Graphical Models for Visual Object Recognition and Tracking

by

Erik B. Sudderth

B.S., Electrical Engineering, University of California at San Diego, 1999

S.M., Electrical Engineering and Computer Science, M.I.T., 2002

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

May, 2006

© 2006 Massachusetts Institute of Technology
All Rights Reserved.

Signature of Author: _____
Department of Electrical Engineering and Computer Science
May 26, 2006

Certified by: _____
William T. Freeman
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Certified by: _____
Alan S. Willsky
Edwin Sibley Webster Professor of Electrical Engineering
Thesis Supervisor

Accepted by: _____
Arthur C. Smith
Professor of Electrical Engineering
Chair, Committee for Graduate Students

Graphical Models for Visual Object Recognition and Tracking

by Erik B. Sudderth

Submitted to the Department of Electrical Engineering
and Computer Science on May 26, 2006
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Electrical Engineering and Computer Science

Abstract

We develop statistical methods which allow effective visual detection, categorization, and tracking of objects in complex scenes. Such computer vision systems must be robust to wide variations in object appearance, the often small size of training databases, and ambiguities induced by articulated or partially occluded objects. Graphical models provide a powerful framework for encoding the statistical structure of visual scenes, and developing corresponding learning and inference algorithms. In this thesis, we describe several models which integrate graphical representations with nonparametric statistical methods. This approach leads to inference algorithms which tractably recover high–dimensional, continuous object pose variations, and learning procedures which transfer knowledge among related recognition tasks.

Motivated by visual tracking problems, we first develop a nonparametric extension of the belief propagation (BP) algorithm. Using Monte Carlo methods, we provide general procedures for recursively updating particle–based approximations of continuous sufficient statistics. Efficient multiscale sampling methods then allow this nonparametric BP algorithm to be flexibly adapted to many different applications. As a particular example, we consider a graphical model describing the hand's three–dimensional (3D) structure, kinematics, and dynamics. This graph encodes global hand pose via the 3D position and orientation of several rigid components, and thus exposes local structure in a high–dimensional articulated model. Applying nonparametric BP, we recover a hand tracking algorithm which is robust to outliers and local visual ambiguities. Via a set of latent occupancy masks, we also extend our approach to consistently infer occlusion events in a distributed fashion.

In the second half of this thesis, we develop methods for learning hierarchical models of objects, the parts composing them, and the scenes surrounding them. Our approach couples topic models originally developed for text analysis with spatial transformations, and thus consistently accounts for geometric constraints. By building integrated scene models, we may discover contextual relationships, and better exploit partially labeled training images. We first consider images of isolated objects, and show that sharing parts among object categories improves accuracy when learning from few examples.

Turning to multiple object scenes, we propose nonparametric models which use Dirichlet processes to automatically learn the number of parts underlying each object category, and objects composing each scene. Adapting these transformed Dirichlet processes to images taken with a binocular stereo camera, we learn integrated, 3D models of object geometry and appearance. This leads to a Monte Carlo algorithm which automatically infers 3D scene structure from the predictable geometry of known object categories.

Thesis Supervisors: William T. Freeman and Alan S. Willsky
Professors of Electrical Engineering and Computer Science

Acknowledgments

Optical illusion is optical truth.
Johann Wolfgang von Goethe

There are three kinds of lies:
lies, damned lies, and statistics.
Attributed to Benjamin Disraeli by Mark Twain

This thesis would not have been possible without the encouragement, insight, and guidance of two advisors. I joined Professor Alan Willsky's research group during my first semester at MIT, and have appreciated his seemingly limitless supply of clever, and often unexpected, ideas ever since. Several passages of this thesis were greatly improved by his thorough revisions. Professor William Freeman arrived at MIT as I was looking for doctoral research topics, and played an integral role in articulating the computer vision tasks addressed by this thesis. On several occasions, his insight led to clear, simple reformulations of problems which avoided previous technical complications.

The research described in this thesis has immeasurably benefitted from several collaborators. Alex Ihler and I had the original idea for nonparametric belief propagation at perhaps the most productive party I've ever attended. He remains a good friend, despite having drafted me to help with lab system administration. I later recruited Michael Mandel from the MIT Jazz Ensemble to help with the hand tracking application; fortunately, his coding proved as skilled as his saxophone solos. More recently, I discovered that Antonio Torralba's insight for visual processing is matched only by his keen sense of humor. He deserves much of the credit for the central role that integrated models of visual scenes play in later chapters.

MIT has provided a very supportive environment for my doctoral research. I am particularly grateful to Prof. G. David Forney, Jr., who invited me to a 2001 Trieste workshop on connections between statistical physics, error correcting codes, and the graphical models which play a central role in this thesis. Later that summer, I had a very productive internship with Dr. Jonathan Yedidia at Mitsubishi Electric Research Labs, where I further explored these connections. My thesis committee, Profs. Tommi Jaakkola and Josh Tenenbaum, also provided thoughtful suggestions which continue to guide my research. The object recognition models developed in later sections were particularly influenced by Josh's excellent course on computational cognitive science.

One of the benefits of having two advisors has been interacting with two exciting research groups. I'd especially like to thank my long–time officemates Martin Wain-

wright, Alex Ihler, Junmo Kim, and Walter Sun for countless interesting conversations, and apologize to new arrivals Venkat Chandrasekaran and Myung Jin Choi for my recent single–minded focus on this thesis. Over the years, many other members of the Stochastic Systems Group have provided helpful suggestions during and after our weekly grouplet meetings. In addition, by far the best part of our 2004 move to the Stata Center has been interactions, and distractions, with members of CSAIL. After seven years at MIT, however, adequately thanking all of these individuals is too daunting a task to attempt here.

