the covariance kernel in the Gaussian process, g. The Matérn class of kernels can be expressed as:

$$\kappa(r\ ;\ \sigma_g^2, \nu, l) = \sigma_g^2\ \frac{2^{1-\nu}}{\Gamma(\nu)}\Big(\frac{r\sqrt{2\nu}}{l}\Big)^{\nu} K_\nu\Big(\frac{r\sqrt{2\nu}}{l}\Big), \tag{7.30}$$

where r is the change in 2D location, K_ν is a modified Bessel function of the second kind, and $\{\sigma_g^2, \nu, l\}$ are the set of hyper-parameters we wish to learn. Note that l here should not be confused with the cluster label. Here, it refers to the characteristic length-scale of the covariance kernel. Additionally, as mentioned previously, allowing for small amounts of color in the shading images improves convergence. As such, we supplement the Matérn class kernel with the following

$$\kappa(c, r\ ;\ \sigma_c, \sigma_g^2, \nu, l) = \sigma_c^{\mathbb{1}[c\neq 0]}\ \sigma_g^2\ \frac{2^{1-\nu}}{\Gamma(\nu)}\Big(\frac{r\sqrt{2\nu}}{l}\Big)^{\nu} K_\nu\Big(\frac{r\sqrt{2\nu}}{l}\Big), \tag{7.31}$$

where c is the change in the color channel, and σ_c is an additional hyper-parameter to learn. The set of all parameters is then $\lambda_g = \{\sigma_c, \sigma_g^2, \nu, l\}$.

■ 7.4.1 Supervised Learning

In the following sections, we will focus on analyzing results on the MIT Intrinsic Image Dataset [48]. Unfortunately, because the 20 images from [48] were released in two batches, some published methods only run their algorithm on a subset of the images. For example, the results from [39] use 16 of the 20 images, while the results from [4] use all 20 images. Furthermore, each method uses different training and test sets; [39] performs leave-one-out-cross-validation (LOOCV), while [4] separates the set into 10 training images and 10 test images. For an accurate comparison, we learned separate parameters using LOOCV and the separate training/test sets used in [4]. For each image, we ran the inference algorithm under a discrete set of parameter choices. The set of parameters that minimized the arithmetic mean of *RS*-MSE was chosen (similar to [39]). This error metric will be described in more detail in Section 7.6.

■ 7.4.2 Unsupervised Learning

Because the presented model is formulated in a Bayesian framework, an alternative approach for *unsupervised* learning is to place an additional prior on the parameters, λ_g, and explicitly infer them. More precisely, we place a prior on λ_g that is uniformly distributed over a discrete set of plausible values. Inference then proceeds in the same sequence as before, with the added sampling step of

$$\lambda_g \overset{\propto}{\sim} p(\lambda_g)p(g|\lambda_g) \propto p(g|\lambda_g). \tag{7.32}$$

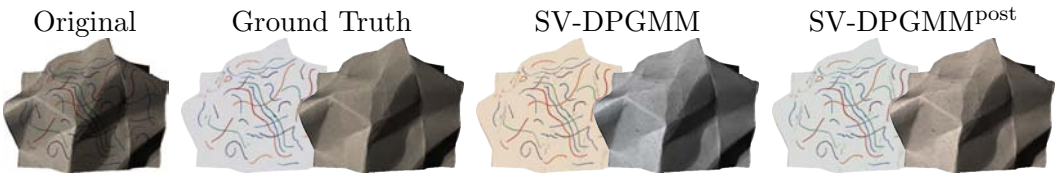


Figure 7.4: An example of correcting color constancy as a post processing step.

This requires computing the likelihood of a Gaussian process realization with parameters λ_g. For the finite realization of interest, this term can be expressed as

$$p(g|\lambda_g) = |\Sigma^g|^{-\frac{1}{2}} (2\pi)^{-\frac{N}{2}} \exp\left[-\frac{1}{2} g^\top (\Sigma^g)^{-1} g\right]. \tag{7.33}$$

Two of these terms, $|\Sigma^g|$ and $g^\top(\Sigma^g)^{-1}g$ pose computational issues for large data sizes that are encountered in images. Fortunately, as discussed in Section 2.9.1, $|\Sigma^g|$ can be approximated using equivalent kernel methods and the determinant can be approximated with circulant matrix determinants. These approximations perform well when the size of the matrix is large and the covariance kernel decays quickly, both of which are satisfied in the problem of interest.

■ 7.5 Post-Processing for Color Constancy

Since the goal is to infer multiple latent variables for each observed pixel, intrinsic image decomposition is inherently an ill-posed problem. While regularizations such as Gaussian processes and DPGMMs restrict the solution space, there is one ambiguity that has not been explicitly addressed; any color channel of the log-shading image can be shifted by an arbitrary amount if the same color channel of the log-reflectance image is shifted by the negative of the same amount. The resulting decomposition will still produce the exact same image. For example, this could correspond to changing the color of the light from white to blue, and adding a yellow tint to reflectance image.

The SV-DPGMM approach implicitly restricts these ambiguities. Because the Gaussian process is assumed to be zero-mean with correlated color channels, the shading image largely favors white lights and grayscale shading images. This is undesirable in many situations such as the images in the MIT Intrinsic Image Dataset. We show one such example in Figure 7.4.

Barron and Malik [4] address this issue of color constancy by placing an explicit prior over absolute log reflectance values and by assuming a specific lighting model (spherical harmonic illuminations). It is difficult to incorproate an explicit lighting model without modeling the 3D surfaces as well. We take a slightly different approach here. In each type of cross-validation, we learn the distribution of the log shading and log reflectance values from the ground truth on the training data via a kernel density estimate. Example distributions are shown in Figure 7.5. It would be ideal if these

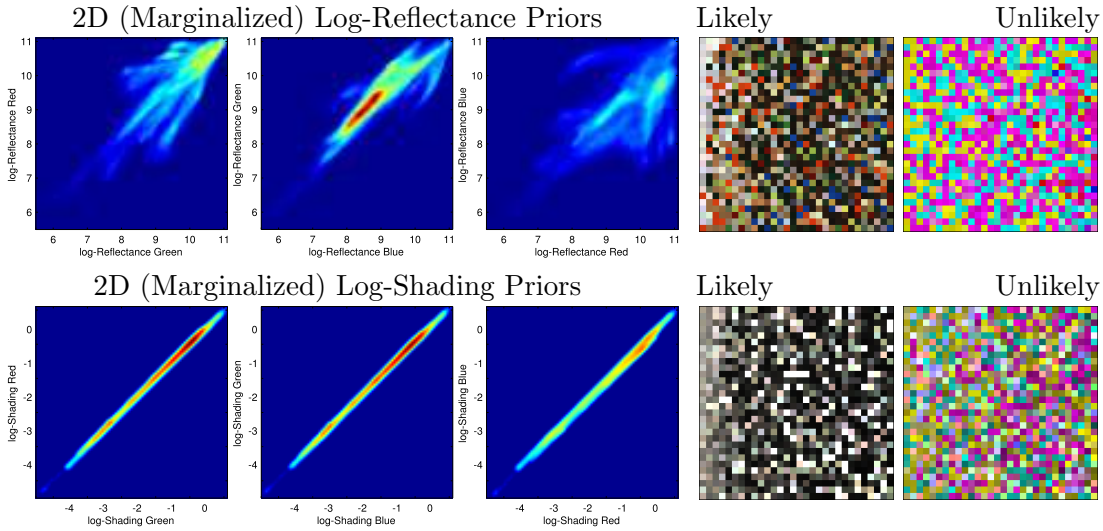


Figure 7.5: Kernel density estimates for the prior log-reflectance and log-shading. The images on the right show colors ranked from likely (left) to unlikely (right) according to the learned distributions.

distributions could be incorporated into the generative model, but the non-parametric nature of the distributions eliminate the conjugacy that was exploited in the inference. As such, we perform a post-processing step that corrects for the ambiguity in the colors. The following procedure can be used in any intrinsic image algorithm to correct for color constancy.

Let $f_s(s_i)$ and $f_r(r_i)$ denote the learned 3D prior distributions on the log-shading and log-reflectance colors, respectively. We estimate these distributions by binning the kernel density estimates into M bins for each dimension, resulting in $f_s, f_r \in \mathbb{R}^{M\times M\times M}$. Given estimated log-shading and log-reflectance images, s and r, and denoting the bins corresponding to pixel i by $s_i, r_i \in \mathcal{M} \triangleq \{1,\ldots,M\}^3$, we aim to maximize

$$\Delta^* = \arg\max_{\Delta} \prod_{i=1}^{N} f_s(s_i+\Delta) f_r(r_i-\Delta), \tag{7.34}$$

where $\Delta^* \in \mathbb{R}^3$ is the optimal 3D constant color shift applied to both the log-shading and log-reflectance images. This optimization can be expressed as

$$\begin{aligned}\Delta^* &= \arg\max_{\Delta} \sum_{i=1}^{N} \log f_s(s_i+\Delta) + \log f_r(r_i-\Delta) \\ &= \arg\max_{\Delta} \sum_{m\in\mathcal{M}} N_s(m)\log f_s(m+\Delta) + N_r(m)\log f_r(m-\Delta),\end{aligned} \tag{7.35}$$

where $N_s(m)$ and $N_r(m)$ denote the number of pixels in the shading and reflectance image assigned to bin m. This optimization can be computed exhaustively (but efficiently) via FFTs since each term is essentially a 3D convolution.

■ 7.6 Experimental Results

We now present some results obtained on real images using the SV-DPGMM. We focus our experiments on the MIT Intrinsic Image dataset [48], which contains ground truth reflectance and shading components for 20 images. For each image, we simulate the Markov chain until convergence. We then take the mean of 25 samples from the stationary distribution. The reflectance image estimate is obtained with $\exp[\langle \mu_z \rangle]$ and the shading image estimate with $\exp[\langle g \rangle]$, where $\langle \bullet \rangle$ denotes the mean over the 25 samples. Since the simulated Markov chains tend to explore a local mode within the 25 iterations, we run 10 chains independently and show the resulting pixel-wise median shading and reflectance images over the 10 independent chains. We can think of each chain as finding the local mean shading and reflectance, and then using the median of the 10 independent chains to find the mean that is in the middle. As we soon show, while this procedure slightly improves results, running a single chain still achieves state-of-the-art results.

In the following section, we compare our algorithm with Retinex, and the two state-of-the-art methods from [4] and [39]. For an accurate comparison to each method, we train the model parameters using the same training and test sets described in each of the previous methods. For each algorithm, we compute three different metrics from [4] and [48] R-MSE, S-MSE, and RS-MSE. R-MSE and S-MSE compute the global scale-invariant reflectance and shading mean squared error, respectively. We first define a scale-invariant mean squared error as

$$\text{MSE}_{\text{SI}}(\hat{x}, x^*) = \min_{\alpha} \frac{1}{N} \sum_{i=1}^{N} \|\alpha \hat{x_i} - x_i^*\|_2^2, \tag{7.36}$$

where $\hat{x}$ denotes a predicted image, x^* denotes a ground-truth image, and N counts the number of pixels. The R-MSE and S-MSE can then be expressed as

$$R\text{-MSE}(\hat{r}, r^*) = \text{MSE}_{\text{SI}}(\hat{r}, r^*) \tag{7.37}$$

$$S\text{-MSE}(\hat{s}, s^*) = \text{MSE}_{\text{SI}}(\hat{s}, s^*). \tag{7.38}$$

Errors are computed in the image domain versus log-image domain.

RS-MSE is the metric from [48], which computes the average of local scale-invariant MSEs. Given a window of pixels (e.g., 20×20) denoted by w, RS-MSE finds the optimal scaling for each window of the images. Defining the set of all windows in an image to

Table 7.2: Comparing SV-DPGMM Inference Methods

	S-MSE	R-MSE	RS-MSE	gS-MSE	gR-MSE	gRS-MSE
SV-DPGMM$^{\text{it1}}$	0.0548	0.0309	0.0362	0.0202	0.0196	0.0205
SV-DPGMM$^{\text{it2}}$	0.0532	0.0238	0.0302	0.0193	0.0146	0.0181
SV-DPGMM$^{\text{marg1}}$	0.0300	0.0146	0.0248	0.0097	0.0085	0.0121
SV-DPGMM$^{\text{marg2}}$	0.0321	0.0175	0.0271	0.0106	0.0109	0.0154
SV-DPGMM	0.0321	0.0144	0.0239	0.0093	0.0078	0.0111
SV-DPGMM$^{\text{opt}}$	0.0352	0.0172	0.0286	0.0120	0.0104	0.0157

be $\mathcal{W}$, the local scale-invariant MSE for two images can be expressed as

$$\text{LMSE}_{\text{SI}}(\hat{x}, x^*) = \sum_{w \in \mathcal{W}} \text{MSE}_{\text{SI}}(\hat{x}_w, x^*_w). \tag{7.39}$$

The RS-MSE can then be expressed as the average of the local scale-invariant MSEs for each color channel of the shading and reflectance, normalized so a predicted image of all zeros produces an error of 1:

$$RS\text{-MSE}(\hat{r}, r^*, \hat{s}, s^*) = \frac{1}{2}\left(\frac{\text{LMSE}_{\text{SI}}(\hat{r}, r^*)}{\text{LMSE}_{\text{SI}}(0, r^*)} + \frac{\text{LMSE}_{\text{SI}}(\hat{s}, s^*)}{\text{LMSE}_{\text{SI}}(0, s^*)}\right) \tag{7.40}$$

These metrics are complementary. The first two metrics measure global performance, whereas the latter measures local performance. Additionally, we compute both the arithmetic and geometric mean (denoted with a preface 'g') across the images. We note that [39, 48] use the arithmetic mean while [4] uses the geometric mean.

■ 7.6.1 Cross-Validation Performance

We first compare different inference algorithms in SV-DPGMMs while performing leave-one-out-cross-validation on the 16 images of the original dataset presented in [48]. We consider the following inference methods: iterative inference via Algorithm 7.1 (SV-DPGMM$^{\text{it1}}$); iterative inference via Algorithm 7.1 but that samples shading first (SV-DPGMM$^{\text{it2}}$); marginalized inference via Algorithm 7.2 (SV-DPGMM$^{\text{marg1}}$); marginalized inference via Algorithm 7.2 but that samples shading first (SV-DPGMM$^{\text{marg2}}$); marginalized split/merge inference via Algorithm 7.3 (SV-DPGMM). Additionally, we consider an optimization-based procedure (SV-DPGMM$^{\text{opt}}$) that follows the same steps as SV-DPGMM, but replaces all sampling steps with an optimization. Furthermore, it accepts any proposed splits or merges with a Hastings ratio larger than 0.5. The comparison of these different inference schemes is summarized in Table 7.2. We see that the four inference methods based on Algorithms 7.1–7.2 are quite sensitive, since their results vary dramatically based on if the shading or reflectance is first estimated. In contrast, Algorithm 7.3 compute these jointly and does not suffer from the same sen-

Table 7.3: Comparing SV-DPGMM Inference Methods

	S-MSE	R-MSE	RS-MSE	gS-MSE	gR-MSE	gRS-MSE
SV-DPGMM$^{\text{unsup}}$	0.0298	0.0166	0.0260	0.0096	0.0098	0.0136
SV-DPGMM$^{\text{single}}$	0.0328	0.0151	0.0249	0.0100	0.0087	0.0124
SV-DPGMM$^{K=10}$	0.0321	0.0147	0.0241	0.0095	0.0083	0.0120
SV-DPGMM	0.0321	0.0144	0.0239	0.0093	0.0078	0.0111
SV-DPGMM$^{\text{post}}$	0.0317	0.0135	0.0239	0.0072	0.0060	0.0111

sitivity. Since the training is only based on RS-MSE, it is reasonable that SV-DPGMM does not perform the best across all metrics.

Next, we consider different variants of the SV-DPGMM model. In particular we consider the following: unsupervised training (SV-DPGMM$^{\text{unsup}}$); supervised training on a single Markov chain (SV-DPGMM$^{\text{single}}$); supervised training and computing the median across 10 Markov chains (SV-DPGMM); and SV-DPGMM with the color constancy post-processing (SV-DPGMM$^{\text{post}}$). Additionally, we compare to a simpler model that uses a finite mixture model with a 10-dimensional Dirichlet distribution prior instead of the Dirichlet process (SV-DPGMM$^{K=10}$). The single Markov chain results were obtained by averaging the *errors* for 10 independent Markov chains, instead of combining the 10 Markov chains with a median image. The comparison of these different variants is summarized in Table 7.3. We see that the unsupervised method generally performs worse than the supervised training. In principle, unsupervised learning has an advantage, in that it yields a set of parameters for each observed image. However, the sample space that includes a prior on the GP covariance kernels may be too difficult to sufficiently explore. Furthermore, combining multiple chains, using a Dirichlet process, and post-processing to enforce color constancy all improve results. We note that the RS-MSE and gRS-MSE do not change with post-processing since these metrics are invariant to global shifts in any color channel.

Next, we compare SV-DPGMM$^{\text{post}}$ to the following methods in Table 7.4: the Retinex algorithm; the method of [39] without Retinex ([39]−Ret.); and the method of [39] with Retinex ([39]+Ret.). We see that SV-DPGMM outperforms all methods in Table 7.4. The arithmetic mean of the shading MSE is the only metric on which the SV-DPGMM yields worse performance. Upon examination of the actual results, we have found that this abnormally high error is due to making a large error in the shading estimate on one of the 16 images. This result is confirmed in the geomertric mean of S-MSE, which is less affected by large errors. Since the training was only based on the RS-MSE, it is understandable that SV-DPGMM does not perform the best across all metrics. However, it is encouraging that it is generally better than all previous methods. We remind the reader that the only differences between SV-DPGMM and [39]−Ret. are the incorporation of the Dirichlet process, a more expressive shading smoothness, and more robust, marginalized inference. Moreover, many of the simplified

Table 7.4: Leave-One-Out-Cross-Validation on 16 images from [48]

	S-MSE	R-MSE	RS-MSE	gS-MSE	gR-MSE	gRS-MSE
Retinex	0.0400	0.0292	0.0297	0.0219	0.0225	0.0185
[39]−Ret.	0.0311	0.0172	0.0304	0.0107	0.0134	0.0156
[39]+Ret.	0.0287	0.0205	0.0277	0.0119	0.0150	0.0166
SV-DPGMM$^{\text{post}}$	0.0317	0.0135	0.0239	0.0072	0.0060	0.0111

Table 7.5: Separate Train/Test Validation on 20 images from [48]

	S-MSE	R-MSE	RS-MSE	gS-MSE	gR-MSE	gRS-MSE
SIRFS Reported	-	-	-	0.0064	0.0098	0.0125
SIRFS Locally Run	0.0201	0.0158	0.0247	0.0068	0.0115	0.0125
SV-DPGMM	0.0306	0.0148	0.0229	0.0113	0.0092	0.0136
SV-DPGMM$^{\text{post}}$	0.0303	0.0141	0.0229	0.0092	0.0074	0.0136

inference algorithms described in Tables 7.2–7.3 also outperform current methods. We note that our optimization procedure of a more expressive model is only comparable to [39]. We believe this is due to the particular realization converging to a local extrema. Methods such as [39] circumvent these issues by choosing the best result from multiple randomized initializations.

Table 7.5 compares the algorithm when trained on half the images and tested on the other half, as was done [4]. We compare results with the algorithm from [4], called SIRFS. We note that the numbers from SIRFS were obtained by running their publicly available source code and are slightly different than those originally reported in [4]. Regardless, SV-DPGMM performs better in three of the six metrics, but has the advantage of being simpler while avoiding the need to model the 3D scene geometry.

We visualize results from each algorithm in Figure 7.6. All results except those from SIRFS were obtained using LOOCV. This is somewhat of an unfair comparison since the LOOCV has more training examples for each test image. On the other hand, half of the results obtained using SIRFS were actually in their training set. In general, the reflectance image obtained from SV-DPGMM exhibits superior piecewise color constancy and has less bleeding of reflectance into the shading images. We note that the post-processing to fix color constancy does not always improve results, as shown in the first image of Figure 7.6. However, the previous numerical comparisons show that it generally improves the result. Furthermore, SV-DPGMM makes gross errors occasionally, such as the first cup on the second page of Figure 7.6. We suspect that these errors are due to allowing color in the shading images and that an image-specific covariance kernel may correct these errors.

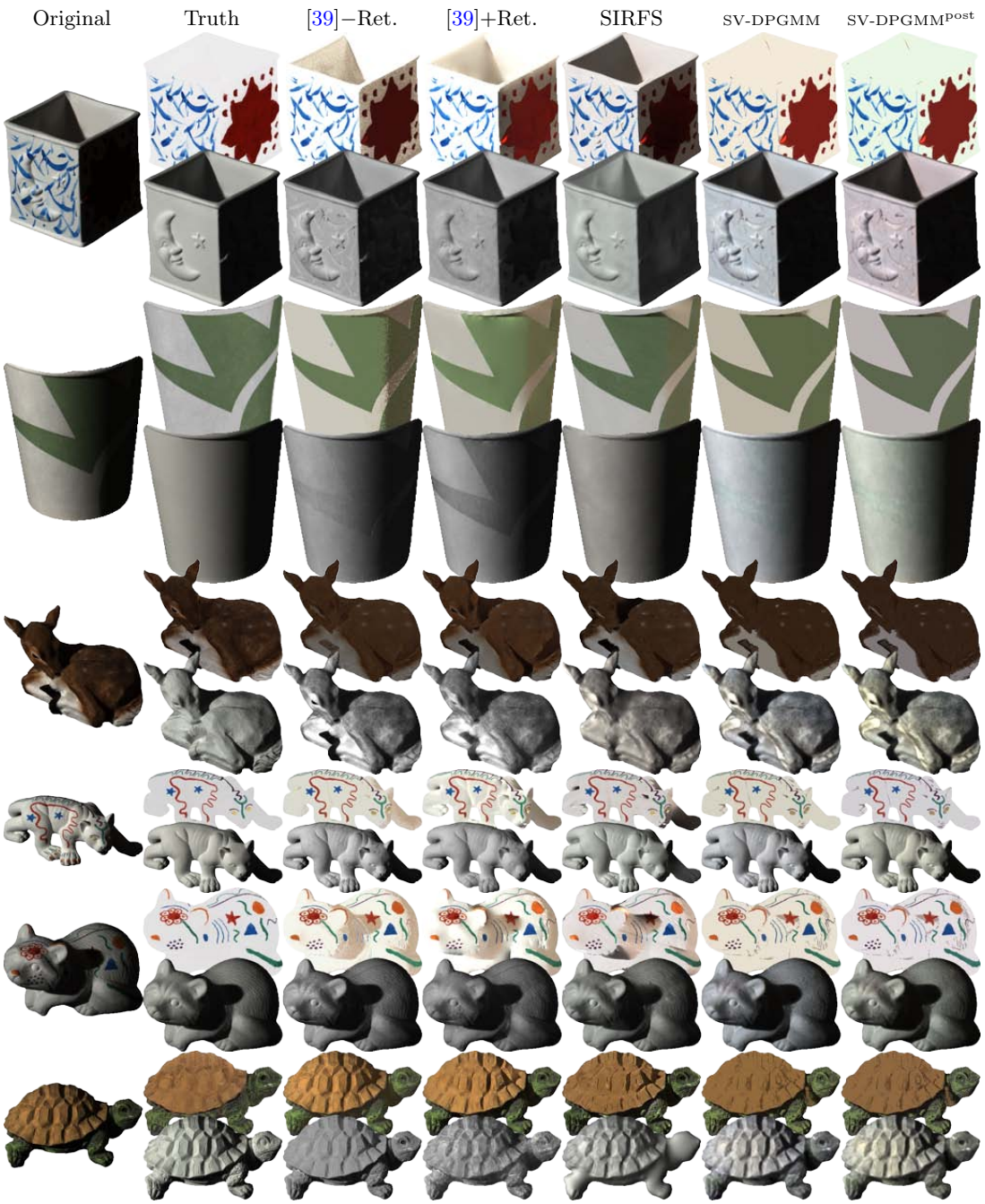


Figure 7.6: Visual comparison of results. The first row for each image is the estimated reflectance image, and the second row is the estimated shading image. [4] is trained via separate train/test sets, and all other algorithms are trained using LOOCV. Consequently, comparisons against algorithms are not completely accurate.

Figure 7.6: *(cont.)* Visual comparison of results. The first row for each image is the estimated reflectance image, and the second row is the estimated shading image. [4] is trained via separate train/test sets, and all other algorithms are trained using LOOCV. Consequently, comparisons against algorithms are not completely accurate.

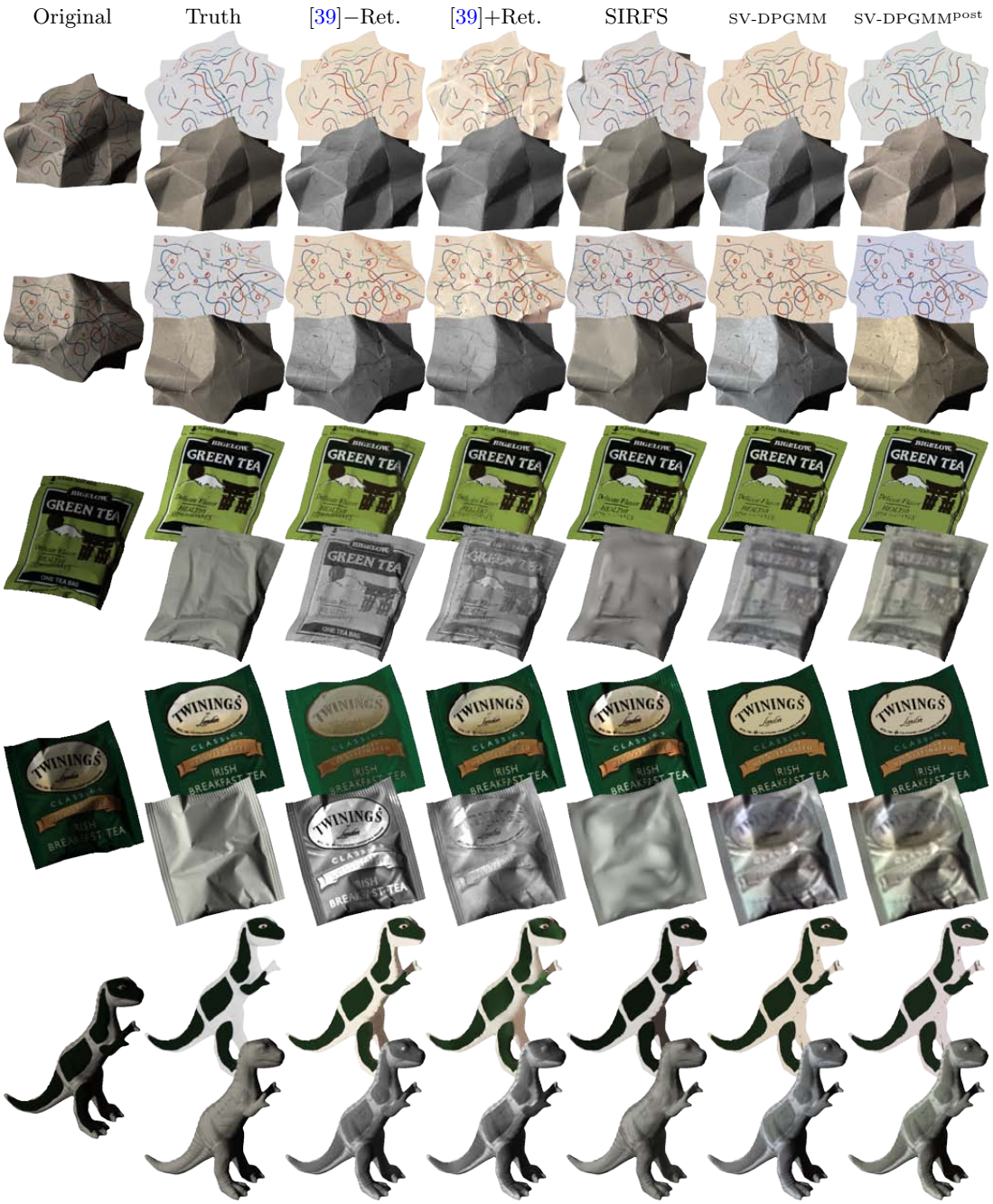


Figure 7.6: *(cont.)* Visual comparison of results. The first row for each image is the estimated reflectance image, and the second row is the estimated shading image. [4] is trained via separate train/test sets, and all other algorithms are trained using LOOCV. Consequently, comparisons against algorithms are not completely accurate.

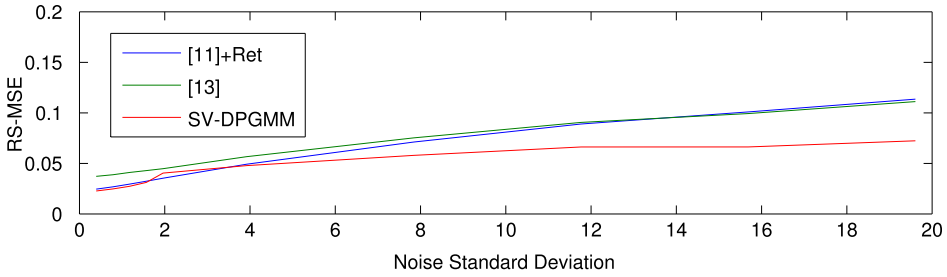


Figure 7.7: Performance with additive noise.

■ 7.6.2 Sensitivity to Noise

Lastly, we consider the case of noisy observations. Images from [48] do not have any camera noise, so we inject artificial additive Gaussian noise in the observed image. We note that this synthetic noise does not contain the same noise characteristics assumed in SV-DPGMM, which models Gaussian noise in the *log* domain. Results for varying levels of noise variance are shown in Figure 7.7. This plot illustrates that SV-DPGMM, which explicitly characterizes noise, outperforms other methods in the noisy regime even with the model mismatch.

■ 7.7 Discussion

This chapter presents the spatially-varying Dirichlet process Gaussian mixture model, an extension to the DPGMM that allows the mixture parameters to jointly change. The application of intrinsic image decomposition shows that non-parametric Bayesian approaches can be scaled to computer vision if one is clever about the inference. The DP Sub-Cluster algorithm of Chapter 5 enables efficient and effective MCMC inference techniques that marginalize over a large set of variables. Applying the SV-DPGMM to intrinsic image decomposition has been useful in improving results. Furthermore, these results show that the incorporation of a Retinex term is not needed to achieve state-of-the-art results in traditional shading and reflectance decomposition problems.

Chapter 8

Conclusion

This thesis has focused on developing and analyzing MCMC algorithms for two specific types of discrete labeling problems in computer vision and machine learning: the Markov random field and the probabilistic mixture model. We have shown that MCMC algorithms are not only feasible, but that they can also improve results across multiple applications in computer vision. We have developed two overarching frameworks for sampling: the Permutation-based Gaussian-Inspired Metropolis Hastings shape sampling algorithm, and the Dirichlet Process Sub-Cluster algorithm for mixture models. Both algorithms have shown to converge orders of magnitude faster than previous methods and improve by augmenting the sample space with additional auxiliary variables. We now summarize the contributions of this work to each of the addressed problems.

8.1 Contributions to Shape Sampling

The following highlights some of our contributions to the problem of sampling shapes and discrete MRFs.

Discretization of Shape Sampling

One important observation of sampling shapes was the connection of traditionally-defined "implicit shapes" via level-set methods to the discretized lattice grid. In the discretized representation, one can use traditional Gibbs sampling to sample from the posterior distribution of label configurations. Furthermore, priors in level-set methods such as the curve-length penalty can be easily implemented in Gibbs sampling via the approximation described in Chapter 3 and a simple table look-up.

PGIMH Complexity and Generalization of Gibbs Sampling

The developed PGIMH algorithm augments the sample space of labels with an explicit ordering of the pixels. A random block of pixels is then changed conditioned on the ordering. We have drawn interesting ties between PGIMH and blocked Gibbs sampling, and have shown that the computational complexity of PGIMH grows linearly with the block size instead of exponentially like in blocked Gibbs sampling. Furthermore, PGIMH also simplifies to traditional Gibbs sampling when the block is a single pixel.

Arbitrary Topology Constraints in 2D Shapes

Using the PGIMH algorithm, we have shown how one can incorporate arbitrary constraints on 2D topologies. This extends the work of Han [50] and Ségonne [104] which only restrict the topology of the shape or the genus of the shape to remain constant. In contrast, we have shown that any 2D topology constraint, such as the connected-component-preserving constraint, can be efficiently incorporated. Furthermore, to our knowledge, this is the first attempt to embed topology constraints into a probabilistic sampling framework.

Temporal Dynamics on Shapes

We have shown a particular model formulation that incorporates dynamics on the shapes. Furthermore, by exploiting the PGIMH algorithm, we have shown how samples in a particle filter can be propagated without needing to update weights. This also precludes the need to use sequential resampling techniques. The presented layered model has shown to produce state-of-the-art results on the SegTrack dataset [118].

■ 8.2 Contributions to Probabilistic Mixture Models

The following highlights some of our contributions to the problem of inference in mixture models.

Scalable MCMC Algorithms without Model Approximations

The DP Sub-Cluster algorithm pairs a non-ergodic restricted Gibbs sampler with split and merge moves. In addition to being highly parallelizable, the algorithm benefits from satisfying the limiting guarantees of Markov chain theory without needing finite model approximations. We have shown empirically that the DP Sub-Cluster algorithm converges to a better solution, and does so 10–10^3 times faster than previous MCMC methods, and that the resulting distribution is a much more likely sample.

Sub-Cluster Splits and Merges

The mixture model is augmented with auxiliary sub-clusters that are chosen to learn a 2-component mixture model within the confines of each regular cluster. The proposed splits are then constructed from the instantiated sub-clusters. As such, unlike all previous split and merge proposals which construct a proposal on the spot, the DP Sub-Cluster learns likely splits of the data over many iterations. Furthermore, conditioned on the sub-clusters, splits and merges can be proposed in constant time from summary statistics. We have shown that an exact algorithm can be constructed by combining deterministic split/merge moves with randomized, data-independent split/merge moves. The randomized moves can be ignored with a slight approximation to the distribution of the sub-clusters.

Extensions to HDPs

The DP Sub-Cluster is extensible to many different mixture models, both finite and infinite. For example, we have shown that it can be extended to HDPs. We have also found that in topic modeling, topic distributions have significant overlap, and larger moves on the entire corpus are needed. The proposed global splits and merges address this issue and improve convergence where other split/merge algorithms (e.g., [122]) could not. We have additionally motivated the need to inspect metrics besides cross-validation techniques to analyze convergence in HDP MCMC algorithms.

Spatially Varying Dirichlet Process Gaussian Mixture Model

Lastly, we have presented a new nonparametric mixture model, called the spatially-varying Dirichlet process Gaussian mixture model, that allows the cluster parameters to change jointly in space. We have applied the SV-DPGMM to the problem of intrinsic image decomposition. By exploiting the DP Sub-Cluster algorithm, we have developed efficient inference methods that marginalize over a significant number of parameters and produce state-of-the-art results on the MIT Intrinsic Image Dataset [48].

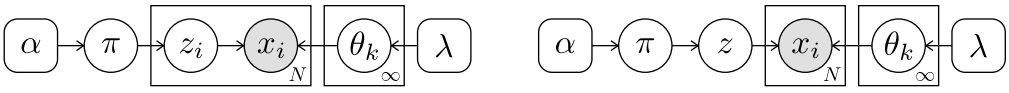
■ 8.3 Future Work

In this thesis, we considered two fairly different labeling problems. While we have listed potential future work for each project at the end of their corresponding chapter, we highlight two other suggestions here that combine work across multiple chapters.

■ 8.3.1 Spatially-Coherent Mixture Models

One potential direction for future work is to combine the MRF model that captures local spatial-coherence with the Dirichlet process as a global prior over label distributions. This idea is related to the distance dependent Chinese restaurant process (dd-CRP) [8] and the region-based hierarchical distance dependent CRPs (rdd-CRP) [41], which both include spatial dependence in a nonparametric mixture model. These approaches typically require sequentially iterating through points because of their constructive definitions. We now discuss an alternative to these approaches.