The successes I have had in my many, many years as a student are in large part due to the love and encouragement of my family. I cannot thank my parents enough for giving me the opportunity to freely pursue my interests, academic and otherwise. Finally, as I did four years ago, I thank my wife Erika for ensuring that my life is never entirely consumed by research. She has been astoundingly helpful, understanding, and patient over the past few months; I hope to repay the favor soon.

Contents

List of Figures

List of Algorithms

Chapter 1

Introduction

IMAGES and video can provide richly detailed summaries of complex, dynamic environments. Using computer vision systems, we may then automatically detect and recognize objects, track their motion, or infer three–dimensional (3D) scene geometry. Due to the wide availability of digital cameras, these methods are used in a huge range of applications, including human–computer interfaces, robot navigation, medical diagnosis, visual effects, multimedia retrieval, and remote sensing [91].

To see why these vision tasks are challenging, consider an environment in which a robot must interact with pedestrians. Although the robot will (hopefully) have some model of human form and behavior, it will undoubtedly encounter people that it has never seen before. These individuals may have widely varying clothing styles and physiques, and may move in sudden and unexpected ways. These issues are not limited to humans; even mundane objects such as chairs and automobiles vary widely in visual appearance. Realistic scenes are further complicated by partial occlusions, 3D object pose variations, and illumination effects.

Due to these difficulties, it is typically impossible to directly identify an isolated patch of pixels extracted from a natural image. Machine vision systems must thus propagate information from local features to create globally consistent scene interpretations. Statistical methods are widely used to characterize this local uncertainty, and learn robust object appearance models. In particular, *graphical models* provide a powerful framework for specifying precise, modular descriptions of computer vision tasks. Inference algorithms must then be tailored to the high–dimensional, continuous variables and complex distributions which characterize visual scenes. In many applications, physical description of scene variations is difficult, and these statistical models are instead learned from sparsely labeled training images.

This thesis considers two challenging computer vision applications which explore complementary aspects of the scene understanding problem. We first describe a kinematic model, and corresponding Monte Carlo methods, which may be used to track 3D hand motion from video sequences. We then consider less constrained environments, and develop hierarchical models relating objects, the parts composing them, and the scenes surrounding them. Both applications integrate nonparametric statistical methods with graphical models, and thus build algorithms which flexibly adapt to complex variations in object appearance.

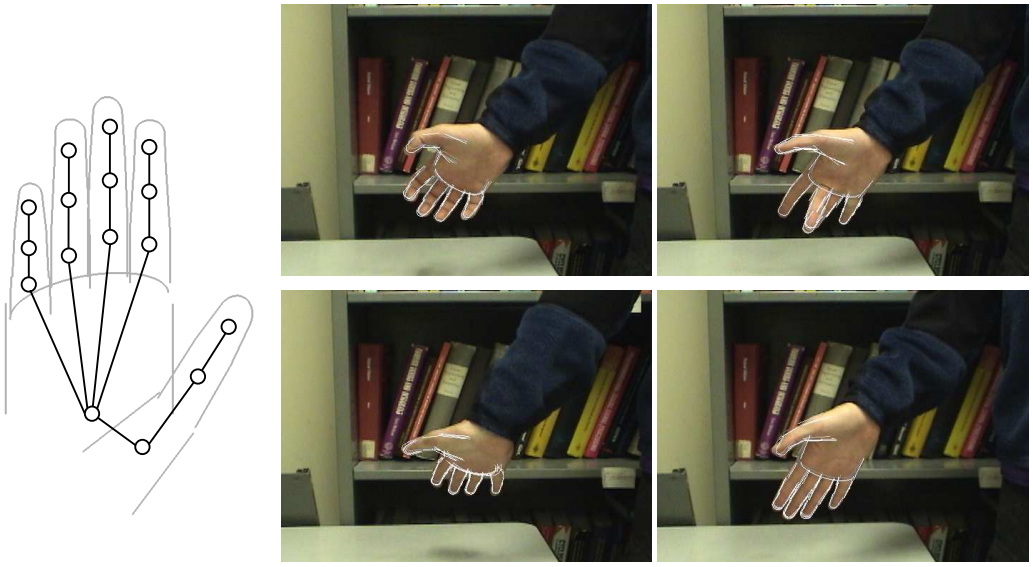


Figure 1.1. Visual tracking of articulated hand motion. *Left:* Representation of the hand as a collection of sixteen rigid bodies (nodes) connected by revolute joints (edges). *Right:* Four frames from a hand motion sequence. White edges correspond to projections of 3D hand pose estimates.

■ 1.1 Visual Tracking of Articulated Objects

Visual tracking systems use video sequences to estimate object or camera motion. Some of the most challenging tracking applications involve *articulated* objects, whose jointed motion leads to complex pose variations. In particular, human motion capture is widely used in visual effects and scene understanding applications [103, 214]. Estimates of human, and especially hand, motion are also used to build more expressive computer interfaces [333]. As illustrated in Fig. 1.1, this thesis develops probabilistic methods for tracking 3D hand and finger motion from monocular image sequences.