Consider the graphical model for a DPMM shown in Figure 8.1a. Each z_i conditioned on π and θ is independent as indicated in the plate notation. Alternatively, we can consider the case where the z's have inter-dependence via some Markov random field structure. This relationship is depicted in Figure 8.1b, where z is now the joint N-dimensional random variable. A generative model for this model could, for example,



(a) DPMM (b) DPMM with Spatial-Coherence

Figure 8.1: Graphical models for the DPMM and the a spatially coherent DPMM.

be described by the following process:

$$\pi \sim \text{GEM}(1, \alpha), \tag{8.1}$$

$$z \overset{\propto}{\sim} \prod_{i=1}^{N} \psi_1(z_i) \prod_{i,j \in \mathcal{E}} \psi_2(z_i, z_j), \tag{8.2}$$

$$\theta_k \sim f_\theta(\theta_k; \lambda), \quad \forall k \in \{1, 2, \dots\}, \tag{8.3}$$

$$x_i \sim f_x(x_i; \theta_{z_i}), \quad \forall i \in \{1, \dots, N\}, \tag{8.4}$$

where $\psi_1(\cdot)$ and $\psi_2(\cdot)$ denote generic singleton and pairwise potentials of the MRF and $\mathcal{E}$ denotes the set of edges in the MRF of z. The potential functions can be specified to fit the problem of interest. For example, in segmentation, $\psi_2(z_i, z_j)$ may take on the form of an Ising or Potts model and be

$$\psi_2(z_i, z_j) = \exp\left[\gamma \cdot \mathbb{1}[z_i = z_j]\right]. \tag{8.5}$$

If the singleton potentials, ψ_1 are chosen to be

$$\psi_1(z_i) = \pi_{z_i} = \text{Cat}(z_i; \pi), \tag{8.6}$$

then convenient conjugacy properties of Dirichlet process mixture models still hold. In particular, the posterior on π does not depend on the MRF of z, since it can be expressed as

$$p(\pi|z) \propto p(\pi) \cdot \prod_{i=1}^{N} \text{Cat}(z_i; \pi) \prod_{i,j \in \mathcal{E}} \psi_2(z_i, z_j), \tag{8.7}$$

$$\propto p(\pi) \cdot \prod_{i=1}^{N} \text{Cat}(z_i; \pi), \tag{8.8}$$

$$\propto \text{Dir}(\pi_1, \dots, \pi_K, \pi_{K+1}; N_1, \dots, N_K, \alpha). \tag{8.9}$$

Furthermore, the posterior on the parameters, θ_k, are similarly still decoupled as in the traditional DPMM case.

Inference in this spatially coherent mixture model could then follow in a similar approach as any DPMM sampling algorithm, such as the DP Sub-Cluster algorithm. The only modification that would be needed is when sampling z, one must account for

Figure 8.2: An example image where shading cues give information about object boundaries.

the MRF with an algorithm such as PGIMH. This is one simple way that the PGIMH algorithm and the DP Sub-Cluster algorithm could be combined to spatially coherent applications such as image segmentation.

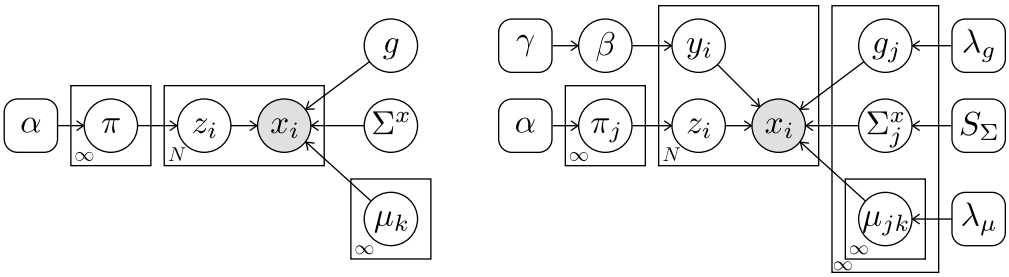
■ 8.3.2 Segmentation via Intrinsic Images

Chapter 7 presented a generative model for an object that is explicitly decomposed into shading and reflectance components. One motivation of using generative probabilistic models is that they are often modular and can be built upon to represent more complicated problems. We view the SV-DPGMM as a model that solves a very specific component of computer vision. Here, we briefly consider building upon the model to perform object segmentation as well.

Consider the image shown in Figure 8.2. Many clues about object boundaries are hidden in the shading image. For example, the legs of the dog darken towards the back of each leg. Each leg of the human also has a very different shading model, both of which are different from the ground, which darkens towards to the top left corner. With an adequate prior smoothness on shading images, one may hope to recover object segmentations based on intrinsic image decompositions.

A rearranged graphical model of the SV-DPGMM for a single object is displayed in Figure 8.3a. We could additionally place another Dirichlet process prior on the object level, as indicated in Figure 8.3b. This Hierarchical SV-DPGMM contains object proportions, $\beta \sim \text{GEM}(1, \gamma)$, and an object label y_i for each pixel. Each object, indexed by j, is then assigned its own SV-DPGMM with its own shading and reflectance image. This model could also impose a Markov random field on object labels, similar to what was mentioned in the previous section.

We suspect that the Hierarchical SV-DPGMM may have trouble when objects do not touch each other, since the ambiguity between the shading and reflectance always



(a) SV-DPGMM (b) Hierarchical SV-DPGMM

Figure 8.3: Graphical models for the SV-DPGMM and the proposed Hiearchical SV-DPGMM.

allows one to explain the other. However, we do believe that this model takes an interesting approach to the classical image segmentation problem.

■ 8.4 Final Thoughts

This thesis attempts to address efficient sampling techniques for two discrete labeling problems in computer vision. While the proposed methods achieve orders of magnitude in speed gains as compared to previous sampling methods, we believe there is still more work to be done. We hope that the frameworks and analysis of the PGIMH and DP Sub-Cluster algorithms can help researchers in making complicated models more accessible and motivate others to improve upon these methods.

Appendix A

Derivations Pertaining to Shape Dynamics

In this appendix, we derive expressions related to Chapter 4. We formally prove the proposition that particle filtering can be done without weight updates, and then develop the approximation used to marginalize over the independent flow.

■ A.1 Particle Filtering without Weight Updates

We now prove Proposition 4.4.1, reproduced below for convenience.

Proposition 4.4.1 : (Particle Filtering without Weight Updates) *Let* $\{z_1^t, \ldots, z_S^t\}$ *be a set of* S *samples, each drawn from some density* $q(z^t)$*, and* w_s^t *be the importance weight for sample* s *such that the set of weighted samples approximates the posterior distribution at time* t*, denoted* $p(z^t|x^0, \ldots, x^t)$*. If at time* $t+1$*, a sample* s *is drawn from the posterior*

$$z_s^{t+1} \sim p(z^{t+1}|x^{t+1}, z^t = z_s^t), \tag{A.1}$$

the weights do not need to be updated to accurately represent the new posterior distribution, $p(z^{t+1}|x^0, \ldots, x^{t+1})$*.*

Proof. We show that an expectation of an arbitrary function, $h(\cdot)$, can be approximated by the sum of the weighted samples. Denoting $w(z^t)$ as the associated weight for the random sample, z^t, we have

$$\begin{aligned}
\sum_s w_s^t h(z_s^{t+1}) &\approx \mathbf{E}_q \left[w(z^t) h(z^{t+1}) \right] \\
&= \mathbf{E}_q \left[\mathbf{E}_{z^{t+1}|x^{t+1},z^t} \left[w(z^t) h(z^{t+1}) | z^t \right] \right] \\
&= \int_{z^t} q(z^t) \int_{z^{t+1}} p(z^{t+1}|x^{t+1}, z^t) w(z^t) h(z^{t+1}) dz^{t+1} dz^t.
\end{aligned}$$

Substituting the equation for importance weights (i.e., $w(z^t) = \frac{p(z^t|x^0,\ldots,x^t)}{q(z^t)}$) results in

$$\begin{aligned}\sum_s w_s^t h(z_s^{t+1}) &= \int_{z^t}\int_{z^{t+1}} p(z^t|x^0,\ldots,x^t)p(z^{t+1}|x^{t+1},z^t)h(z^{t+1})dz^{t+1}dz^t \\ &= \int_{z^t}\int_{z^{t+1}} p(z^{t+1},z^t|x^0,\ldots,x^{t+1})h(z^{t+1})dz^{t+1}dz^t \\ &= \int_{z^{t+1}} p(z^{t+1}|x^{1:t+1})h(z^{t+1})dz^{t+1} \\ &= \mathbf{E}_{z^{t+1}|x^{1:t+1}}\left[h(z^{t+1})\right].\end{aligned}$$

Therefore, when w_s^t correctly represent importance weights, propagating particles using both prior and data evidence does not require updating the weights. □

■ A.2 Approximate Marginalization of Independent Flow

Equations (4.21)–(4.24) describe the approximation used to marginalize out the independent flow f. We describe this derivation in more detail here.

$$p(z_m^t|g_m^t, z_{\setminus m}^t, z^{t-1}, a_m^{t-1}, x^t, \mathfrak{Z}^t) \tag{A.2}$$
$$\propto p(x^t, z_m^t|g_m^t, z_{\setminus m}^t, z_m^{t-1}, a_m^{t-1}, \mathfrak{Z}^t) \tag{A.3}$$
$$= p(x^t|z^t, g_m^t, z_m^{t-1}, a_m^{t-1}, \mathfrak{Z}^t)p(z_m^t|g_m^t, z_m^{t-1}, a_m^{t-1}, \mathfrak{Z}^t) \tag{A.4}$$
$$= \int p(f_m^t|g_m^t)p(z_m^t|fz_m^{t-1})p(a_m^t|fa_m^{t-1})p(x^t|z^t, a^t, \mathfrak{Z}^t)df_m^t \tag{A.5}$$
$$= Q_L(z_m^t)\prod_i \int p(f_{m,i}^t|g_{m,i}^t)Q_S(z_{m,i}^t|fz_{m,i}^{t-1})p(a_{m,i}^t|fa_{m,i}^{t-1})p(x_i^t|z_i^t, a_i^t, \mathfrak{Z}^t)df_m^t \tag{A.6}$$
$$= Q_L(z_m^t)\prod_i \int \mathcal{N}(f_{m,i}^t; g_{m,i}^t, \sigma_f^2)Q_S(z_{m,i}^t|fz_{m,i}^{t-1})p(a_{m,i}^t|fa_{m,i}^{t-1})p(x_i^t|z_i^t, a_i^t, \mathfrak{Z}^t)df_m^t \tag{A.7}$$

We note that evolving an image (e.g., a) with a flow (e.g., f) can be expressed as evolving with a different flow (e.g., g) with an additional offset:

$$fa_i = a_{i+f_i} = a_{i+g_i+f_i-g_i} = ga_{i+f_i-g_i}. \tag{A.8}$$

Using this relationship, we can express terms in Equation (A.7) as follows. The symmetric area difference prior and appearance likelihood can be expressed as

$$Q_S(z_{m,i}^t|fz_{m,i}^{t-1}) = Q_S(z_{m,i}^t|gz_{m,i+f_i-g_i}^{t-1}) = Q_S(z_{m,i}^t|gz_{m,i+j}^{t-1}) \tag{A.9}$$
$$p(a_{m,i}^t|fa_{m,i}^{t-1}) = p(a_{m,i}^t|ga_{m,i+f_i-g_i}^{t-1}) \quad = p(a_{m,i}^t|ga_{m,i+j}^{t-1}), \tag{A.10}$$

where we denote $j \triangleq f_i - g_i$. Furthermore, the independent deviation of f from g can be expressed as

$$\mathcal{N}(f_{m,i}^t; g_{m,i}^t, \sigma_f^2) = \mathcal{N}(j, 0, \sigma_f^2). \tag{A.11}$$

Combining Equations (A.7)–(A.11) results in

$$p(z_m^t | g_m^t, z_{\setminus m}^t, z^{t-1}, a_m^{t-1}, x^t, \mathfrak{z}^t) \tag{A.12}$$

$$= Q_L(z_m^t) \prod_i \int \mathcal{N}(j; 0, \sigma_f^2) Q_S(z_{m,i}^t | g z_{m,i+j}^{t-1}) p(a_{m,i}^t | g a_{m,i+j}^{t-1}) p(x_i^t | z_i^t, a_i^t, \mathfrak{z}^t) dj \tag{A.13}$$

Denoting $\mathcal{L}_{m,i}^t(j)$ as

$$\mathcal{L}_{m,i}^t(j) = Q_S(z_{m,i}^t | g_{m,i+j}^{t-1}) p(a_{m,i}^t | g a_{m,i+j}^t) p(x_i^t | z_i^t, a_i^t, \mathfrak{z}^t), \tag{A.14}$$

followed by a discrete approximation to the integral results in

$$p(z_m^t | g_m^t, z_{\setminus m}^t, z^{t-1}, a_m^{t-1}, x^t, \mathfrak{z}^t) \propto Q_L(z_m^t) \prod_i \int \mathcal{N}(j; 0, \sigma_f^2) \mathcal{L}_{m,i}^t(j) dj \tag{A.15}$$

$$\approx Q_L(z_m^t) \prod_i \sum_j \mathcal{N}(j; 0, \sigma_f^2) \mathcal{L}_{m,i}^t(j) \tag{A.16}$$

$$= Q_L(z_m^t) \prod_i \sum_j h_f(j) \mathcal{L}_{m,i}^t(j), \tag{A.17}$$

which is exactly Equation (4.24).

Appendix B

Derivations Pertaining to DPMM Sub-Clusters

In this appendix, we derive expressions related to Chapter 5. We compute the Hastings ratios for the split and merge proposals, and develop the approximation that automatically rejects all proposed deterministic merge moves.

■ B.1 Auxiliary Variable Prior and Posterior Distributions

For the naïve choice for auxiliary parameter distributions of Equations (5.13)–(5.15), reproduced here for convenience

$$\begin{aligned}
p(\overline{\pi}_k) &= \mathrm{Dir}(\overline{\pi}_{k\ell}, \overline{\pi}_{kr}; \tfrac{\alpha}{2}, \tfrac{\alpha}{2}), \\
p(\overline{\theta}_k) &= f_\theta(\overline{\theta}_{k\ell}; \lambda) f_\theta(\overline{\theta}_{kr}; \lambda), \\
p(\overline{z}|\overline{\pi}, \overline{\theta}, x, z) &= \prod_{k=1}^{K} \prod_{i \in \mathcal{I}_k} \sum_{h \in \{\ell, r\}} \frac{\overline{\pi}_{kh} f_x(x_i; \overline{\theta}_{kh})}{Z_i(x, z, \overline{\pi}_k, \overline{\theta}_k)} \mathbb{1}[\overline{z}_i = h], \\
Z_i(x, z, \overline{\pi}_k, \overline{\theta}_k) &\triangleq \overline{\pi}_{k\ell} f_x(x_i; \overline{\theta}_{k\ell}) + \overline{\pi}_{kr} f_x(x_i; \overline{\theta}_{kr}),
\end{aligned}$$

the joint distribution for auxiliary parameters for cluster k can be expressed as

$$\begin{aligned}
&p(\overline{\pi}_k, \overline{\theta}_k, \overline{z}_{\{k\}} | x, \pi, z, \theta) \\
&= \mathrm{Dir}(\overline{\pi}_{k\ell}, \overline{\pi}_{kr}; \tfrac{\alpha}{2}, \tfrac{\alpha}{2}) f_\theta(\overline{\theta}_{k\ell}; \lambda) f_\theta(\overline{\theta}_{kr}; \lambda) \prod_{i \in \mathcal{I}_k} \frac{\overline{\pi}_{k\overline{z}_i} f_x(x_i; \overline{\theta}_{k\overline{z}_i})}{Z_i(x, z, \overline{\pi}_k, \overline{\theta}_k)} && \text{(B.1)} \\
&= \mathrm{Dir}(\overline{\pi}_{k\ell}, \overline{\pi}_{kr}; \tfrac{\alpha}{2}, \tfrac{\alpha}{2}) \prod_{h=\{\ell, r\}} \overline{\pi}_{kh}^{N_{kh}} f_x(x_{\mathcal{I}_{kh}}; \overline{\theta}_{kh}) f_\theta(\overline{\theta}_{kh}; \lambda) \prod_{i \in \mathcal{I}_k} \tfrac{1}{Z_i(x, z, \overline{\pi}_k, \overline{\theta}_k)}. && \text{(B.2)}
\end{aligned}$$

By simply ignoring terms that do not correspond to the variable of interest, the posterior distributions for the the sub-cluster weights and parameters can be expressed as

$$p(\overline{\pi}_k|\bullet) \propto \text{Dir}(\overline{\pi}_{k\ell}, \overline{\pi}_{kr}; \tfrac{\alpha}{2}, \tfrac{\alpha}{2})\, \overline{\pi}_{k\ell}^{N_{k\ell}}\, \overline{\pi}_{kr}^{N_{kr}} \prod_{i\in\mathcal{I}_k} \tfrac{1}{Z_i(x,z,\overline{\pi}_k,\overline{\theta}_k)} \tag{B.3}$$

$$= \text{Dir}(\overline{\pi}_{k\ell}, \overline{\pi}_{kr}; \tfrac{\alpha}{2} + N_{k\ell}, \tfrac{\alpha}{2} + N_{kr}) \prod_{i\in\mathcal{I}_k} \tfrac{1}{Z_i(x,z,\overline{\pi}_k,\overline{\theta}_k)}, \tag{B.4}$$

$$p(\overline{\theta}_k|\bullet) \propto f_\theta(\overline{\theta}_{k\ell}; \lambda)\, f_\theta(\overline{\theta}_{kr}; \lambda)\, f_x(x_{\mathcal{I}_{k\ell}}; \theta_{k\ell})\, f_x(x_{\mathcal{I}_{kr}}; \theta_{kr}) \prod_{i\in\mathcal{I}_k} \tfrac{1}{Z_i(x,z,\overline{\pi}_k,\overline{\theta}_k)} \tag{B.5}$$

$$\propto f_\theta(\overline{\theta}_{k\ell}; \lambda^*_{k\ell})\, f_\theta(\overline{\theta}_{kr}; \lambda^*_{kr}) \prod_{i\in\mathcal{I}_k} \tfrac{1}{Z_i(x,z,\overline{\pi}_k,\overline{\theta}_k)}, \tag{B.6}$$

where $\bullet$ is used to denote all other variables, and we have assumed conjugate priors for explanatory purposes. The product term, $\prod_{i\in\mathcal{I}_k} \frac{1}{Z_i(x,z,\overline{\pi}_k,\overline{\theta}_k)}$, complicates these distributions because they no longer follow the form of regular-cluster parameters.

If the sub-cluster parameters follow the distribution of Equation (5.19), reproduced here,

$$p(\overline{\theta}_k|x,z,\pi) \propto f_\theta(\overline{\theta}_{k\ell}; \lambda) f_\theta(\overline{\theta}_{kr}; \lambda) \prod_{i\in\mathcal{I}_k} Z_i(x,z,\overline{\pi}_k,\overline{\theta}_k) \tag{B.7}$$

the joint distribution can be expressed as

$$p(\overline{\pi}_k, \overline{\theta}_k, \overline{z}_{\{k\}}|x,\pi,z,\theta)$$

$$= \text{Dir}(\overline{\pi}_{k\ell}, \overline{\pi}_{kr}; \tfrac{\alpha}{2}, \tfrac{\alpha}{2}) f_\theta(\overline{\theta}_{k\ell}; \lambda) f_\theta(\overline{\theta}_{kr}; \lambda) \textstyle\prod_{i\in\mathcal{I}_k} \overline{\pi}_{k\overline{z}_i} f_x(x_i; \overline{\theta}_{k\overline{z}_i}) \tag{B.8}$$

$$= \text{Dir}(\overline{\pi}_{k\ell}, \overline{\pi}_{kr}; \tfrac{\alpha}{2}, \tfrac{\alpha}{2}) \textstyle\prod_{h=\{\ell,r\}} \overline{\pi}_{kh}^{N_{kh}} f_x(x_{\mathcal{I}_{kh}}; \overline{\theta}_{kh}) f_\theta(\overline{\theta}_{kh}; \lambda). \tag{B.9}$$

Following a similar argument as before, this results in the desired sub-cluster posterior distributions expressed in Equations (5.20)–(5.22).

■ B.2 Hastings Ratios for Splits

In this section, we derive the Hastings ratio for a deterministic split proposal. We first note the following useful distribution

$$p(z)p(\pi|z) = \frac{\alpha^K \Gamma(\alpha) \prod_k \Gamma(N_k)}{\Gamma(\alpha+N)} \frac{\Gamma(\alpha+N)}{\Gamma(\alpha)\prod_k \Gamma(N_k)} \pi_{K+1}^{\alpha} \prod_k \pi_k^{N_k-1} = \alpha^K \pi_{K+1}^{\alpha} \prod_{k=1}^{K} \pi_k^{N_k-1}. \tag{B.10}$$

The Hastings ratio for a split can be expressed as

$$H^{\text{det}}_{\text{split-}\natural} = \frac{p(\hat{\pi},\hat{z},\hat{\theta},x)p(\hat{\overline{\pi}},\hat{\overline{\theta}},\hat{\overline{z}}|x,\hat{z})}{p(\pi,z,\theta,x)p(\overline{\pi},\overline{\theta},\overline{z}|x,z)} \cdot \frac{Q^{K+1}_{\text{merge-}\flat\sharp}}{Q^{K}_{\text{split-}\natural}} \cdot \frac{q(\pi,z,\theta,\overline{\pi},\overline{\theta},\overline{z}|\hat{\pi},\hat{z},\hat{\theta},\hat{\overline{\pi}},\hat{\overline{\theta}},\hat{\overline{z}},Q^{K+1}_{\text{merge-}\flat\sharp})}{q(\hat{\pi},\hat{z},\hat{\theta},\hat{\overline{\pi}},\hat{\overline{\theta}},\hat{\overline{z}}|\pi,z,\theta,\overline{\pi},\overline{\theta},\overline{z},Q^{K}_{\text{split-}\natural})}. \tag{B.11}$$

Because of the deferred proposal for auxiliary variables, this can be simplified to

$$H^{\text{det}}_{\text{split-}\natural} = \frac{p(\hat{\pi},\hat{z},\hat{\theta},x)}{p(\pi,z,\theta,x)} \cdot \frac{Q^{K+1}_{\text{merge-}\flat\sharp}}{Q^{K}_{\text{split-}\natural}} \cdot \frac{q(\pi,z,\theta|\hat{\pi},\hat{z},\hat{\theta},\hat{\overline{\pi}},\hat{\overline{\theta}},\hat{\overline{z}},Q^{K+1}_{\text{merge-}\flat\sharp})}{q(\hat{\pi},\hat{z},\hat{\theta}|\pi,z,\theta,\overline{\pi},\overline{\theta},\overline{z},Q^{K}_{\text{split-}\natural})}. \tag{B.12}$$

We now analyze these terms separately.

The ratio of posteriors for z and π can be easily simplified to

$$\frac{p(\hat{z})p(\hat{\pi}|\hat{z})}{p(z)p(\pi|z)} = \frac{\alpha\hat{\pi}_{\flat}^{\hat{N}_{\flat}-1}\hat{\pi}_{\sharp}^{\hat{N}_{\sharp}-1}}{\pi_{\natural}^{N_{\natural}-1}}. \tag{B.13}$$

This results in the following posterior ratio

$$\frac{p(\hat{\pi},\hat{z},\hat{\theta},x)}{p(\pi,z,\theta,x)} = \boxed{\frac{\alpha\hat{\pi}_{\flat}^{\hat{N}_{\flat}-1}\hat{\pi}_{\sharp}^{\hat{N}_{\sharp}-1}}{\pi_{\natural}^{N_{\natural}-1}} \cdot \frac{f_\theta(\hat{\theta}_{\flat};\lambda)f_x(x_{\mathcal{I}_{\flat}};\hat{\theta}_{\flat})\, f_\theta(\hat{\theta}_{\sharp};\lambda)f_x(x_{\mathcal{I}_{\sharp}};\hat{\theta}_{\sharp})}{f_\theta(\theta_{\natural};\lambda)f_x(x_{\mathcal{I}_{\natural}};\theta_{\natural})}}. \tag{B.14}$$

The ratio of proposal distributions can similarly be simplified. We first note that the proposal for the new labels is an indicator function at the particular split or merge move:

$$q(\hat{z}|z,\overline{z},Q_{\text{split-}\natural}) = \mathbb{I}[\hat{z} = \text{split-}\natural(z,\overline{z})], \tag{B.15}$$

$$q(z|\hat{z},\hat{\overline{z}},Q_{\text{merge-}\flat\sharp}) = \mathbb{I}[z = \text{merge-}\flat\sharp(\hat{z})]. \tag{B.16}$$

We note one important observation; the reverse label move that merges the two proposed clusters does not depend on auxiliary variables. This results in all split moves being exactly reversible by a merge move. The proposal ratio for proposed labels can then simplify to

$$\boxed{\frac{q(z|\hat{z},\hat{\overline{z}},Q_{\text{merge-}\flat\sharp})}{q(\hat{z}|z,\overline{z},Q_{\text{split-}\natural})} = 1} \tag{B.17}$$

Conditioned on these new labels, we use the Reversible-Jump MCMC (RJMCMC) [44] algorithm to calculate the term for the π's. RJMCMC is a generalization of Metropolis-Hastings where auxiliary variables may be used to propose deterministic moves in mismatched dimensions. A detailed description of the RJMCMC algorithm can be found in Section 2.5.1.

The split proposal can be expressed as a mapping from the original space $[\pi_\natural, v]$ to the new space, $[\hat{\pi}_\flat, \hat{\pi}_\sharp]$,

$$[\pi_\natural, v] \rightarrow [\hat{\pi}_\flat, \hat{\pi}_\sharp], \tag{B.18}$$

where $v \sim \text{Beta}(\hat{N}_\flat, \hat{N}_\sharp)$ and the deterministic mapping between dimensions is

$$\hat{\pi}_\flat = \pi_\natural v, \quad \hat{\pi}_\sharp = \pi_\natural(1-v). \tag{B.19}$$

Because of the relationship between the Beta distribution and the Dirichlet distribution, the above construction can also be seen as a draw from a Dirichlet distribution followed by scaling by $\pi_\natural$. The Jacobian matrix can be expressed as

$$J_\pi = \begin{bmatrix} \frac{\partial \hat{\pi}_\flat}{\partial \pi_\natural} & \frac{\partial \hat{\pi}_\flat}{\partial v} \\ \frac{\partial \hat{\pi}_\sharp}{\partial \pi_\natural} & \frac{\partial \hat{\pi}_\sharp}{\partial v} \end{bmatrix} = \begin{bmatrix} v & \pi_\natural \\ (1-v) & -\pi_\natural \end{bmatrix}, \tag{B.20}$$

which has a corresponding absolute value determinant

$$|\det(J_\pi)| = |-\pi_\natural v - (\pi_\natural(1-v))| = \pi_\natural. \tag{B.21}$$

Therefore, the RJ proposal ratio for the $\pi's$ can be expressed as

$$\frac{q(\pi|\hat{\bullet})}{q(\hat{\pi}|\bullet)} = \frac{1}{\text{Beta}(v; \hat{N}_\flat, \hat{N}_\sharp)}\pi_\natural = \frac{\Gamma(\hat{N}_\flat)\Gamma(\hat{N}_\sharp)}{\Gamma(N_\natural)} v^{\hat{N}_\flat - 1}(1-v)^{\hat{N}_\sharp - 1}\pi_\natural \tag{B.22}$$

$$= \frac{\Gamma(\hat{N}_\flat)\Gamma(\hat{N}_\sharp)}{\Gamma(N_\natural)} \left(\frac{\hat{\pi}_\flat}{\pi_\natural}\right)^{\hat{N}_\flat - 1} \left(\frac{\hat{\pi}_\sharp}{\pi_\natural}\right)^{\hat{N}_\sharp - 1} \pi_\natural \tag{B.23}$$

$$= \boxed{\frac{\Gamma(\hat{N}_\flat)\Gamma(\hat{N}_\sharp)}{\Gamma(N_\natural)} \hat{\pi}_\flat^{\hat{N}_\flat - 1} \hat{\pi}_\sharp^{\hat{N}_\sharp - 1} \pi_\natural^{-N_\natural + 1}}. \tag{B.24}$$

We note that we slightly abuse the notation above, and by the ratio, $\frac{q(\pi|\hat{\bullet})}{q(\hat{\pi}|\bullet)}$, we actually mean the true RJMCMC ratio, $\frac{1}{q(v|\bullet)}|\det J_\pi|$.

A similar reversible-jump proposal ratio can be found for the θ's. This requires one to define a 6-dimensional augmented space which we do not express here. Because the Jacobian can be written as the identity matrix, the determinant is simply 1. Thus, the ratio can be expressed as

$$\boxed{\frac{q(\theta|\hat{\bullet})}{q(\hat{\theta}|\bullet)} = \frac{q(\theta_\natural|x, z, \overline{z})}{q(\hat{\theta}_\flat|x, \hat{z}, \hat{\overline{z}})q(\hat{\theta}_\sharp|x, \hat{z}, \hat{\overline{z}})}}. \tag{B.25}$$

Combining the boxed expressions, results in the split Hastings ratio of Equation (5.30).

■ B.3 Hastings Ratios for Merges

The Hastings ratio for a deterministic merge move is more complicated. This is due to the following proposal ratio for labels

$$\frac{q(z|\hat{z},\hat{\overline{z}},Q^{K-1}_{\text{split-}\natural})}{q(\hat{z}|z,\overline{z},Q^{K}_{\text{merge-}\flat\sharp})}. \tag{B.26}$$

The numerator calculates the probability of splitting the proposed merged cluster back into the original two clusters. This means that the sub-cluster labels, $\hat{\overline{z}}$, must exactly correspond to the current clusters for the ratio to be non-zero. In practice, this is very unlikely for most situations. Consider the case where cluster $\flat$ and cluster $\sharp$ have essentially the same weights and parameters (e.g., both Gaussian with the same mean and covariance). In the limit, this results in $\Pr(\hat{\overline{z}}_{\mathcal{I}_\flat} = \ell, \hat{\overline{z}}_{\mathcal{I}_\sharp} = r) = \left(\frac{1}{2}\right)^{N_\flat+N_\sharp}$ since all configurations are the same. This means with only probability $\left(\frac{1}{2}\right)^{N_\flat+N_\sharp}$ do the proposed sub-cluster labels exactly correspond to the original regular-clusters. This probability clearly diminishes very quickly as N grows. When the proposed sub-cluster label does not correspond to the regular-clusters, the proposal is automatically rejected since $q(z|\hat{z},\hat{\overline{z}},Q^{K-1}_{\text{split-}\natural}) = 0$.

As the clusters $\flat$ and $\sharp$ become more separable, the probability of proposing merged sub-cluster labels that exactly correspond to the regular-cluster labels increases. Imagine the limiting case where clusters $\flat$ and $\sharp$ are infinitely far apart so that the only non-zero label assignment is the one that splits the data points correctly. In this case, the probability of the labels is one, and the probability of proposing the corresponding merged sub-cluster labels also is one. This results in a proposal ratio for labels equaling one. However, under the same assumption that the clusters $\flat$ and $\sharp$ are very separable, it should be intuitive that they should *not* be merged to begin with. As such, the posterior ratio in the Hastings ratio will begin to dominate the overall acceptance ratio, and consequently still approach zero.

More precisely, the probability of accepting a proposed merge can be expressed as

$$\begin{aligned}
&\min[1, H^{\text{det}}_{\text{merge-}\flat\sharp}] \\
&= \min\left[1, \frac{\Gamma(\hat{N}_\natural) f_x(x_{\mathcal{I}_\natural};\lambda)}{\alpha \prod_{k\in\{\flat,\sharp\}} \Gamma(N_k) f_x(x_{\mathcal{I}_k};\lambda)} \; \mathbb{1}\big[(\hat{\overline{z}}_{\mathcal{I}_{\natural\ell}}, \hat{\overline{z}}_{\mathcal{I}_{\natural r}}) = (z_{\mathcal{I}_\flat}, z_{\mathcal{I}_\sharp})\big]\right] &\text{(B.27)}\\
&= \min\left[1, \frac{\Gamma(\hat{N}_\natural) f_x(x_{\mathcal{I}_\natural};\lambda)}{\alpha \prod_{k\in\{\flat,\sharp\}} \Gamma(N_k) f_x(x_{\mathcal{I}_k};\lambda)}\right] \; \mathbb{1}\big[(\hat{\overline{z}}_{\mathcal{I}_{\natural\ell}}, \hat{\overline{z}}_{\mathcal{I}_{\natural r}}) = (z_{\mathcal{I}_\flat}, z_{\mathcal{I}_\sharp})\big]. &\text{(B.28)}
\end{aligned}$$

Again, we have assumed conjugate priors for simplifying the explanation.

We note the following inequality

$$f_x(x_{\mathcal{I}_\natural};\lambda) = \int f_x(x_{\mathcal{I}_\natural};\theta_\natural) f_\theta(\theta_\natural;\lambda) d\theta_\natural \leq \int f_x(x_{\mathcal{I}_\natural};\theta^*_\natural) f_\theta(\theta_\natural;\lambda) d\theta_\natural = f_x(x_{\mathcal{I}_\natural};\theta^*_\natural), \tag{B.29}$$

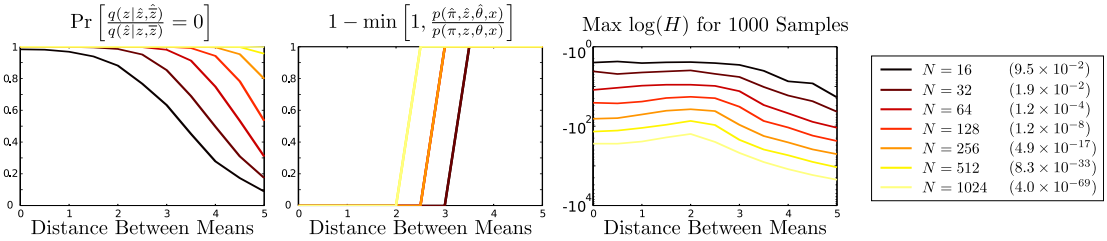


Figure B.1: Probability quantities associated with rejecting a merge proposal. The numbers in the parenthesis correspond to the maximum observed upper bounds to the acceptance ratio over all samples and separations.

where $\theta^*_\natural$ is the mode of the distribution and can be expressed as, $\theta^*_\natural = \max_{\theta_\natural} f_x(x_{\mathcal{I}_\natural};\theta_\natural)$. For any realization of data, the probability of accepting a proposed merge can then be upper bounded by the following

$$\begin{aligned}
&\min[1, H_{\text{merge-}\flat\sharp}] \\
&= \min\left[1, \frac{\Gamma(\hat{N}_\natural) f_x(x_{\mathcal{I}_\natural};\lambda)}{\alpha\prod_{k\in\{\flat,\sharp\}}\Gamma(N_k) f_x(x_{\mathcal{I}_k};\lambda)}\right] \Pr\left[(\hat{\overline{z}}_{\mathcal{I}_{\natural\ell}}, \hat{\overline{z}}_{\mathcal{I}_{\natural r}}) = (z_{\mathcal{I}_\flat}, z_{\mathcal{I}_\sharp})|x,z\right] \qquad \text{(B.30)} \\
&\leq \min\left[1, \frac{\Gamma(\hat{N}_\natural) f_x(x_{\mathcal{I}_\natural};\lambda)}{\alpha\prod_{k\in\{\flat,\sharp\}}\Gamma(N_k) f_x(x_{\mathcal{I}_k};\lambda)}\right] \Pr\left[(\hat{\overline{z}}_{\mathcal{I}_{\natural\ell}}, \hat{\overline{z}}_{\mathcal{I}_{\natural r}}) = (z_{\mathcal{I}_\flat}, z_{\mathcal{I}_\sharp})|x,z,\overline{\theta}^*_l\right]. \qquad \text{(B.31)}
\end{aligned}$$

We test our approximation to compute the probability that this acceptance ratio is 0 on the following synthetic data. We generate data from two 1D Gaussian distributions with mean separated by a varying amount. A Gaussian prior is used on the mean, and the variance is assumed to be known. The probability that the proposed merge is automatically rejected due to the second term in Equation (B.31) for varying separations and number of data points is shown in the first panel of Figure B.1. Each value on each curve was calculated with 1,000 samples of N data points. Clearly, as N increases, the proposed merge is rejected automatically more and more frequently. As expected, as the separation between the two clusters increases, the automatic rejection is less likely.

Next, we show the probability of rejecting the merge due to the first term in Equation (B.31). The curves are shown in the second panel of Figure B.1. This plot shows that as the separation increases, the posterior ratio favors rejecting the sample more and more. The overall log acceptance ratio is shown in the last panel of Figure B.1 (note the additional log scale). In the legend, the values in the parenthesis indicate the maximum observed upper bounds to the acceptance ratio over all samples and separations. Even for relatively small N, approximating the proposals for merge moves to always be rejected is very good. We note that the samplers are typically run for less than 10^3 iterations even for large datasets of $N \approx 10^6$, reinforcing the validity of the approximation even more.

Appendix C

Derivations Pertaining to HDP Sub-Topics

In this appendix, we derive expressions related to Chapter 6. We begin by deriving the joint prior distribution, $p(\beta, z)$. We then show that the typical set in HDP models is very far from the mode of the distribution, which is when all the words are placed into a single, all-encompassing topic. Lastly, the Hastings ratios for the local and global split/merge proposals are computed.