Hand pose is typically described by the angles of the thumb and fingers' joints, relative to the wrist or palm. Even coarse models of the hand's geometry have 26 continuous degrees of freedom: each finger has four rotational degrees of freedom, while the palm may take any 3D position and orientation [333]. This high dimensionality makes brute force search over all possible 3D poses intractable. Because hand motion may be erratic and rapid, even at video frame rates, simple local search procedures are often ineffective. Although there are dependencies among the hand's joint angles, they have a complex structure which, except in special cases [334], is not well captured by simple global dimensionality reduction techniques [293].

Visual tracking problems are further complicated by the projections inherent in the imaging process. Videos of hand motion typically contain many frames exhibiting *self–occlusion*, in which some fingers partially obscure other parts of the hand. These situations make it difficult to locally match hand parts to image features, since the

global hand pose determines which local edge and color cues should be expected for each finger. Furthermore, because the appearance of different fingers is typically very similar, accurate association of hand components to image cues is only possible through global geometric reasoning.

In some applications, 3D hand position must be identified from a single image. Several authors have posed this as a classification problem, where classes correspond to some discretization of allowable hand configurations [12, 256]. An image of the hand is precomputed for each class, and efficient algorithms for high–dimensional nearest neighbor search are used to find the closest 3D pose. These methods are most appropriate in applications such as sign language recognition, where only a small set of poses is of interest. When general hand motion is considered, the database of precomputed pose images may grow unacceptably large. A recently proposed method for interpolating between classes [295] makes no use of the image data during the interpolation, and thus makes the restrictive assumption that the transition between any pair of hand pose classes is highly predictable.

When video sequences are available, hand dynamics provide an important cue for tracking algorithms. Due to the hand's many degrees of freedom and nonlinearities in the imaging process, exact representation of the posterior distribution over model configurations is intractable. Trackers based on extended and unscented Kalman filters [204, 240, 270] have difficulties with the multimodal uncertainties produced by ambiguous image evidence. This has motivated many researchers to consider nonparametric representations, including particle filters [190, 334] and deterministic multiscale discretizations [271, 293]. However, the hand's high dimensionality can cause these trackers to suffer catastrophic failures, requiring the use of constraints which severely limit the hand's motion [190] or restrictive prior models of hand configurations and dynamics [293, 334].

Instead of reducing dimensionality by considering only a limited set of hand motions, we propose a graphical model describing the statistical structure underlying the hand's kinematics and imaging. Graphical models have been used to track view–based human body representations [236], contour models of restricted hand configurations [48] and simple object boundaries [47], view–based 2.5D "cardboard" models of hands and people [332], and a full 3D kinematic human body model [261, 262]. As shown in Fig. 1.1, nodes of our graphical model correspond to rigid hand components, which we individually parameterize by their 3D pose. Via a distributed representation of the hand's structure, kinematics, and dynamics, we then track hand motion without explicitly searching the space of global hand configurations.

■ 1.2 Object Categorization and Scene Understanding

Object recognition systems use image features to localize and categorize objects. We focus on the so–called *basic level* recognition of visually identifiable categories, rather than the differentiation of object instances. For example, in street scenes like those

Figure 1.2. Partial segmentations of street scenes highlighting four different object categories: cars (red), buildings (magenta), roads (blue), and trees (green).

shown in Fig. 1.2, we seek models which correctly classify previously unseen buildings and automobiles. While such basic level categorization is natural for humans [182, 228], it has proven far more challenging for computer vision systems. In particular, it is often difficult to manually define physical models which adequately capture the wide range of potential object shapes and appearance. We thus develop statistical methods which learn object appearance models from labeled training examples.

Most existing methods for object categorization use 2D, image–based appearance models. While pixel–level object segmentations are sometimes adequate, many applications require more explicit knowledge about the 3D world. For example, if robots are to navigate in complex environments and manipulate objects, they require more than a flat segmentation of the image pixels into object categories. Motivated by these challenges, our most sophisticated scene models cast object recognition as a 3D problem, leading to algorithms which partition estimated 3D structure into object categories.

■ 1.2.1 Recognition of Isolated Objects

We begin by considering methods which recognize cropped images depicting individual objects. Such images are frequently used to train computer vision algorithms [78, 304], and also arise in systems which use motion or saliency cues to focus attention [315]. Many different recognition algorithms may then be designed by coupling standard machine learning methods with an appropriate set of image features [91]. In some cases, simple pixel or wavelet–based features are selected via discriminative learning techniques [3, 304]. Other approaches combine sophisticated edge–based distance metrics with nearest neighbor classifiers [18, 20]. More recently, several recognition systems have employed interest regions which are affinely adapted to locally correct for 3D object pose variations [54, 81, 181, 266]. Sec. 5.1 describes these *affine covariant* regions [206, 207] in more detail.

Many of these recognition algorithms use *parts* to characterize the internal structure of objects, identifying spatially localized modules with distinctive visual appearances. Part–based object representations play a significant role in human perception [228], and also have a long history in computer vision [195]. For example, *pictorial structures* couple template–based part appearance models with spring–like spatial constraints [89]. More recent work provides statistical methods for learning pictorial structures, and computationally efficient algorithms for detecting object instances in test images [80]. *Constellation models* provide a closely related framework for part–based appearance modeling, in which parts characterize the expected location and appearance of discrete interest points [77, 82, 318].

In many cases, systems which recognize multiple objects are derived from independent models of each category. We believe that such systems should instead consider relationships among different object categories during the training process. This approach provides several benefits. At the lowest level, significant computational savings are possible if different categories share a common set of features. More importantly, jointly trained recognition systems can use similarities between object categories to their advantage by learning features which lead to better generalization [77, 299]. This transfer of knowledge is particularly important when few training examples are available, or when unsupervised discovery of new objects is desired.

■ 1.2.2 Multiple Object Scenes

In most computer vision applications, systems must detect and recognize objects in cluttered *visual scenes*. Natural environments like the street scenes of Fig. 1.2 often exhibit huge variations in object appearance, pose, and identity. There are two common approaches to adapting isolated object classifiers to visual scenes [3]. The "sliding window" method considers rectangular blocks of pixels at some discretized set of image positions and scales. Each of these windows is independently classified, and heuristics are then used to avoid multiple partially overlapping detections. An alternative "greedy" approach begins by finding the single most likely instance of each object category. The pixels or features corresponding to this instance are then removed, and subsequent hypotheses considered until no likely object instances remain.

Although they constrain each image region to be associated with a single object, these recognition frameworks otherwise treat different categories independently. In complex scenes, however, contextual knowledge may significantly improve recognition performance. At the coarsest level, the overall spatial structure, or *gist*, of an image provides priming information about likely object categories, and their most probable locations within the scene [217, 298]. Models of spatial relationships between objects can also improve detection of categories which are small or visually indistinct [7, 88, 126, 300, 301]. Finally, contextual models may better exploit partially labeled training databases, in which only some object instances have been manually identified.

Motivated by these issues, this thesis develops integrated, hierarchical models for multiple object scenes. The principal challenge in developing such models is specifying

tractable, scalable methods for handling uncertainty in the number of objects. *Grammars*, and related rule–based systems, provide one flexible family of hierarchical representations [27, 292]. For example, several different models impose distributions on multiscale, tree–based segmentations of the pixels composing simple scenes [2, 139, 265, 274]. In addition, an *image parsing* [301] framework has been proposed which explains an image using a set of regions generated by generic or object–specific processes. While this model allows uncertainty in the number of regions, and hence objects, its use of high–dimensional latent variables require good, discriminatively trained proposal distributions for acceptable MCMC performance. The BLOG language [208] provides another promising method for reasoning about unknown objects, although the computational tools needed to apply BLOG to large–scale applications are not yet available. In later sections, we propose a different framework for handling uncertainty in the number of object instances, which adapts nonparametric statistical methods.

■ 1.3 Overview of Methods and Contributions

This thesis proposes novel methods for visually tracking articulated objects, and detecting object categories in natural scenes. We now survey the statistical methods which we use to learn robust appearance models, and efficiently infer object identity and pose.

■ 1.3.1 Particle–Based Inference in Graphical Models

Graphical models provide a powerful, general framework for developing statistical models of computer vision problems [95, 98, 108, 159]. However, graphical formulations are only useful when combined with efficient learning and inference algorithms. Computer vision problems, like the articulated tracking task introduced in Sec. 1.1, are particularly challenging because they involve high–dimensional, continuous variables and complex, multimodal distributions. Realistic graphical models for such problems must represent outliers, bimodalities, and other non–Gaussian statistical features. The corresponding optimal inference procedures for these models typically involve integral equations for which no closed form solution exists. It is thus necessary to develop families of approximate representations, and corresponding computational methods.

The simplest approximations of intractable, continuous–valued graphical models are based on discretization. Although exact inference in general discrete graphs is NP hard, approximate inference algorithms such as loopy belief propagation (BP) [231, 306, 339] often produce excellent empirical results. Certain vision problems, such as dense stereo reconstruction [17, 283], are well suited to discrete formulations. For problems involving high–dimensional variables, however, exhaustive discretization of the state space is intractable. In some cases, domain–specific heuristics may be used to dynamically exclude those configurations which appear unlikely based upon the local evidence [48, 95]. In more challenging applications, however, the local evidence at some nodes may be inaccurate or misleading, and these approximations lead to distorted estimates.

For temporal inference problems, particle filters [11, 70, 72, 183] have proven to be

an effective, and influential, alternative to discretization. They provide the basis for several of the most effective visual tracking algorithms [190, 260]. Particle filters approximate conditional densities nonparametrically as a collection of representative elements. Monte Carlo methods are then used to propagate these weighted particles as the temporal process evolves, and consistently revise estimates given new observations.

Although particle filters are often effective, they are specialized to temporal problems whose corresponding graphs are simple Markov chains. Many vision applications, however, are characterized by more complex spatial or model–induced structure. Motivated by these difficulties, we propose a *nonparametric belief propagation* (NBP) algorithm which allows particle–based inference in arbitrary graphs. NBP approximates complex, continuous sufficient statistics by kernel–based density estimates. Efficient, multiscale Gibbs sampling algorithms are then used to fuse the information provided by several messages, and propagate particles throughout the graph. As several computational examples demonstrate, the NBP algorithm may be applied to arbitrarily structured graphs containing a broad range of complex, non–linear potential functions.

■ 1.3.2 Graphical Representations for Articulated Tracking

As discussed in Sec. 1.1, articulated tracking problems are complicated by the high dimensionality of the space of possible object poses. In fact, however, the kinematic and dynamic behavior of objects like hands exhibits significant structure. To exploit this, we consider a redundant *local* representation in which each hand component is described by its 3D position and orientation. Kinematic constraints, including self–intersection constraints not captured by joint angle representations, are then naturally described by a graphical model. By introducing a set of auxiliary occlusion masks, we may also decompose color and edge–based image likelihoods to provide direct evidence for the pose of individual fingers.