C.1 Calculating the $p(\beta, z)$ Distribution

To calculate the prior distribution, $p(\beta, z)$, we will rely heavily on the Chinese Restaurant Franchise (CRF) representation of the HDP given in [116], which we summarize in Section 2.9.3. In particular, we will make use of τ_{ji}, the table assignment for customer i in restaurant j, and the dish assignment κ_{jt} assigned to table t in restaurant j. The derivation is outlined in the following steps.

1. Find $p(\beta, \kappa, \tau, z)$
2. Find $p(\kappa, \tau|\beta, z)$
3. Combine to find $p(\beta, z)$

C.1.1 Deriving the Joint: $p(\beta, \kappa, \tau, z)$

Based on the generative process of the CRF, $p(\kappa, \tau)$ can be expressed as

$$p(\tau) = \prod_{j=1}^{D} \mathrm{CRP}(\alpha, n_{j\cdot\cdot}) = \prod_{j=1}^{D} \frac{\Gamma(\alpha)\alpha^{m_{j\cdot}}}{\Gamma(\alpha + n_{j\cdot\cdot})} \prod_{t=1}^{m_{j\cdot}} \Gamma(n_{jt\cdot}), \tag{C.1}$$

$$p(\kappa|\tau) = \mathrm{CRP}(\gamma, m_{\cdot\cdot}) = \frac{\Gamma(\gamma)\gamma^{K}}{\Gamma(\gamma + m_{\cdot\cdot})} \prod_{k=1}^{K} \Gamma(m_{\cdot k}), \tag{C.2}$$

where CRP$(\cdot)$ represents a sample from a Chinese Restaurant Process. These expressions can be combined to form the joint:

$$p(\kappa, \tau) = \left[\frac{\Gamma(\gamma)\gamma^K}{\Gamma(\gamma + m_{\cdot\cdot})} \prod_{k=1}^{K} \Gamma(m_{\cdot k}) \right] \cdot \left[\prod_{j=1}^{D} \frac{\Gamma(\alpha)\alpha^{m_{j\cdot}}}{\Gamma(\alpha + n_{j\cdot\cdot})} \prod_{t=1}^{m_{j\cdot}} \Gamma(n_{jt\cdot}) \right]. \tag{C.3}$$

We note that z, which directly assigns a customer to a dish, is a deterministic function conditioned on κ and τ. More precisely, it can be expressed as

$$p(z|\kappa, \tau) = \prod_{j=1}^{D} \prod_{i=1}^{N_j} \mathbb{I}[z_{ji} = \kappa_{j\tau_{ji}}]. \tag{C.4}$$

Additionally, it is well known that $p(\beta|m)$ is the following Dirichlet distribution

$$p(\beta|m) = \text{Dir}(\beta_1, \ldots, \beta_K, \beta_{K+1}; m_{\cdot 1}, \cdots, m_{\cdot K}, \gamma). \tag{C.5}$$

This can be seen by drawing on the fact that any partitioning of the space in a Dirichlet process results in a Dirichlet distribution. Since m is a summary statistic of k, we have

$$p(\beta|\kappa, \tau, z) = p(\beta|m) = \frac{\Gamma(\gamma + m_{\cdot\cdot})\beta_{K+1}^{\gamma-1}}{\Gamma(\gamma)\prod_{k=1}^{K}\Gamma(m_{\cdot k})} \prod_{k=1}^{K} \beta_k^{m_{\cdot k}-1} \tag{C.6}$$

Finally, assuming z is consistent with κ and τ (i.e., Equation (C.4) evaluates to 1 instead of 0), the entire joint prior of interest can be expressed as

$$\begin{aligned} &p(\beta, \kappa, \tau, z) \\ &= \left[\tfrac{\Gamma(\gamma)\gamma^K}{\Gamma(\gamma+m_{\cdot\cdot})} \prod_{k=1}^{K} \Gamma(m_{\cdot k}) \right] \left[\prod_{j=1}^{D} \tfrac{\Gamma(\alpha)\alpha^{m_{j\cdot}}}{\Gamma(\alpha+n_{j\cdot\cdot})} \prod_{t=1}^{m_{j\cdot}} \Gamma(n_{jt\cdot}) \right] \left[\tfrac{\Gamma(\gamma+m_{\cdot\cdot})\beta_{K+1}^{\gamma-1}}{\Gamma(\gamma)\prod_{k=1}^{K}\Gamma(m_{\cdot k})} \prod_{k=1}^{K} \beta_k^{m_{\cdot k}-1} \right] \\ &= \gamma^K \beta_{K+1}^{\gamma-1} \alpha^{m_{\cdot\cdot}} \left[\prod_{k=1}^{K} \beta_k^{m_{\cdot k}-1} \right] \left[\prod_{j=1}^{D} \frac{\Gamma(\alpha)}{\Gamma(\alpha + n_{j\cdot\cdot})} \prod_{t=1}^{m_{j\cdot}} \Gamma(n_{jt\cdot}) \right] \end{aligned} \tag{C.7}$$

■ C.1.2 Deriving the Conditional $p(\kappa, \tau|\beta, z)$

We now show how to express $p(\kappa, \tau|\beta, z)$. We note that conditioning on z is equivalent to assigning each customer a particular dish. Thus, we need to calculate the probability of any particular configuration of tables such that each customer gets the correct dish.

Suppose there are three customers with assignments $z_{11} = 1$, $z_{12} = 2$, and $z_{13} = 2$. While x_{11} must sit at a different table than x_{12} and x_{13} (i.e., $\tau_{12} \neq \tau_{11} \neq \tau_{23}$), nothing can be said about the relationship between τ_{12} and τ_{13}. This results from the fact that two customers can be served the same dish at different tables.

An equivalent metaphor for the process conditioned on z is that a customer comes

into a restaurant having been assigned dish k. The customer then chooses to sit at an occupied table serving dish k with probability proportional to the number of customers there, or starts a new table that serves dish k with probability $\alpha\beta_k$. This process is equivalent to $D \times K$ independent CRPs, each with $n_{j\cdot k}$ customers and $\alpha\beta_k$ as the concentration parameter. Thus, we can write this easily as

$$p(\kappa, \tau|\beta, z) = \prod_{j=1}^{D}\prod_{k=1}^{K} \mathrm{CRP}(\alpha\beta_k, n_{j\cdot k}) \tag{C.8}$$

$$= \prod_{j=1}^{D}\prod_{k=1}^{K} \frac{(\alpha\beta_k)^{m_{jk}}\Gamma(\alpha\beta_k)}{\Gamma(\alpha\beta_k + n_{j\cdot k})} \prod_{t=1}^{m_{jk}} \Gamma(n_{jtk}) \tag{C.9}$$

$$= \alpha^{m_{\cdot\cdot}} \left[\prod_{j=1}^{D}\prod_{k=1}^{K} \frac{\beta_k^{m_{jk}}\Gamma(\alpha\beta_k)}{\Gamma(\alpha\beta_k + n_{j\cdot k})}\right] \left[\prod_{j=1}^{D}\prod_{t=1}^{m_{jk}} \Gamma(n_{jt\cdot})\right], \tag{C.10}$$

where we have used the fact that every table only serves one dish to equate the following

$$\prod_{k=1}^{K}\prod_{t=1}^{m_{jk}} \Gamma(n_{jtk}) = \prod_{t=1}^{m_{j\cdot}} \Gamma(n_{jt\cdot}). \tag{C.11}$$

■ C.1.3 Finding the Prior $p(\beta, z)$

We now note the following relationship:

$$p(\beta, \kappa, \tau, z) = p(\beta, z)p(\kappa, \tau|\beta, z). \tag{C.12}$$

Finding the expression for $p(\beta, z)$ is as simple as substituting the previously found expressions. Assuming consistency between z with κ and τ, we can ignore the $p(z|\kappa, \tau)$ term in Equation (C.4), resulting in

$$p(\beta, z) = \frac{p(\beta, \kappa, \tau, z)}{p(\kappa, \tau|\beta, z)}, \tag{C.13}$$

$$= \frac{\gamma^K \beta_{K+1}^{\gamma-1} \alpha^{m_{\cdot\cdot}} \left[\prod_{k=1}^{K} \beta_k^{m_{\cdot k}-1}\right] \left[\prod_{j=1}^{D} \frac{\Gamma(\alpha)}{\Gamma(\alpha+n_{j\cdot\cdot})} \prod_{t=1}^{m_{j\cdot}} \Gamma(n_{jt\cdot})\right]}{\alpha^{m_{\cdot\cdot}} \left[\prod_{j=1}^{D}\prod_{k=1}^{K} \frac{\Gamma(\alpha\beta_k)}{\Gamma(\alpha\beta_k+n_{j\cdot k})} \beta_k^{m_{jk}}\right] \left[\prod_{j=1}^{D}\prod_{t=1}^{m_{j\cdot}} \Gamma(n_{jt\cdot})\right]}, \tag{C.14}$$

$$= \frac{\gamma^K \beta_{K+1}^{\gamma-1} \left[\prod_{k=1}^{K} \beta_k^{-1}\right] \left[\prod_{j=1}^{D} \frac{\Gamma(\alpha)}{\Gamma(\alpha+n_{j\cdot\cdot})}\right]}{\prod_{j=1}^{D}\prod_{k=1}^{K} \frac{\Gamma(\alpha\beta_k)}{\Gamma(\alpha\beta_k+n_{j\cdot k})}}, \tag{C.15}$$

$$= \gamma^K \beta_{K+1}^{\gamma-1} \prod_{k=1}^{K} \beta_k^{-1} \left[\prod_{j=1}^{D} \frac{\Gamma(\alpha)}{\Gamma(\alpha + n_{j\cdot\cdot})} \prod_{k=1}^{K} \frac{\Gamma(\alpha\beta_k + n_{j\cdot k})}{\Gamma(\alpha\beta_k)}\right]. \tag{C.16}$$

This concludes the derivation of finding the expression for $p(\beta, z)$.

C.1.4 Notes on $p(\beta, z)$

We highlight a few notes on the derived expression for $p(\beta, z)$. At first glance, parts of Equation (C.16) may seem a bit odd. For example, the β_k^{-1} term seems like an improper prior, and the term inside the square brackets just seems like the product of Dirichlet-Categorical distributions. We remind the reader that meaning of β in $p(\beta, z)$ slightly differs from the infinite-length global topic proportions. In particular, β is defined over the partitions imposed by z. Moreover, because z takes on exactly K non-empty partitions, $\beta_k > 0 \; \forall k \in \{1, \ldots, K\}$, and β_k^{-1} will never result in division by zero.

The term in the square brackets is very similar to a Dirichlet-Categorical (defined in Section 2.3.3. A product of D independent Dirichlet-Categorical distributions can be expressed as

$$\prod_{j=1}^{D} \text{DirCat}(z_j; \alpha\beta_1, \ldots, \alpha\beta_{K+1}) = \prod_{j=1}^{D} \frac{\Gamma(\alpha)}{\Gamma(\alpha + n_{j\cdot\cdot})} \prod_{k=1}^{K+1} \frac{\Gamma(\alpha\beta_k + n_{j\cdot k})}{\Gamma(\alpha\beta_k)},$$

where we have used the fact that $\sum_{k=1}^{K} \alpha\beta_k = \alpha$. While this expression looks similar to the one in Equation (C.16), the inner product is over $K + 1$ terms instead of K. Furthermore, Equation (C.16) implicitly assumes that $n_{j\cdot(K+1)}$ is zero, since z can only take on K unique partitions. For these reasons, the term inside the square brackets of Equation (C.16) is not the product of Dirichlet-Categoricals.

Equation (C.16) cannot be analytically integrated to validate that it has the correct normalization. We know, however, that Equation (C.10) describing $p(\kappa, \tau|\beta, z)$ trivially integrates to one by the construction of the independent CRPs. Because $p(\kappa, \tau|\beta, z)$ is a valid distribution and $p(\beta, \kappa, \tau, z)/p(\kappa, \tau|\beta, z)$ has no dependence on κ or τ, the derived expression for $p(\beta, z)$ must be a valid distribution, conditioned on $p(\beta, \kappa, \tau, z)$ being the correct joint distribution.

C.2 Joint Model Likelihoods

When the desired topic distributions have a lot of overlap, the resulting likelihood of a sample from the typical set under the posterior is typically much smaller than the mode of the distribution. To illustrate this observation, we consider the Associated Press dataset of [9]. We initialize a sample with K initial topics, and then run a sampling algorithm while restricting the addition of new topics. The resulting joint posterior log likelihood for the entire model is shown in Figure C.1 Clearly, the joint likelihood is highest for one global topic. However, we do not expect a sample from the posterior to only contain one topic. This observation means that the configuration with a single topic is not in the typical set of the posterior, even though it has the highest likelihood.

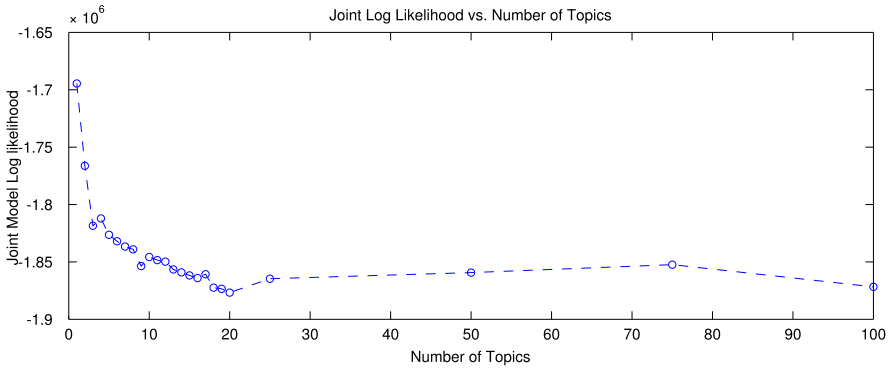


Figure C.1: Joint posterior model log likelihood plotted against number of topics for the Associated Press dataset [9].

For this reason, a deterministic split proposal that has $\frac{q(z|\hat{v},\hat{\overline{v}})}{q(\hat{z}|v,\overline{v})}$ close to unity will almost always reject the sample, since the model likelihood decreases. This is slightly abnormal since, in general, non-deterministic proposals for a split will have ratios, $\frac{q(z|\hat{v},\hat{\overline{v}})}{q(\hat{z}|v,\overline{v})}$, that evaluate to much less than unity. This results from the fact that there is only one way to merge two clusters, but typically many ways to split a cluster into two. As such, the deterministic split proposals described Chapter 5 do not work well in HDPs.

■ C.3 Hastings Ratios for Local Proposals

For the proposed local split of topic $\natural$, the only variables that are changed are $\beta_\natural$ and z_{ji} for all points with label $\natural$. Thus, the ratio of posteriors for the local split can be expressed as

$$\frac{p(\hat{\beta},\hat{z},x)}{p(\beta,z,x)} = \frac{\gamma\beta_\natural}{\hat{\beta}_\flat\hat{\beta}_\sharp}\left[\prod_{j=1}^{D}\frac{\Gamma(\alpha\beta_\natural)}{\Gamma(\alpha\beta_\natural+n_{j\cdot\natural})}\prod_{k\in\{\flat,\sharp\}}\frac{\Gamma(\alpha\hat{\beta}_k+\hat{n}_{j\cdot k})}{\Gamma(\alpha\hat{\beta}_k)}\right]\frac{p(x|\hat{z})}{p(x|z)} \tag{C.17}$$

Consider the proposal over the main variables $\hat{\beta}$ and $\hat{z}$. A split move is proposed as follows. Conditioned on β and z, we propose a new $\hat{\beta}$ and $\hat{z}$ with the following:

$$\hat{z} \sim q(\hat{z}|v,\overline{v}) \tag{C.18}$$

$$(\tilde{\beta}_\flat,\tilde{\beta}_\sharp) \sim \mathrm{Dir}(\hat{\tilde{m}}_{\cdot\flat},\hat{\tilde{m}}_{\cdot\sharp}) \tag{C.19}$$

$$(\hat{\beta}_\flat,\hat{\beta}_\sharp) = \beta_\natural\cdot(\tilde{\beta}_\flat,\tilde{\beta}_\sharp), \tag{C.20}$$

where $q(\hat{z}|v,\overline{v})$ is described in the main chapter, and we have used $\hat{\tilde{m}} \triangleq \tilde{m}(\hat{z})$. Here, we considering calculating the proposal ratio for the β's. We use the reversible jump

algorithm [44] to calculate the ratio. The function f maps us from

$$[\beta_{\natural}, \tilde{\beta}_{\natural\ell}] \rightarrow [\hat{\beta}_{\flat}, \hat{\beta}_{\sharp}] \tag{C.21}$$

The Jacobian matrix for the mapping is then

$$J_{\beta} = \begin{bmatrix} \frac{\partial \hat{\beta}_{\flat}}{\partial \beta_{\natural}} & \frac{\partial \hat{\beta}_{\flat}}{\partial \tilde{\beta}_{\natural\ell}} \\ \frac{\partial \hat{\beta}_{\sharp}}{\partial \beta_{\natural}} & \frac{\partial \hat{\beta}_{\sharp}}{\partial \tilde{\beta}_{\natural\ell}} \end{bmatrix} = \begin{bmatrix} \tilde{\beta}_{\natural\ell} & \beta_{\natural} \\ (1 - \tilde{\beta}_{\natural\ell}) & -\beta_{\natural} \end{bmatrix}, \tag{C.22}$$

which has an absolute value determinant of

$$|\det(J_{\beta})| = \left| \beta_{\natural} \cdot \tilde{\beta}_{\natural\ell} - \beta_{\natural}(1 - \tilde{\beta}_{\natural\ell}) \right| = \beta_{\natural}. \tag{C.23}$$

The ratio of proposals for a split proposal can then be expressed as:

$$\begin{aligned} \frac{q(\beta|z, \hat{v}, \hat{\overline{v}})}{q(\hat{\beta}|\hat{z}, v, \overline{v})} &= \frac{\beta_{\natural}}{\mathrm{Dir}(\hat{\beta}_{\flat}/\beta_{\natural}, \hat{\beta}_{\sharp}/\beta_{\natural}; \hat{\tilde{m}}_{\cdot\flat}, \hat{\tilde{m}}_{\cdot\sharp})} \\ &= \beta_{\natural} \frac{\Gamma(\hat{\tilde{m}}_{\cdot\flat})\Gamma(\hat{\tilde{m}}_{\cdot\sharp})}{\Gamma(\hat{\tilde{m}}_{\cdot\flat} + \hat{\tilde{m}}_{\cdot c})} \left(\frac{\hat{\beta}_{\flat}}{\beta_{\natural}}\right)^{1-\hat{\tilde{m}}_{\cdot\flat}} \left(\frac{\hat{\beta}_{\sharp}}{\beta_{\natural}}\right)^{1-\hat{\tilde{m}}_{\cdot\sharp}} \\ &= \frac{\Gamma(\hat{\tilde{m}}_{\cdot\flat})\Gamma(\hat{\tilde{m}}_{\cdot\sharp})}{\Gamma(\hat{\tilde{m}}_{\cdot\flat} + \hat{\tilde{m}}_{\cdot\sharp})} \frac{\hat{\beta}_{\flat}^{1-\hat{\tilde{m}}_{\cdot\flat}} \hat{\beta}_{\sharp}^{1-\hat{\tilde{m}}_{\cdot\sharp}}}{\beta_{\natural}^{1-\hat{\tilde{m}}_{\cdot\flat}-\hat{\tilde{m}}_{\cdot\sharp}}} \end{aligned} \tag{C.24}$$