Because the pose of each hand component is described by a six–dimensional continuous variable, discretized state representations are intractable. We instead apply the NBP algorithm, and thus develop a tracker which propagates local pose estimates to infer global hand motion. The resulting algorithm updates particle–based estimates of finger position and orientation via likelihood functions which consistently discount occluded image regions.

■ 1.3.3 Hierarchical Models for Scenes, Objects, and Parts

The second half of this thesis considers the object recognition and scene understanding applications introduced in Sec. 1.2. In particular, we develop a family of hierarchical generative models for objects, the parts composing them, and the scenes surrounding them. Our models share information between object categories in three distinct ways. First, parts define distributions over a common low–level feature vocabularly, leading to computational savings when analyzing new images. In addition, and more unusually, objects are defined using a common set of parts. This structure leads to the discovery of parts with interesting semantic interpretations, and can improve performance when

few training examples are available. Finally, object appearance information is shared between the many scenes in which that object is found.

This generative approach is motivated by the pragmatic need for learning algorithms which require little manual supervision and labeling. While discriminative models often produce accurate classifiers, they typically require very large training sets even for relatively simple categories [304]. In contrast, generative approaches can discover large, visually salient categories (such as foliage and buildings [266]) without supervision. Partial segmentations can then be used to learn semantically interesting categories (such as cars and pedestrians) which are less visually distinctive, or present in fewer training images. Moreover, by employing a single hierarchy describing multiple objects or scenes, the learning process automatically shares information between categories.

In the simplest case, our scene models assemble 2D objects in a "jigsaw puzzle" fashion. To allow scale–invariant object recognition, we generalize these models to describe the 3D structure and appearance of object categories. Binocular stereo training images are used to approximately calibrate these geometric models. Because we consider objects with predictable 3D structure, we may then automatically recover a coarse reconstruction of the scene depths underlying test images.

■ 1.3.4 Visual Learning via Transformed Dirichlet Processes

Our hierarchical models are adapted from *topic models* originally proposed for the analysis of text documents [31, 289]. These models make the so–called *bag of words* assumption, in which raw documents are converted to word counts, and sentence structure is ignored. While it is possible to develop corresponding *bag of features* models for images [14, 54, 79, 266], which model the appearance of detected interest points and ignore their location, we show that doing so neglects valuable information, and reduces recognition performance. To consistently account for spatial structure, we augment these hierarchies with *transformation* [97, 156, 210] variables describing the location of each object in each training image. Through these transformations, we learn parts which describe features relative to a "canonical" coordinate frame, without requiring alignment of the training or test images.

To better learn robust, data–driven models which require few manually specified parameters, we employ the *Dirichlet process* (DP) [28, 83, 254]. In nonparametric Bayesian statistics, DPs are commonly used to learn mixture models whose number of components is not fixed, but instead inferred from data [10, 76, 222]. A *hierarchical Dirichlet process* (HDP) [288, 289] models multiple related datasets by reusing a common set of mixture components in different proportions. We extend the HDP framework by allowing the global, shared mixture components to undergo a random set of transformations. The resulting *transformed Dirichlet process* (TDP) produces models which automatically learn the number of parts underlying each object category, and the number of object instances composing each scene. Our use of continuous transformation variables then leads to efficient, Rao–Blackwellized Gibbs samplers which jointly recognize objects and infer 3D scene structure.

1.4 Thesis Organization

We now provide an overview of the methods and results which are considered by subsequent thesis chapters. The introductory paragraphs of each chapter provide more detailed outlines.

Chapter 2: Nonparametric and Graphical Models

We begin by reviewing a broad range of statistical methods upon which the models in this thesis are based. This chapter first describes exponential families of probability distributions, and provides detailed computational methods for two families (the Dirichlet–multinomial and normal–inverse–Wishart) used extensively in later chapters. We then provide an introduction to graphical models, emphasizing the statistical assumptions underlying these structured representations. Turning to computational issues, we discuss several different variational methods, including the belief propagation and expectation maximization algorithms. We also discuss Monte Carlo methods, which provide complementary families of learning and inference algorithms. The chapter concludes with an introduction to the Dirichlet process, which is widely used in nonparametric Bayesian statistics. We survey the statistical theory underlying these robust methods, before discussing learning algorithms and hierarchical extensions.

Chapter 3: Nonparametric Belief Propagation

In this chapter, we develop an approximate inference algorithm for graphical models describing continuous, non–Gaussian random variables. We begin by reviewing particle filters, which track complex temporal processes via sample–based density estimates. We then propose a nonparametric belief propagation (NBP) algorithm which extends the Monte Carlo methods underlying particle filters to general graphical models. For simplicity, we first describe the NBP algorithm for graphs whose potentials are Gaussian mixtures. Via importance sampling methods, we then adapt NBP to graphs defined by a very broad range of analytic potentials. NBP fuses information from different parts of the graph by sampling from products of Gaussian mixtures. Using multiscale, KD–tree density representations, we provide several efficient computational methods for these updates. We conclude by validating NBP's performance in simple Gaussian graphical models, and a part–based model which describes the appearance of facial features.

Chapter 4: Visual Hand Tracking

The fourth chapter applies the NBP algorithm to visually track articulated hand motion. We begin with a detailed examination of the kinematic and structural constraints underlying hand motion. Via a local representation of hand components in terms of their 3D pose, we construct a graphical model exposing internal hand structure. Using a set of binary auxiliary variables specifying the occlusion state of each pixel, we also locally factorize color and edge–based likelihoods. Applying NBP to this model, we derive a particle–based hand tracking algorithm, in which quaternions are used to

consistently estimate finger orientation. Via an efficient analytic approximation, we may also marginalize occlusion masks, and thus infer occlusion events in a distributed fashion. Simulations then demonstrate that NBP effectively refines coarse initial pose estimates, and tracks hand motion in extended video sequences.

Chapter 5: Object Categorization using Shared Parts

The second half of this thesis focuses on methods for robustly learning object appearance models. This chapter begins by describing the set of sparse, affinely adapted image features underlying our recognition system. We then propose general families of spatial transformations which allow consistent models of object and scene structure. Considering images of isolated objects, we first develop a parametric, fixed–order model which uses shared parts to describe multiple object categories. Monte Carlo methods are used to learn this model's parameters from training images. We then adapt Dirichlet processes to this recognition task, and thus learn an appropriate number of shared parts automatically. Empirical results on a dataset containing sixteen object categories demonstrate the benefits of sharing parts, and the advantages of learning algorithms derived from nonparametric models.

Chapter 6: Scene Understanding via Transformed Dirichlet Processes

In this chapter, we generalize our hierarchical object appearance models to more complex visual scenes. We first develop a parametric model which describes objects via a common set of shared parts, and contextual relationships among the positions at which a fixed set of objects is observed. To allow uncertainty in the number of object instances underlying each image, we then propose a framework which couples Dirichlet processes with spatial transformations. Applying the resulting transformed Dirichlet process, we develop Monte Carlo methods which robustly learn part–based models of an unknown set of visual categories. We also extend this model to describe 3D scene structure, and thus reconstruct feature depths via the predictable geometry of object categories. These scene models are tested on datasets depicting complex street and office environments.

Chapter 7: Contributions and Recommendations

We conclude by surveying the contributions of this thesis, and outline directions for future research. Many of these ideas combine aspects of our articulated object tracking and scene understanding frameworks, which have complementary strengths.

Chapter 2

Nonparametric and Graphical Models

STATISTICAL methods play a central role in the design and analysis of machine vision systems. In this background chapter, we review several learning and inference techniques upon which our later contributions are based. We begin in Sec. 2.1 by describing exponential families of probability densities, emphasizing the roles of sufficiency and conjugacy in Bayesian learning. Sec. 2.2 then shows how graphs may be used to impose structure on exponential families. We contrast several types of graphical models, and provide results clarifying their underlying statistical assumptions.

To apply graphical models in practical applications, computationally efficient learning and inference algorithms are needed. Sec. 2.3 describes several variational methods which approximate intractable inference tasks via message–passing algorithms. In Sec. 2.4, we discuss a complementary class of Monte Carlo methods which use stochastic simulations to analyze complex models. In this thesis, we propose new inference algorithms which integrate variational and Monte Carlo methods in novel ways.

Finally, we conclude in Sec. 2.5 with an introduction to nonparametric methods for Bayesian learning. These infinite–dimensional models achieve greater robustness by avoiding restrictive assumptions about the data generation process. Despite this flexibility, variational and Monte Carlo methods can be adapted to allow tractable analysis of large, high–dimensional datasets.

■ 2.1 Exponential Families

An *exponential family* of probability distributions [15, 36, 311] is characterized by the values of certain *sufficient statistics*. Let x be a random variable taking values in some sample space $\mathcal{X}$, which may be either continuous or discrete. Given a set of statistics or *potentials* $\{\phi_a \mid a \in \mathcal{A}\}$, the corresponding exponential family of densities is given by

$$p(x \mid \theta) = \nu(x) \exp\left\{ \sum_{a \in \mathcal{A}} \theta_a \phi_a(x) - \Phi(\theta) \right\} \tag{2.1}$$

where $\theta \in \mathbb{R}^{|\mathcal{A}|}$ are the family's *natural* or *canonical* parameters, and $\nu(x)$ is a non-negative *reference measure*. In some applications, the parameters θ are set to fixed constants, while in other cases they are interpreted as latent random variables. The *log partition function* $\Phi(\theta)$ is defined to normalize $p(x \mid \theta)$ so that it integrates to one:

$$\Phi(\theta) = \log \int_{\mathcal{X}} \nu(x) \exp\left\{ \sum_{a \in \mathcal{A}} \theta_a \phi_a(x) \right\} dx \tag{2.2}$$

For discrete spaces, dx is taken to be counting measure, so that integrals become summations. This construction is valid when the canonical parameters θ belong to the set Θ for which the log partition function is finite:

$$\Theta \triangleq \left\{ \theta \in \mathbb{R}^{|\mathcal{A}|} \mid \Phi(\theta) < \infty \right\} \tag{2.3}$$

Because $\Phi(\theta)$ is a convex function (see Prop. 2.1.1), Θ is necessarily convex. If Θ is also open, the exponential family is said to be *regular*. Many classic probability distributions form regular exponential families, including the Bernoulli, Poisson, Gaussian, beta, and gamma densities [21, 107]. For example, for scalar Gaussian densities the sufficient statistics are $\{x, x^2\}$, $\nu(x) = 1$, and Θ constrains the variance to be positive.

Exponential families are typically parameterized so that no linear combination of the potentials $\{\phi_a \mid a \in \mathcal{A}\}$ is almost everywhere constant. In such a *minimal* representation,[1] there is a unique set of canonical parameters θ associated with each density in the family, whose *dimension* equals $d \triangleq |\mathcal{A}|$. Furthermore, the exponential family defines a d–dimensional Riemannian manifold, and the canonical parameters a coordinate system for that manifold. By characterizing the convex geometric structure of such manifolds, *information geometry* [6, 15, 52, 74, 305] provides a powerful framework for analyzing learning and inference algorithms. In particular, as we discuss in Sec. 2.3, results from *conjugate duality* [15, 311] underlie many algorithms used in this thesis.