Combining these expressions results in the following Hastings ratio for a local split

$$\begin{aligned} H^{\text{local}}_{\text{split-}\natural} &= \frac{p(\hat{\beta}, \hat{z}, x)}{p(\beta, z, x)} \frac{Q^{K+1}_{\text{merge-}\flat\sharp}}{Q^{K}_{\text{split-}\natural}} \frac{q(\beta|z, \hat{v}, \hat{\overline{v}})}{q(\hat{\beta}|\hat{z}, v, \overline{v})} \frac{1}{q(\hat{z}|x, v, \overline{v}, Q_{\text{split-}\natural})} \\ &= \frac{\gamma\beta_{\natural}}{\hat{\beta}_{\flat}\hat{\beta}_{\sharp}} \left[\prod_{j=1}^{D} \frac{\Gamma(\alpha\beta_{\natural})}{\Gamma(\alpha\beta_{\natural} + n_{j\cdot\natural})} \prod_{k\in\{\flat,\sharp\}} \frac{\Gamma(\alpha\hat{\beta}_k + \hat{n}_{j\cdot k})}{\Gamma(\alpha\hat{\beta}_k)}\right] \frac{p(x|\hat{z})}{p(x|z)} \\ &\quad \times \frac{Q^{K+1}_{\text{merge-}\flat\sharp}}{Q^{K}_{\text{split-}\natural}} \frac{\Gamma(\hat{\tilde{m}}_{\cdot\flat})\Gamma(\hat{\tilde{m}}_{\cdot\sharp})}{\Gamma(\hat{\tilde{m}}_{\cdot\flat} + \hat{\tilde{m}}_{\cdot\sharp})} \frac{\hat{\beta}_{\flat}^{1-\hat{\tilde{m}}_{\cdot\flat}} \hat{\beta}_{\sharp}^{1-\hat{\tilde{m}}_{\cdot\sharp}}}{\beta_{\natural}^{1-\hat{\tilde{m}}_{\cdot\flat}-\hat{\tilde{m}}_{\cdot\sharp}}} \frac{1}{q(\hat{z}|x, v, \overline{v}, Q_{\text{split-}\natural})} \\ &= \frac{\gamma\Gamma(\hat{\tilde{m}}_{\cdot b})\Gamma(\hat{\tilde{m}}_{\cdot c})}{\Gamma(\hat{\tilde{m}}_{\cdot b} + \hat{\tilde{m}}_{\cdot c})} \frac{\beta_a^{\hat{\tilde{m}}_{\cdot b}-\hat{\tilde{m}}_{\cdot c}}}{\hat{\beta}_b^{\hat{\tilde{m}}_{\cdot b}} \hat{\beta}_c^{\hat{\tilde{m}}_{\cdot c}}} \frac{p(x|\hat{z})}{p(x|z)} \frac{1}{q(\hat{z}|x, v, \overline{v}, Q_{\text{split-}\natural})} \\ &\quad \times \frac{Q^{K+1}_{\text{merge-}\flat\sharp}}{Q^{K}_{\text{split-}\natural}} \prod_{j=1}^{D} \frac{\Gamma(\alpha\beta_{\natural})}{\Gamma(\alpha\beta_{\natural} + n_{j\cdot\natural})} \prod_{k\in\{\flat,\sharp\}} \frac{\Gamma(\alpha\hat{\beta}_k + \hat{n}_{j\cdot k})}{\Gamma(\alpha\hat{\beta}_k)}, \end{aligned} \tag{C.25}$$

which matches Equation (6.31). The same algebra can be used to show that the Hastings

ratio for a local merge is

$$H^{\text{local}}_{\text{merge-}\flat\sharp} = \frac{\Gamma(\tilde{m}_{\cdot b} + \tilde{m}_{\cdot c})}{\gamma\Gamma(\tilde{m}_{\cdot b})\Gamma(\tilde{m}_{\cdot c})} \frac{\beta_b^{\tilde{m}_{\cdot b}} \beta_c^{\tilde{m}_{\cdot c}}}{\hat{\beta}_a^{\tilde{m}_{\cdot b} - \tilde{m}_{\cdot c}}} \frac{p(x|z)}{p(x|\hat{z})} \frac{q(z|x, \hat{v}, \hat{\overline{v}}, Q_{\text{split-}\natural})}{1} \times \frac{Q^{K-1}_{\text{split-}\natural}}{Q^{K}_{\text{merge-}\flat\sharp}} \prod_{j=1}^{D} \frac{\Gamma(\alpha\hat{\beta}_\natural + n_{j\cdot\natural})}{\Gamma(\alpha\hat{\beta}_\natural)} \prod_{k\in\{\flat,\sharp\}} \frac{\Gamma(\alpha\beta_k)}{\Gamma(\alpha\beta_k + \hat{n}_{j\cdot k})}, \tag{C.26}$$

where $\tilde{m} \triangleq \tilde{m}(z)$ is a function of the original z.

■ C.4 Hastings Ratios for Global Proposals

For the proposed global split of topic $\natural$, all β's and z's change. Instead of denoting the empty β with β_{K+1}, we use the notation β_E here. The ratio of posteriors for the global split can be expressed as

$$\frac{p(\hat{\beta}, \hat{z}, x)}{p(\beta, z, x)} = \left[\frac{\hat{\beta}_E}{\beta_E}\right]^{\gamma-1} \frac{p(x|\hat{z})}{p(x|z)} \gamma \prod_{k=1}^{K} \beta_k \prod_{k=1}^{K+1} \hat{\beta}_k^{-1} \prod_{j=1}^{D} \prod_{k=1}^{K} \frac{\Gamma(\alpha\beta_k)}{\Gamma(\alpha\beta_k + n_{j\cdot k})} \prod_{k=1}^{K+1} \frac{\Gamma(\alpha\hat{\beta}_k + \hat{n}_{j\cdot k})}{\Gamma(\alpha\hat{\beta}_k)} \tag{C.27}$$

The proposal ratio for the z's is

$$\frac{q(z|\hat{v}, \hat{\overline{v}})}{q(\hat{z}|v, \overline{v})} q(\tilde{\theta}_\natural|z) \tag{C.28}$$

We follow in the same steps as the previous local proposals by calculating the proposal ratio for the β's.

$$\frac{q(\beta|z)}{q(\hat{\beta}|\hat{z})} = \frac{\text{Dir}(\beta_1, \ldots, \beta_K, \beta_E; \tilde{m}_{\cdot 1}, \ldots, \tilde{m}_{\cdot K}, \gamma)}{\text{Dir}(\hat{\beta}_1, \ldots, \hat{\beta}_{K+1}, \hat{\beta}_E; \hat{\tilde{m}}_{\cdot 1}, \ldots, \hat{\tilde{m}}_{\cdot (K+1)}, \gamma)} \tag{C.29}$$

$$= \frac{\Gamma(\gamma + \sum_{k=1}^{K} \tilde{m}_{\cdot k})}{\Gamma(\gamma + \sum_{k=1}^{K+1} \hat{\tilde{m}}_{\cdot k})} \left(\frac{\beta_E}{\hat{\beta}_E}\right)^{\gamma-1} \prod_{k=1}^{K} \frac{\beta_k^{\tilde{m}_{\cdot k}-1}}{\Gamma(\tilde{m}_{\cdot k})} \prod_{k=1}^{K+1} \frac{\Gamma(\hat{\tilde{m}}_{\cdot k})}{\hat{\beta}_k^{\hat{\tilde{m}}_{\cdot k}-1}} \tag{C.30}$$

Combining these expressions, we arrive at the following Hastings ratio for a global split

$$
\begin{aligned}
H^{\text{global}}_{\text{split-}\natural} &= \frac{p(\hat{\beta},\hat{z},x)}{p(\beta,z,x)}\frac{Q^M_{K+1}}{Q^S_K}\frac{q(\beta|z,\hat{v},\hat{\overline{v}})}{q(\hat{\beta}|\hat{z},v,\overline{v})}\frac{q(z|\hat{v},\hat{\overline{v}})}{q(\hat{z}|v,\overline{v})}q(\tilde{\theta}_a|z) \\
&= \left[\frac{\hat{\beta}_E}{\beta_E}\right]^{\gamma-1}\frac{p(x|\hat{z})}{p(x|z)}\gamma\prod_{k=1}^{K}\beta_k\prod_{k=1}^{K+1}\hat{\beta}_k^{-1}\prod_{j=1}^{D}\prod_{k=1}^{K}\frac{\Gamma(\alpha\beta_k)}{\Gamma(\alpha\beta_k+n_{j\cdot k})}\prod_{k=1}^{K+1}\frac{\Gamma(\alpha\hat{\beta}_k+\hat{n}_{j\cdot k})}{\Gamma(\alpha\hat{\beta}_k)} \\
&\quad\times\frac{Q^M_{K+1}}{Q^S_K}\frac{q(z|\hat{v},\hat{\overline{v}})}{q(\hat{z}|v,\overline{v})}q(\tilde{\theta}_a|z)\frac{\Gamma(\gamma+\sum_{k=1}^{K}\tilde{m}_{\cdot k})}{\Gamma(\gamma+\sum_{k=1}^{K+1}\hat{\tilde{m}}_{\cdot k})}\left(\frac{\beta_E}{\hat{\beta}_E}\right)^{\gamma-1}\prod_{k=1}^{K}\frac{\beta_k^{\tilde{m}_{\cdot k}-1}}{\Gamma(\tilde{m}_{\cdot k})}\prod_{k=1}^{K+1}\frac{\Gamma(\hat{\tilde{m}}_{\cdot k})}{\hat{\beta}_k^{\hat{\tilde{m}}_{\cdot k}-1}} \\
&= \frac{\gamma\Gamma(\gamma+\sum_{k=1}^{K}\tilde{m}_{\cdot k})}{\Gamma(\gamma+\sum_{k=1}^{K+1}\hat{\tilde{m}}_{\cdot k})}\frac{p(x|\hat{z})}{p(x|z)}\frac{q(z|\hat{v},\hat{\overline{v}})}{q(\hat{z}|v,\overline{v})}\frac{q(\tilde{\theta}_a|x,z)}{1}\frac{Q^{K+1}_{\text{merge-}\flat\sharp}}{Q^K_{\text{split-}\natural}} \\
&\quad\times\prod_{k=1}^{K}\frac{\beta_k^{\tilde{m}_{\cdot k}}}{\Gamma(\tilde{m}_{\cdot k})}\prod_{k=1}^{K+1}\frac{\Gamma(\hat{\tilde{m}}_{\cdot k})}{\hat{\beta}_k^{\hat{\tilde{m}}_{\cdot k}}}\prod_{j=1}^{D}\prod_{k=1}^{K}\frac{\Gamma(\alpha\beta_k)}{\Gamma(\alpha\beta_k+n_{j\cdot k})}\prod_{k=1}^{K+1}\frac{\Gamma(\alpha\hat{\beta}_k+\hat{n}_{j\cdot k})}{\Gamma(\alpha\hat{\beta}_k)},
\end{aligned}
\tag{C.31}
$$

which matches Equation (6.39). Similar algebra shows that the Hastings ratio for a global merge is

$$
\begin{aligned}
H^{\text{global}}_{\text{merge-}\flat\sharp} &= \frac{\Gamma\left(\gamma+\sum_{k=1}^{K}\tilde{m}_{\cdot k}\right)}{\gamma\Gamma\left(\gamma+\sum_{k=1}^{K-1}\hat{\tilde{m}}_{\cdot k}\right)}\frac{p(x|\hat{z})}{p(x|z)}\frac{q(z|\hat{v},\hat{\overline{v}})}{q(\hat{z}|v,\overline{v})}\frac{1}{q(\tilde{\theta}_\natural|x,\hat{z})}\frac{Q^{K-1}_{\text{split-}\natural}}{Q^K_{\text{merge-}\flat\sharp}} \\
&\quad\times\prod_{k=1}^{K}\frac{\beta_k^{\tilde{m}_{\cdot k}(z)}}{\Gamma(\tilde{m}_{\cdot k}(z))}\prod_{k=1}^{K-1}\frac{\Gamma(\hat{\tilde{m}}_{\cdot k})}{\hat{\beta}_k^{\hat{\tilde{m}}_{\cdot k}}}\times\prod_{j=1}^{D}\prod_{k=1}^{K}\frac{\Gamma(\alpha\beta_k)}{\Gamma(\alpha\beta_k+n_{j\cdot k})}\prod_{k=1}^{K-1}\frac{\Gamma(\alpha\hat{\beta}_k+\hat{n}_{j\cdot k})}{\Gamma(\alpha\hat{\beta}_k)},
\end{aligned}
\tag{C.32}
$$

which matches Equation (6.40).

Appendix D

Derivations Pertaining to SV-DPGMM

In this appendix, we derive expressions related to Chapter 7. We develop the analytical distributions of the the mean parameters, μ, and the data likelihood x, when marginalizing over the Gaussian process, g. We also show the details of how these marginalizations can be exploited in the DP Sub-Cluster algorithm to propose marginalized split and merge moves.

■ D.1 Marginalization of the Gaussian Process

In this section, we derive the posterior distribution of means, μ, when marginalizing over the shading image, g. We remind the reader that the prior on the log shading image, g, is a finite realization of a Gaussian process. As such, it can be expressed a a joint Gaussian:

$$p(g) = \mathcal{N}(g; 0, \Sigma^g)\,. \tag{D.1}$$

The observations are generated from the SV-DPGMM according to

$$p(x|\mu, z, g, \Sigma^x) = \prod_{i=1}^{N} \mathcal{N}(x_i; \mu_{z_i} + g_i, \Sigma^x) = \mathcal{N}(x; \mu_z + g, \Sigma^x \otimes \mathrm{I}_N)\,. \tag{D.2}$$

Consequently, the joint distribution, $p(x, g, \mu|z, \Sigma^x)$, can then be expressed as

$$\begin{aligned} p(x, g, \mu|z, \Sigma^x) &= p(\mu)p(g)p(x|g, \mu, z, \Sigma^x), \\ &= p(\mu)\mathcal{N}(g; 0, \Sigma^g)\,\mathcal{N}(x; \mu_z + g, \Sigma^x \otimes \mathrm{I}_N)\,, \\ &= p(\mu)\mathcal{N}(g; 0, \Sigma^g)\,\mathcal{N}(g; x - \mu_z, \Sigma^x \otimes \mathrm{I}_N)\,, \\ &= p(\mu)\mathcal{N}(x; \mu_z, \Sigma^g + \Sigma^x \otimes \mathrm{I}_N)\,\mathcal{N}(g; \bullet, \bullet)\,, \end{aligned} \tag{D.3}$$

where each $\bullet$ represents some function of x, μ_z, Σ^g, and Σ^x.

Marginalizing over g is then as simple as eliminating the last Gaussian distribution in the expression above, since it is the only term that depends on g, and it integrates

to 1. This marginalization results in

$$p(x, \mu | z, \Sigma^x) = p(\mu) p(x | \mu, z, \Sigma^x) = p(\mu) \mathcal{N}(x; \mu_z, \Sigma^g + \Sigma^x \otimes \mathrm{I}_N) \,. \tag{D.4}$$

We now express each of these terms separately. The prior on the means is the product of K independent Gaussian distributions:

$$\begin{aligned} p(\mu) &= \textstyle\prod_k \mathcal{N}(\mu_k; \theta, \Sigma^\mu) \,, \\ &= (2\pi)^{-KC/2} \left|\Sigma^\mu\right|^{-K/2} \exp\left[-\frac{1}{2} \textstyle\sum_k (\mu_k - \theta)^\top \Lambda^\mu (\mu_k - \theta) \right] , \\ &= (2\pi)^{-KC/2} \left|\Sigma^\mu\right|^{-K/2} \exp\left[-\frac{1}{2} \textstyle\sum_k \sum_{m,n} (\mu_{km} - \theta_m) \Lambda^\mu_{m,n} (\mu_{kn} - \theta_n) \right] . \end{aligned} \tag{D.5}$$

The posterior on observations marginalizing out g can be expanded to

$$\begin{aligned} p(x | \mu, z, \Sigma^x) &= \mathcal{N}(x; \mu_z, \Sigma^g + \Sigma^x \otimes \mathrm{I}_N) \,, \\ &= (2\pi)^{-NC/2} \left|\Sigma^{g+x}\right|^{-1/2} \exp\left[-\frac{1}{2} (x - \mu_z)^\top \Lambda^{g+x} (x - \mu_z) \right] , \\ &= (2\pi)^{-NC/2} \left|\Sigma^{g+x}\right|^{-1/2} \exp\left[-\frac{1}{2} \sum_{i,m} \sum_{j,n} (x_{im} - \mu_{z_i m}) \Lambda^{g+x}_{im,jn} (x_{jn} - \mu_{z_j n}) \right] , \end{aligned} \tag{D.6}$$

where we denote $\Sigma^{g+x} \triangleq \Sigma^g + \Sigma^x \otimes \mathrm{I}_N$. Combining the above distributions results in the following expression for Equation (D.4):

$$\begin{aligned} p(x, \mu | z, \Sigma^x) = (2\pi)^{-(K+N)C/2} &\left|\Sigma^\mu\right|^{-K/2} \left|\Sigma^{g+x}\right|^{-1/2} \\ &\times \exp\left[-\frac{1}{2} \underbrace{\textstyle\sum_k \sum_{m,n} (\mu_{km} - \theta_m) \Lambda^\mu_{m,n} (\mu_{kn} - \theta_n)}_{A} \right] \\ &\times \exp\left[-\frac{1}{2} \underbrace{\textstyle\sum_{i,m} \sum_{j,n} (x_{im} - \mu_{z_i m}) \Lambda^{g+x}_{im,jn} (x_{jn} - \mu_{z_j n})}_{B} \right] . \end{aligned} \tag{D.7}$$