In the following sections, we further explore the properties of exponential families, emphasizing results which guide the specification of sufficient statistics appropriate to particular learning problems. We then introduce a family of conjugate priors for the canonical parameters θ, and provide detailed computational methods for two exponential families (the normal–inverse–Wishart and Dirichlet–multinomial) used extensively in this thesis. For further discussion of the convex geometry underlying exponential families, see [6, 15, 36, 74, 311].

■ 2.1.1 Sufficient Statistics and Information Theory

In this section, we establish several results which motivate the use of exponential families, and clarify the notion of sufficiency. The following properties of the log partition function establish its central role in the study of exponential families:

[1] We note, however, that *overcomplete* representations play an important role in recent theoretical analyses of variational approaches to approximate inference [305, 306, 311].

Proposition 2.1.1. *The log partition function* $\Phi(\theta)$ *of eq.* (2.2) *is convex (strictly so for minimal representations) and continuously differentiable over its domain* Θ. *Its derivatives are the* cumulants *of the sufficient statistics* $\{\phi_a \mid a \in \mathcal{A}\}$, *so that*

$$\frac{\partial \Phi(\theta)}{\partial \theta_a} = \mathbb{E}_\theta[\phi_a(x)] \triangleq \int_{\mathcal{X}} \phi_a(x)\, p(x \mid \theta)\, dx \tag{2.4}$$

$$\frac{\partial^2 \Phi(\theta)}{\partial \theta_a \partial \theta_b} = \mathbb{E}_\theta[\phi_a(x)\, \phi_b(x)] - \mathbb{E}_\theta[\phi_a(x)]\, \mathbb{E}_\theta[\phi_b(x)] \tag{2.5}$$

Proof. For a detailed proof of this classic result, see [15, 36, 311]. The cumulant generating properties follow from the chain rule and algebraic manipulation. From eq. (2.5), $\nabla^2 \Phi(\theta)$ is a positive semi–definite covariance matrix, implying convexity of $\Phi(\theta)$. For minimal families, $\nabla^2 \Phi(\theta)$ must be positive definite, guaranteeing strict convexity. □

Due to this result, the log partition function is also known as the *cumulant generating function* of the exponential family. The convexity of $\Phi(\theta)$ has important implications for the geometry of exponential families [6, 15, 36, 74].

Entropy, Information, and Divergence

Concepts from information theory play a central role in the study of learning and inference in exponential families. Given a probability distribution $p(x)$ defined on a discrete space $\mathcal{X}$, Shannon's measure of *entropy* (in natural units, or *nats*) equals

$$H(p) = -\sum_{x \in \mathcal{X}} p(x) \log p(x) \tag{2.6}$$

In such diverse fields as communications, signal processing, and statistical physics, entropy arises as a natural measure of the inherent uncertainty in a random variable [49]. The *differential entropy* extends this definition to continuous spaces:

$$H(p) = -\int_{\mathcal{X}} p(x) \log p(x)\, dx \tag{2.7}$$

In both discrete and continuous domains, the (differential) entropy $H(p)$ is concave, continuous, and maximal for uniform densities. However, while the discrete entropy is guaranteed to be non-negative, differential entropy is sometimes less than zero.

For problems of model selection and approximation, we need a measure of the distance between probability distributions. The *relative entropy* or *Kullback-Leibler (KL) divergence* between two probability distributions $p(x)$ and $q(x)$ equals

$$D(p \,||\, q) = \int_{\mathcal{X}} p(x) \log \frac{p(x)}{q(x)}\, dx \tag{2.8}$$

Important properties of the KL divergence follow from *Jensen's inequality* [49], which bounds the expectation of convex functions:

$$\mathbb{E}[f(x)] \geq f(\mathbb{E}[x]) \qquad \text{for any convex } f : \mathcal{X} \to \mathbb{R} \tag{2.9}$$

Applying Jensen's inequality to the logarithm of eq. (2.8), which is concave, it is easily shown that the KL divergence $D(p\,||\,q) \geq 0$, with $D(p\,||\,q) = 0$ if and only if $p(x) = q(x)$ almost everywhere. However, it is not a true distance metric because $D(p\,||\,q) \neq D(q\,||\,p)$. Given a target density $p(x)$ and an approximation $q(x)$, $D(p\,||\,q)$ can be motivated as the information gain achievable by using $p(x)$ in place of $q(x)$ [49]. Interestingly, the alternate KL divergence $D(q\,||\,p)$ also plays an important role in the development of variational methods for approximate inference (see Sec. 2.3).

An important special case arises when we consider the dependency between two random variables x and y. Let $p_{xy}(x, y)$ denote their joint distribution, $p_x(x)$ and $p_y(y)$ their corresponding marginals, and $\mathcal{X}$ and $\mathcal{Y}$ their sample spaces. The *mutual information* between x and y then equals

$$I(p_{xy}) \triangleq D(p_{xy}\,||\,p_x p_y) = \int_{\mathcal{X}} \int_{\mathcal{Y}} p_{xy}(x, y) \log \frac{p_{xy}(x, y)}{p_x(x) p_y(y)}\, dy\, dx \tag{2.10}$$

$$= H(p_x) + H(p_y) - H(p_{xy}) \tag{2.11}$$

where eq. (2.11) follows from algebraic manipulation. The mutual information can be interpreted as the expected reduction in uncertainty about one random variable from observation of another [49].