We now consider expressing the terms inside each of the exponentials separately. The

first term can be expressed as

$$\begin{aligned} A &= \sum_{k,m,n} (\mu_{km} - \theta_m)\Lambda^{\mu}_{m,n}(\mu_{kn} - \theta_n) \\ &= \sum_{k,m,n} \left(\mu_{km}\mu_{kn} - 2\mu_{km}\theta_n + \theta_m\theta_n\right)\Lambda^{\mu}_{m,n} \\ &= \theta^{\top}\Lambda^{\mu}\theta + \sum_{k,m,n} \mu_{km}\mu_{kn}\Lambda^{\mu}_{m,n} - 2\sum_{k,m} \mu_{km}[\Lambda^{\mu}\theta]_m. \end{aligned} \tag{D.8}$$

The second term can be expressed as

$$\begin{aligned} B &= \sum_{i,m,j,n} (x_{im} - \mu_{z_i m})\Lambda^{g+x}_{im,jn}(x_{jn} - \mu_{z_j n}), \\ &= \sum_{k,m,l,n} \sum_{i\in\mathcal{I}_k} \sum_{j\in\mathcal{I}_l} (x_{im} - \mu_{km})\Lambda^{g+x}_{im,jn}(x_{jn} - \mu_{ln}), \\ &= \sum_{k,m,l,n} \sum_{i\in\mathcal{I}_k} \sum_{j\in\mathcal{I}_l} (x_{im}x_{jn} - 2\mu_{km}x_{jn} + \mu_{km}\mu_{ln})\Lambda^{g+x}_{im,jn}, \\ &= x^{\top}\Lambda^{g+x}x + \sum_{k,m,\ell,n} \mu_{km}\mu_{ln} \sum_{i\in\mathcal{I}_k} \sum_{j\in\mathcal{I}_l} \Lambda^{g+x}_{im,jn} + \sum_{k,m} \mu_{km} \sum_{j,n} x_{jn}\Lambda^{g+x}_{im,jn}. \end{aligned} \tag{D.9}$$

Because x is observed, Equation (D.7) is also proportional to

$$p(x, \mu|z, \Sigma^x) \propto p(\mu|z, \Sigma^x, x). \tag{D.10}$$

Furthermore, since $p(x, \mu|z, \Sigma^x)$ is jointly Gaussian, we know that $p(\mu|z, \Sigma^x, x)$ must also be Gaussian. The general form of this Gaussian distribution can be expressed as

$$\begin{aligned} &p(\mu|z, \Sigma^x, x) = \mathcal{N}(\mu; \theta^*, \Sigma^*) \\ &= (2\pi)^{-KC/2} |\Sigma^*|^{-1/2} \exp\left[-\frac{1}{2}(\mu - \theta^*)^{\top}\Lambda^*(\mu - \theta^*)\right] \\ &= (2\pi)^{-KC/2} |\Sigma^*|^{-1/2} \\ &\quad \exp\left[-\frac{1}{2}\left(\theta^{*\top}\Lambda^*\theta^* + \sum_{k,m,\ell,n} \mu_{km}\mu_{\ell n}\Lambda^*_{km,\ell n} - 2\sum_{k,m} \mu_{km} \sum_{\ell,n} \Lambda^*_{km,\ell n}\theta^*_{\ell n}\right)\right] \end{aligned} \tag{D.11}$$

Matching μ terms from Equation (D.11) with Equations (D.8)–(D.9) leads to the fol-

lowing expressions for θ^* and Σ^*:

$$\mu_{km}\mu_{ln} \quad \rightarrow \quad \Lambda^*_{km,ln} = \Lambda^{\mu}_{m,n} \quad + \sum_{i\in\mathcal{I}_k}\sum_{j\in\mathcal{I}_l} \Lambda^{g+x}_{im,jn}, \qquad \forall k = l \tag{D.12}$$

$$\mu_{km}\mu_{ln} \quad \rightarrow \quad \Lambda^*_{km,ln} = \quad \sum_{i\in\mathcal{I}_k}\sum_{j\in\mathcal{I}_l} \Lambda^{g+x}_{im,jn}, \qquad \forall k \neq l \tag{D.13}$$

$$\mu_{km} \quad \rightarrow \quad [\Lambda^*\theta^*]_{km} = [\Lambda^{\mu}\theta]_m + \sum_{i\in\mathcal{I}_k}\sum_{j}\sum_{n} x_{jn}\Lambda^{g+x}_{im,jn} \tag{D.14}$$

The posterior mean on μ, denoted θ^*, is then described with the linear system of equations above, and can be found according to

$$\theta^* = \Sigma^*[\Lambda^*\theta^*]. \tag{D.15}$$

■ D.2 Marginalization of the Gaussian Process and the Means

We now show how a similar marginalization scheme to the one used above can be used to marginalize over the mean parameters and the Gaussian process. We note the following relationship

$$p(x|z,\Sigma^x) = \frac{p(x,\mu|z,\Sigma^x)}{p(\mu|z,\Sigma^x,x)}. \tag{D.16}$$

Combining Equations (D.7) and (D.11), results in

$$\begin{aligned} &p(x|z,\Sigma^x) \\ &= \frac{(2\pi)^{-(K+N)C/2}\left|\Sigma^{\mu}\right|^{-K/2}\left|\Sigma^{g+x}\right|^{-1/2}\exp\left[-\frac{1}{2}\theta^\top\Lambda^{\mu}\theta - \frac{1}{2}x^\top\Lambda^{g+x}x\right]}{(2\pi)^{-KC/2}\left|\Sigma^*\right|^{-1/2}\exp\left[-\frac{1}{2}\theta^{*\top}\Lambda^*\theta^*\right]}, \\ &= \frac{\left|\Lambda^{\mu}\right|^{K/2}\left|\Lambda^{g+x}\right|^{1/2}}{(2\pi)^{NC/2}\left|\Lambda^*\right|^{1/2}}\exp\left[-\frac{1}{2}\theta^\top\Lambda^{\mu}\theta - \frac{1}{2}x^\top\Lambda^{g+x}x + \frac{1}{2}\theta^{*\top}\Lambda^*\theta^*\right]. \end{aligned} \tag{D.17}$$

Because z are the labels corresponding to a DPMM, the prior distribution on z follows the Chinese Restaurant Process (CRP)

$$p(z) = \mathrm{CRP}(z;\alpha) = \frac{\alpha^K\Gamma(\alpha)}{\Gamma(\alpha+N)}\prod_{k=1}^{K}\Gamma(N_k), \tag{D.18}$$

where α is the concentration parameter, and N_k counts the number of observations

associated with cluster k. The posterior on z can then be expressed as

$$\begin{aligned} p(z|x,\Sigma^x) &\propto p(x,z|\Sigma^x), \\ &= p(z)p(x|z,\Sigma^x), \\ &\propto \mathrm{CRP}(z;\alpha)\,|\Lambda^*|^{-1/2}\exp\left[\frac{1}{2}\theta^{*\top}\Lambda^*\theta^*\right], \\ &= |\Lambda^*|^{-1/2}\exp\left[\frac{1}{2}\theta^{*\top}\Lambda^*\theta^*\right]\frac{\alpha^K\Gamma(\alpha)}{\Gamma(\alpha+N)}\prod_{k=1}^{K}\Gamma(N_k). \end{aligned} \tag{D.19}$$

■ D.3 Marginalized Splits and Merges

Following the work in Chapter 5, we propose splits and merges via the DP Sub-Cluster method. A proposed split or merge is subjected to a Hastings ratio which either accepts or rejects the proposed move. Because distributions in the SV-DPGMM are conjugate, it is simple to extend Chapter 5. The only slight variation is that Σ^x is not altered, since marginalizing over g requires conditioning on Σ^x. This results in a proposed split of cluster $\natural$ into clusters $\hat{k}$ and $\sharp$ being accepted with the following Hastings ratio

$$H^{\text{SV-DPGMM}}_{\text{split-}\natural} = \frac{p(\hat{z}|x,\Sigma^x)}{p(z|x,\Sigma^x)}\frac{Q^{K+1}_{\text{merge-}\flat\sharp}}{Q^{K}_{\text{split-}\natural}}\prod_{\{i;z_i=\natural\}}\frac{\tilde{\pi}_\flat\,\mathcal{N}(x_i;\tilde{\mu}_\flat,\Sigma^x)+\tilde{\pi}_\sharp\,\mathcal{N}(x_i;\tilde{\mu}_\sharp,\Sigma^x)}{\tilde{\pi}_{\hat{z}_i}\,\mathcal{N}(x_i;\tilde{\mu}_{\hat{z}_i},\Sigma^x)}, \tag{D.20}$$

where $Q^K_{\text{split-}\natural}$ is the probability of proposing to split one of the K clusters, $Q^K_{\text{merge-}\flat\sharp}$ is the probability of merging two of the K clusters, $\hat{z}$ are the proposed cluster assignments, and $\overline{\pi}$ and $\overline{\mu}$ are the sub-cluster parameters as defined in Chapter 5. We propose K splits uniformly, resulting in $Q^K_{\text{split-}\natural}=1$. Similarly, we propose to merge all possible pairs of clusters, resulting in $Q^K_{\text{merge-}\flat\sharp}=1$. This simplifies the Hastings ratio to the following

$$H^{\text{SV-DPGMM}}_{\text{split-}\natural} = \frac{p(\hat{z}|x,\Sigma^x)}{p(z|x,\Sigma^x)}\prod_{\{i;z_i=\natural\}}\frac{\tilde{\pi}_\flat\,\mathcal{N}(x_i;\tilde{\mu}_\flat,\Sigma^x)+\tilde{\pi}_\sharp\,\mathcal{N}(x_i;\tilde{\mu}_\sharp,\Sigma^x)}{\tilde{\pi}_{\hat{z}_i}\,\mathcal{N}(x_i;\tilde{\mu}_{\hat{z}_i},\Sigma^x)}. \tag{D.21}$$

Similarly, the Hastings ratio for a proposed merge of clusters $\flat$ and $\sharp$ can be well approximated with

$$H^{\text{SV-DPGMM}}_{\text{merge-}\flat\sharp} \approx \frac{p(\hat{z}|x,\Sigma^x)}{p(z|x,\Sigma^x)}\prod_{\{i;z_i=\natural\}}\frac{\tilde{\pi}_{z_i}\,\mathcal{N}(x_i;\tilde{\mu}_{z_i},\Sigma^x)}{\tilde{\pi}_\flat\,\mathcal{N}(x_i;\mu_\flat,\Sigma^x)+\tilde{\pi}_\sharp\,\mathcal{N}(x_i;\tilde{\mu}_\sharp,\Sigma^x)}. \tag{D.22}$$

An approximation is necessary for the same reasons one was needed in the non-deterministic merge moves of Chapter 5.

List of Symbols

The following tables summarize the symbols used in this thesis. Most of the notation is consistent throughout the entire thesis, but the specific symbols used in each chapter are summarized separately for convenience.

Global Notation of Variables

Symbol	Definition
x	Set of all observed data, $\{x_1, \ldots, x_N\}$
z	Set of discrete labels, $\{z_1, \ldots, z_N\}$
θ	Generic parameters of an arbitrary distribution
μ	Mean of a Gaussian distribution
σ^2	Variance of a Gaussian distribution
Σ°	Covariance matrix of the Gaussian distribution of $\circ$
Λ°	Precision matrix of the Gaussian distribution of $\circ$
λ	Hyper-parameters of an arbitrary distribution
I_N	$N \times N$ identity matrix

Global Notation of Indexing

Symbol	Definition
$(\circ)_i$	The i^{th} element of $(\circ)$
i	Index into N, $i \in \{1, \ldots, N\}$
k	Index into K, $k \in \{1, \ldots, K\}$
$\backslash i$	Excluding index i
$\mathcal{I}$	Set of indices, $\mathcal{I} \subseteq \{1, \ldots, N\}$
N	Number of data points
N_k	Number of points with label k
K	Number of unique labels in z

Global Miscellaneous Notation

Symbol	Definition
$\hat{\circ}$	A proposed sample of $\circ$
H	The Hastings ratio or Reversible Jump MCMC ratio
J	Jacobian matrix
h	A 2D filter
$p(\circ)$	The probability distribution of $\circ$
$q(\circ)$	A user-specific proposal distribution for $\circ$
$\mathbb{E}_{\circ}[f(\circ)]$	Expected value of the function, f, of the random variable, $\circ$
$f_{\circ}(\triangle;\lambda)$	A distribution for $\circ$, parametrized by λ, and evaluated at $\triangle$
$\otimes$	Kroenecker product

Shape Sampling Notation

Symbol	Definition
o	Auxiliary ordering random variable
φ	Level-set function
α	Curve-length penalty
C	The curve implied by z
T_n	Topological number of foreground region
$T_{\overline{n}}$	Topological number of background region
T_n^+	Extended topological number of foreground region
$T_{\overline{n}}^+$	Extended topological number of background region

Shape Sampling with Dynamics Notation

Symbol	Definition
a	Appearance model
$\mathfrak{z}$	Layer-ordering permutation
v	Visible-layer image
g	Smooth Gaussian process flow
f	Smooth + Independent Gaussian process flow
t	An index into time
$\mathcal{O}$	The set of observed pixels
$\mathcal{D}$	The set of disoccluded pixels
$\mathcal{R}$	The set of revealed pixels

Dirichlet Process Mixture Models

Symbol	Definition
N	Number of observations
N_k	Number of points associated with cluster k
i	Index into the data points, $\{1, \dots, N\}$
ℓ, r	Indices for the left and right sub-clusters, respectively
k	Index into the unique cluster labels, $\{1, 2, \dots\}$
h	Index into the unique sub-cluster labels, $\{\ell, r\}$
$\mathcal{I}_k$	Indices of points assigned to cluster k
$\mathcal{I}_{kh}$	Indices of points assigned to cluster k and sub-cluster h
K	Number of non-empty clusters
x	Set of observations
α	Concentration parameter of the DP
λ	Hyper-parameters of the base-measure
π	Infinite-length mixture weights
θ	Infinite-length mixture parameters
z	Cluster assignments
v	The set of all regular-cluster variables, z, π, θ
$\overline{\pi}_k$	Pair of sub-cluster weights, $\{\overline{\pi}_{k\ell}, \overline{\pi}_{kr}\}$ for regular-cluster k
$\overline{\theta}_k$	Pair of sub-cluster parameters, $\{\overline{\theta}_{k\ell}, \overline{\theta}_{kr}\}$ for regular-cluster k
$\overline{z}$	Sub-cluster assignments
$\overline{v}$	The set of all sub-cluster variables, $\overline{z}, \overline{\pi}, \overline{\theta}$
$\tilde{\pi}$	Temporary weights used in constructing proposals
$\tilde{\theta}$	Temporary parameters used in constructing proposals
$\natural$	Index of cluster that is to be split
$\flat, \sharp$	Indices of clusters that are to be merged
$Q^K_{\text{split-}\natural}$	Probability of proposing a split with K non-empty components
$Q^K_{\text{merge-}\flat\sharp}$	Probability of proposing a merge with K non-empty components

Hierarchical Dirichlet Processes

Symbol	Definition
D	Number of documents
N_j	Number of words in document j
K	Number of non-empty topics
j	Index into the documents, $\{1, \ldots, D\}$
i	Index into the words, $\{1, \ldots, N_j\}$
ℓ, r	Indices for the left and right sub-clusters, respectively
k	Index into the unique cluster labels, $\{1, \ldots, K\}$
h	Index into the unique sub-cluster labels, $\{\ell, r\}$
$\mathcal{I}_k$	Indices of words assigned to cluster k
$\mathcal{I}_{kh}$	Indices of words assigned to cluster k and sub-cluster h
κ_{jt}	Dish assignment of table t in restaurant j
τ_{ji}	Table assignment of customer i in restaurant j
m_{jk}	Number of tables serving dish k in restaurant j
n_{jtk}	Number of customers at table t eating dish k in restaurant j
$m_{j\cdot}$	Number of tables in restaurant j
$m_{\cdot k}$	Number of tables in franchise serving dish k
$n_{jt\cdot}$	Number of customers at table t in restaurant j
$n_{j\cdot k}$	Number of customers eating dish k in restaurant j
$n_{j\cdot\cdot}$	Number of customers in restaurant j
x	Corpus of documents
γ	Concentration parameter of the global-level DP
α	Concentration parameter of the document-level DP
λ	Hyper-parameters for the global-level DP base measure
β	Infinite-length global topic weights
π_j	Infinite-length topic weights for document j
θ	Infinite-length topic parameters
z	Topic assignments
v	The set of all regular-topic variables, z, β, π, θ

Hierarchical Dirichlet Processes (cont.)

Symbol	Definition
$\overline{\beta}_k$	Pair of global sub-topic weights, $\{\overline{\beta}_{k\ell}, \overline{\beta}_{kr}\}$ for regular-cluster k
$\overline{\pi}_{jk}$	Pair of sub-topic weights, $\{\overline{\pi}_{k\ell}, \overline{\pi}_{kr}\}$ for document j and regular-cluster k
$\overline{\theta}_k$	Pair of sub-topic parameters, $\{\overline{\theta}_{k\ell}, \overline{\theta}_{kr}\}$ for regular-cluster k
$\overline{z}$	Sub-topic assignments
$\overline{v}$	The set of all sub-topic variables, $\overline{z}, \overline{\beta}, \overline{\pi}, \overline{\theta}$
$\tilde{\beta}$	Temporary global weights used in constructing proposals
$\tilde{\pi}$	Temporary document-specific weights used in constructing proposals
$\tilde{\theta}$	Temporary parameters used in constructing proposals
$\natural$	Index of topic that is to be split
$\flat, \sharp$	Indices of topics that are to be merged
$Q^K_{\text{split-}\natural}$	Probability of proposing a split with K non-empty components
$Q^K_{\text{merge-}\flat\sharp}$	Probability of proposing a merge with K non-empty components

Intrinsic Images via SV-DPGMM

Symbol	Definition
N	Number of pixels in the image
N_k	Number of points associated with cluster k
i, j	Indices into the pixels, $\{1, \ldots, N\}$
m, n	Indices into the three color channels
k, l	Indices into the unique cluster labels, $\{1, 2, \ldots\}$
K	Number of non-empty clusters
x	The observed image in the log domain
α	Concentration parameter of the DP
κ	Covariance kernel for g
σ_g^2	Signal variance of κ
ν	Matérn kernel smoothness for kernel κ
l	Characteristic length-scale of κ
λ_g	Set of hyper-parameters, $\{\kappa, \sigma_g^2, \nu, l\}$ for g
θ	3×1 hyper-parameter representing mean of μ_k
Σ^μ	3×3 hyper-parameter covariance of μ_k
λ_μ	Set of hyper-parameters, $\{\theta, \Sigma^\mu\}$, for μ_k
S_Σ	Discrete set of 3×3 observation covariances
π	Infinite-length mixture weights
μ_k	3×1 mean of cluster k
z	Cluster assignments
g	Gaussian process log-shading image
μ_z	Log-reflectance image
Σ^x, Λ^x	3×3 Toeplitz observation covariance and precision matrices
Σ^g, Λ^g	$3N \times 3N$ Topelitz log-shading covariance and precision matrices
$\mathcal{I}$	Indices of pixels assigned to cluster k
$\mathcal{S}$	Randomly sub-sampled indices for inference
$\natural$	Index of cluster that is to be split
$\flat, \sharp$	Indices of clusters that are to be merged
$Q^K_{\text{split-}\natural}$	Probability of proposing a split with K non-empty components
$Q^K_{\text{merge-}\flat\sharp}$	Probability of proposing a merge with K non-empty components

Bibliography

[1] C. E. Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *Annals of Statistics*, 2(6):1152–1174, 1974.

[2] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[3] S. Baker, D. Scharstein, J.P. Lewis, S. Roth, M.J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. In *International Conference on Computer Vision*, 2007.

[4] J. Barron and J. Malik. Shape, illumination, and reflectance from shading. Technical report, Univeristy of California, Berkeley, 2013.

[5] H. Barrow and J. Tenenbaum. Recovering intrinsic scene characteristics from images. In *Computer Vision Systems*, 1978.

[6] G. Bertrand. Simple points, topological numbers and geodesic neighborhoods in cubic grids. *Pattern Recognition Letters*, 15:1003–1011, October 1994.

[7] A. Blake. Boundary conditions for lightness computation in Mondrian world. In *Computer Vision, Graphics, and Image Processing*, 1985.

[8] D. M. Blei and P. I. Frazier. Distance dependent Chinese restaurant processes. *Journal of Machine Learning Research*, 12:2461–2488, 2011.

[9] D. M. Blei, T. L. Griffiths, M. I. Jordan, and J. B. Tenenbaum. Hierarchical topic models and the nested Chinese restaurant process. In *Neural Information Processing Systems*, 2003.

[10] D. M. Blei and M. I. Jordan. Variational inference for Dirichlet process mixtures. *Bayesian Analysis*, 1:121–144, 2005.

[11] T. J. Broida and R. Chellappa. Estimation of object motion parameters from noisy images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1):90–99, January 1986.

[12] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *European Conference on Computer Vision*, 2010.

[13] T. Brox and J. Malik. Large displacement optical flow: Descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 500–513, 2011.

[14] T. Brox and J. Weickert. Level set image segmentation with multiple regions. *IEEE Transactions on Image Processing*, 15(10):3213–3218, October 2006.

[15] M. Bryant and E. Sudderth. Truly nonparametric online variational inference for Hierarchical Dirichlet processes. In *Neural Information Processing Systems*, 2012.

[16] C. A. Bush and S. N. MacEachern. A semiparametric Bayesian model for randomised block designs. *Biometrika*, 83:275–285, 1996.

[17] T Cham and J. M. Rehg. A multiple hypothesis approach to figure tracking. In *Computer Vision and Pattern Recognition*, 1999.

[18] J. Chang and J. W. Fisher III. Analysis of orientation and scale in smoothly varying textures. *International Conference on Computer Vision*, 2009.

[19] J. Chang and J. W. Fisher III. Efficient MCMC sampling with implicit shape representations. In *Computer Vision and Pattern Recognition*, June 2011.

[20] J. Chang and J. W. Fisher III. Efficient topology-controlled sampling of implicit shapes. In *IEEE International Conference on Image Processing*, September 2012.

[21] J. Chang and J. W. Fisher III. Parallel sampling of DP mixture models using sub-clusters splits. In *Neural Information Processing Systems*, December 2013.

[22] J. Chang and J. W. Fisher III. Topology-constrained layered tracking with latent flow. In *International Conference on Computer Vision*, December 2013.

[23] S. Chen and R. J. Radke. Markov chain Monte Carlo shape sampling using level sets. *NORDIA, in conjunction with ICCV*, 2009.

[24] P. Y. Choi and M. Hebert. Learning and predicting moving object trajectory: a piecewise trajectory segment approach. Technical Report CMU-RI-TR-06-42, Robotics Institute, Pittsburgh, PA, August 2006.

[25] K Choo and D. J. Fleet. People tracking using hybrid Monte Carlo filtering. In *International Conference on Computer Vision*, 2001.

[26] L. D. Cohen and I. Cohen. Finite-element method for active contour models and balloons for 2-d and 3-d images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1131–1147, November 1993.

[27] D. B. Dahl. An improved merge-split sampler for conjugate Dirichlet process mixture models. Technical report, University of Wisconsin - Madison Dept. of Statistics, 2003.

[28] T. Darrell and A. Pentland. Robust estimation of a multi-layered motion representation. In *IEEE Workshop on Visual Motion*, 1991.

[29] D. Demirdjian, L. Taycher, G. Shakhnarovich, K. Grauman, and T. Darrell. Avoiding the "streetlight effect": tracking by exploring likelihood modes. In *International Conference on Computer Vision*, 2005.

[30] M. D. Escobar and M. West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90(430):577–588, 1995.

[31] A. C. Fan, J. W. Fisher III, W. M. Wells III, J. J. Levitt, and A. S. Willsky. MCMC curve sampling for image segmentation. *Medical Image Computing and Computer Assisted Intervention*, 2007.

[32] S. Favaro and Y. W. Teh. MCMC for normalized random measure mixture models. *Statistical Science*, 28(3):335–359, 2013.

[33] T. S. Ferguson. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230, 1973.

[34] E. B. Fox, E. B. Sudderth, M. I. Jordan, and A. S. Willsky. An HDP-HMM for systems with state persistence. In *International Conference on Machine Learning*, July 2008.

[35] K. Fragkiadaki and Jianbo Shi. Detection free tracking: Exploiting motion and topology for segmenting and tracking under entanglement. In *Computer Vision and Pattern Recognition*, 2011.

[36] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. In *International Journal of Computer Vision*, 2000.

[37] B. V. Funt, M. S. Drew, and M. Brockington. Recovering shading from color images. In *European Conference on Computer Vision*, 1992.

[38] Y. Gal and Z. Ghahramani. Pitfalls in the use of parallel inference for the Dirichlet process. In *Workshop on Big Learning, NIPS*, 2013.

[39] P. V. Gehler, R. Carsten, M. Kiefel, L. Zhang, and B. Schölkopf. Recovering intrinsic images with a global sparsity prior on reflectance. In *Neural Information Processing Systems*, 2011.

[40] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, November 1984.

[41] S. Ghosh, A. B. Ungureanu, E. B. Sudderth, and D. Blei. Spatial distance dependent Chinese restaurant process for image segmentation. In *Neural Information Processing Systems*, 2011.

[42] W. R. Gilks, W. Richardson, and D. J. Spiegelhalter. *Markov Chain Monte Carlo in Practice*. Chapman & Hall / CRC Press, 1996.

[43] N.J. Gordon, D.J. Salmond, and A. F M Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEEE Transacations of Radar and Signal Processing*, 140(2):107–113, April 1993.

[44] P. J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.

[45] P. J. Green and S. Richardson. Modelling heterogeneity with and without the Dirichlet process. *Scandinavian Journal of Statistics*, pages 355–375, 2001.

[46] U. Grenander and M. I. Miller. Computational anatomy: an emerging discipline. *Quarterly of Applied Mathematics*, LVI(4):617–694, 1998.

[47] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101:5228–5235, April 2004.

[48] R Grosse, M. K. Johnson, E. Adelson, and W. T. Freeman. A ground-truth dataset and baseline evaluations for intrinsic image algorithms. In *International Conference on Computer Vision*, 2009.

[49] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph based video segmentation. In *Computer Vision and Pattern Recognition*, 2010.

[50] X. Han, C. Xu, and J. L. Prince. A topology preserving level set method for geometric deformable models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(6):755–768, 2003.

[51] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[52] D. Heath and W. Sudderth. De Finetti's theorem on exchangeable variables. *American Statistician*, 30(4):188–189, 1976.

[53] M. Heiler and C. Schnorr. Natural image statistics for natural image segmentation. *International Conference on Computer Vision*, 2003.

[54] B. K. P. Horn. *Robot Vision*. MIT Press, Cambridge, MA, 1986.

[55] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 1981.

[56] N. Houhou, J.P. Thiran, and X. Bresson. Fast texture segmentation model based on the shape operator and active contour. *Computer Vision and Pattern Recognition*, 2008.

[57] M. Hughes and E. Sudderth. Memoized online variational inference for Dirichlet process mixture models. In *Neural Information Processing Systems*, 2013.

[58] M. C. Hughes, E. B. Fox, and E. B. Sudderth. Effective split-merge Monte Carlo methods for nonparametric models of sequential data. In *Neural Information Processing Systems*, December 2012.

[59] M. Isard and A. Blake. A mixed-state condensation tracker with automatic model-switching. In *International Conference on Computer Vision*, 1998.

[60] H. Ishwaran and L. F. James. Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96:161–173, 2001.

[61] H. Ishwaran and M. Zarepour. Exact and approximate sum-representations for the Dirichlet process. *Canadian Journal of Statistics*, 30:269–283, 2002.

[62] Nebojsa J. and B. J. Frey. Learning flexible sprites in video layers. In *Computer Vision and Pattern Recognition*, 2001.

[63] S. Jain and R. Neal. A split-merge Markov chain Monte Carlo procedure for the Dirichlet process mixture model. *Journal of Computational and Graphical Statistics*, 13:158–182, 2000.

[64] S. Jain and R. Neal. Splitting and merging components of a nonconjugate Dirichlet process mixture model. *Bayesian Analysis*, 2(3):445–472, 2007.

[65] M. R. Jerrum and A. J. Sinclair. Approximating the permanent. *SIAM Journal on Computing*, 18:1149–1178, 1989.

[66] M. I. Jordan. Graphical models. *Statistical Science*, 19(1):140–155, 2004.

[67] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME*, 82(1):35–45, 1960.

[68] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, V1(4):321–331, January 1988.

[69] J. Kim, J. W. Fisher III, A. Yezzi, M. Cetin, and A.S. Willsky. A nonparametric statistical method for image segmentation using information theory and curve evolution. *IEEE Transactions on Image Processing*, 14:1486–1502, October 2005.

[70] K. Kim, D. Lee, and I. Essa. Gaussian process regression flow for analysis of motion trajectories. In *International Conference on Computer Vision*, 2011.

[71] D. E. Knuth. *The Art of Computer Programming.* Addison-Wesley, Boston, 1969.

[72] T. Y. Kong and A. Rosenfeld. Digital topology: introduction and survey. *Computer Vision, Graphics, and Image Processing*, 48:357–393, December 1989.

[73] B. Kulis and M. Jordan. Revisiting k-means: New algorithms via Bayesian nonparametrics. In *International Conference on Machine Learning*, 2012.

[74] K. Kurihara, M. Welling, and Y. W. Teh. Collapsed variational Dirichlet process mixture models. In *International Joint Conference on Artificial Intelligence*, 2007.

[75] E. Land and J. McCann. Lightness and retinex theory. In *Journal of the Optical Society of America*, 1971.

[76] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[77] Y. J. Lee, J. Kim, and K. Grauman. Key-segments for video object segmentation. In *International Conference on Computer Vision*, 2011.

[78] Y. Li, J. Sun, C.K. Tang, and H.Y. Shum. Lazy snapping. In *SIGGRAPH*, 2004.

[79] P. Liang, M. I. Jordan, and B. Taskar. A permutation-augmented sampler for DP mixture models. In *International Conference on Machine Learning*, 2007.

[80] D. Lin and J. W. Fisher III. Efficient Sampling from Combinatorial Space via Bridging. In *Artificial Intelligence and Statistics*, 2012.

[81] D. Lin, E. Grimson, and J. W. Fisher III. Construction of dependent Dirichlet processes based on Poisson processes. In *Neural Information Processing Systems*, 2010.

[82] C. Liu, W. T. Freeman, E. H. Adelson, and Y. Weiss. Human-assisted motion annotation. In *Computer Vision and Pattern Recognition*, 2008.

[83] D. Lovell, R. P. Adams, and V. K. Mansingka. Parallel Markov chain Monte Carlo for Dirichlet process mixtures. In *Workshop on Big Learning, NIPS*, 2012.

[84] T. Ma and L. J. Latecki. Maximum weight cliques with mutex constraints for video object segmentation. In *Computer Vision and Pattern Recognition*, 2012.

[85] S. N. MacEachern. Estimating normal means with a conjugate style Dirichlet process prior. In *Communications in Statistics: Simulation and Computation*, 1994.

[86] S. N. MacEachern and P. Müller. Estimating mixture of Dirichlet process models. *Journal of Computational and Graphical Statistics*, 7(2):223–238, June 1998.

[87] D. Malioutov, J. Johnson, M. Choi, and A. Willsky. Low-rank variance approximation in gmrf models: Single and multiscale approaches. *IEEE Transactions on Signal Processing*, 56(10):4621–4634, 2008.

[88] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *International Conference on Computer Vision*, 2001.

[89] Y. Matsushita, K. Nishino, Ikeuchi K., and Sakauchi M. Illumination normalization with time-dependent intrinsic images for video surveillance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10):1336–1347, 2004.

[90] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[91] J. W. Miller and M. T. Harrison. A simple example of Dirichlet process mixture inconsistency for the number of components. In *Neural Information Processing Systems*, 2013.

[92] R. Neal. Bayesian mixture modeling. In *International Workshop on Maximum Entropy and Bayesian Methods of Statistical Analysis*, 1992.

[93] R. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, June 2000.

[94] D. Newman, A. Asuncion, P. Smyth, and M. Welling. Distributed algorithms for topic models. *Journal of Machine Learning Research*, 10:1801–1828, December 2009.

[95] S. Osher and R. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer, 2002.

[96] O. Papaspiliopoulos and G. O. Roberts. Retrospective Markov chain Monte Carlo methods for Dirichlet process hierarchical models. *Biometrika*, 95(1):169–186, 2008.

[97] J. Pitman. Combinatorial stochastic processes. Technical report, U.C. Berkeley Dept. of Statistics, 2002.

[98] W. K. Pratt. *Digital Image Processing*. John Wiley & Sons, 2007.

[99] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[100] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006.

[101] Y. Rathi, N. Vaswani, and A. Tannenbaum. A generic framework for tracking using particle filter with dynamic shape prior. *IEEE Transactions on Image Processing*, 16(5):1370–1382, May 2007.

[102] I. Reid and K. Connor. Multiview segmentation and tracking of dynamic occluding layers. *Image and Vision Computing*, 2010.

[103] S. L. Scott. Bayesian methods for hidden Markov models: Recursive computing in the 21st century. *Journal of the American Statistical Association*, 97:337–351, 2002.

[104] F. Ségonne. Active contours under topology control–genus preserving level sets. *International Journal of Computer Vision*, 79:107–117, August 2008.

[105] J.A. Sethian. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Cambridge University Press, second edition, June 1999.

[106] J. Sethuraman. A constructive definition of Dirichlet priors. *Statstica Sinica*, 4:639–650, 1994.

[107] L. Shen, P. Tan, and S. Lin. Intrinsic image decomposition with non-local texture cues. In *Computer Vision and Pattern Recognition*, 2008.

[108] Y. Shi and W.C. Karl. Real-time tracking using level sets. In *Computer Vision and Pattern Recognition*, 2005.

[109] B. W. Silverman. Spline smoothing: The equivalent variable kernel method. *Annals of Statistics*, 12(3):898–916, 1984.

[110] P. Sollich and C. K. I. Williams. Using the equivalent kernel to understand Gaussian process regression. In *Neural Information Processing Systems*, 2005.

[111] M. L. Stein. A kernel approximation to the kriging predictor of a spatial process. *Annals of the Institute of Statistical Mathematics*, 43(1):61–75, 1991.

[112] E. B. Sudderth. *Graphical Models for Visual Object Recognition and Tracking*. PhD thesis, Massachusetts Institute of Technology, 2006.

[113] D. Sun, E. Sudderth, and M. J. Black. Layered image motion with explicit occlusions, temporal consistency, and depth ordering. In *NIPS*, 2010.

[114] D. Sun, E.B. Sudderth, and M.J. Black. Layered segmentation and optical flow estimation over time. In *Computer Vision and Pattern Recognition*, 2012.

[115] M. F. Tappen, W. T. Freeman, and E. H. Adelson. Recovering intrinsic images from a single image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(9):1459–1472, 2005.

[116] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[117] Y. W. Teh, K. Kurihara, and M. Welling. Collapsed variational inference for HDP. In *Neural Information Processing Systems*, volume 20, 2008.

[118] D. Tsai, M. Flagg, and J. M. Rehg. Motion coherent tracking with multi-label MRF optimization. In *British Machine Vision Conference*, 2010.

[119] J. N. Tsitsiklis. Efficient algorithms for globally optimal trajectories. *IEEE Transactions on Automatic Control*, 40(9):1528–1538, September 1995.

[120] R. Urtasun, D.J. Fleet, and P. Fua. 3d people tracking with Gaussian process dynamical models. In *Computer Vision and Pattern Recognition*, 2006.

[121] L. A. Vese and T. F. Chan. A multiphase level set framework for image segmentation using the mumford and shah model. *International Journal of Computer Vision*, 50(3):271–293, 2002.

[122] C. Wang and D Blei. A split-merge MCMC algorithm for the Hierarchical Dirichlet process. arXiv:1207.1657 [stat.ML], 2012.

[123] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *IEEE Transactions on Image Processing*, 1994.

[124] Y. Weiss. Smoothness in layers: Motion segmentation using nonparametric mixture estimation. In *Computer Vision and Pattern Recognition*, 1997.

[125] Y. Weiss. Deriving intrinsic images from image sequences. In *International Conference on Computer Vision*, 2001.

[126] M. West, P. Müller, and S. N. MacEachern. Hierarchical priors and mixture models, with application in regression and density estimation. *Aspects of Uncertainity*, pages 363–386, 1994.

[127] S. A. Williamson, A. Dubey, and E. P. Xing. Parallel Markov chain Monte Carlo for nonparametric mixture models. In *International Conference on Machine Learning*, 2013.

[128] C. Yang, R. Duraiswami, N. A. Gumerov, and L. Davis. Improved fast gauss transform and efficient kernel density estimation. In *International Conference on Computer Vision*, 2003.