Projections onto Exponential Families

In many cases, learning problems can be posed as a search for the best approximation of an empirically derived *target* density $\tilde{p}(x)$. As discussed in the previous section, the KL divergence $D(\tilde{p}\,||\,q)$ is a natural measure of the accuracy of an approximation $q(x)$. For exponential families, the optimal approximating density is elegantly characterized by the following *moment–matching* conditions:

Proposition 2.1.2. *Let $\tilde{p}$ denote a target probability density, and p_θ an exponential family. The approximating density minimizing $D(\tilde{p}\,||\,p_\theta)$ then has canonical parameters $\hat{\theta}$ chosen to match the expected values of that family's sufficient statistics:*

$$\mathbb{E}_{\hat{\theta}}[\phi_a(x)] = \int_{\mathcal{X}} \phi_a(x)\, \tilde{p}(x)\, dx \qquad\qquad a \in \mathcal{A} \tag{2.12}$$

For minimal families, these optimal parameters $\hat{\theta}$ are uniquely determined.

Proof. From the definition of KL divergence (eq. (2.8)), we have

$$\begin{aligned} D(\tilde{p}\,||\,p_\theta) &= \int_{\mathcal{X}} \tilde{p}(x) \log \frac{\tilde{p}(x)}{p(x \mid \theta)}\, dx \\ &= \int_{\mathcal{X}} \tilde{p}(x) \log \tilde{p}(x)\, dx - \int_{\mathcal{X}} \tilde{p}(x) \left[\log \nu(x) + \sum_{a \in \mathcal{A}} \theta_a \phi_a(x) - \Phi(\theta) \right] dx \\ &= -H(\tilde{p}) - \int_{\mathcal{X}} \tilde{p}(x) \log \nu(x)\; dx - \sum_{a \in \mathcal{A}} \theta_a \int_{\mathcal{X}} \phi_a(x)\, \tilde{p}(x)\, dx + \Phi(\theta) \end{aligned}$$

Taking derivatives with respect to θ_a and setting $\partial D(\tilde{p}\,||\,p_\theta)\,/\partial\theta_a = 0$, we then have

$$\frac{\partial\Phi(\theta)}{\partial\theta_a} = \int_{\mathcal{X}} \phi_a(x)\,\tilde{p}(x)\,dx \qquad\qquad a \in \mathcal{A}$$

Equation (2.12) follows from the cumulant generating properties of $\Phi(\theta)$ (eq. (2.4)). Because $\Phi(\theta)$ is strictly convex for minimal families (Prop. 2.1.1), the canonical parameters $\hat{\theta}$ satisfying eq. (2.12) achieve the unique global minimum of $D(\tilde{p}\,||\,p_\theta)$. □

In information geometry, the density satisfying eq. (2.12) is known as the *I–projection* of $\tilde{p}(x)$ onto the *e–flat manifold* defined by the exponential family's canonical parameters [6, 52]. Note that the optimal projection depends only the potential functions' expected values under $\tilde{p}(x)$, so that these statistics are *sufficient* to determine the closest approximation.

In many applications, rather than an explicit target density $\tilde{p}(x)$, we instead observe L independent samples $\{x^{(\ell)}\}_{\ell=1}^{L}$ from that density. In this situation, we define the *empirical density* of the samples as follows:

$$\tilde{p}(x) = \frac{1}{L}\sum_{\ell=1}^{L} \delta\big(x, x^{(\ell)}\big) \tag{2.13}$$

Here, $\delta\big(x, x^{(\ell)}\big)$ is the Dirac delta function for continuous $\mathcal{X}$, and the Kronecker delta for discrete $\mathcal{X}$. Specializing Prop. 2.1.2 to this case, we find a correspondence between information projection and *maximum likelihood (ML)* parameter estimation.

Proposition 2.1.3. *Let p_θ denote an exponential family with canonical parameters θ. Given L independent, identically distributed samples $\{x^{(\ell)}\}_{\ell=1}^{L}$, with empirical density $\tilde{p}(x)$ as in eq. (2.13), the maximum likelihood estimate $\hat{\theta}$ of the canonical parameters coincides with the empirical density's information projection:*

$$\hat{\theta} = \arg\max_{\theta} \sum_{\ell=1}^{L} \log p(x^{(\ell)} \mid \theta) = \arg\min_{\theta} D(\tilde{p}\,||\,p_\theta) \tag{2.14}$$

These optimal parameters are uniquely determined for minimal families, and characterized by the following moment matching conditions:

$$\mathbb{E}_{\hat{\theta}}[\phi_a(x)] = \frac{1}{L}\sum_{\ell=1}^{L} \phi_a(x^{(\ell)}) \qquad\qquad a \in \mathcal{A} \tag{2.15}$$

Proof. Expanding the KL divergence from $\tilde{p}(x)$ (eq. (2.13)), we have

$$\begin{aligned} D(\tilde{p}\,||\,p_\theta) &= \int_{\mathcal{X}} \tilde{p}(x) \log \tilde{p}(x)\, dx - \int_{\mathcal{X}} \tilde{p}(x) \log p(x \mid \theta)\; dx \\ &= -H(\tilde{p}) - \int_{\mathcal{X}} \frac{1}{L} \sum_{\ell=1}^{L} \delta\big(x, x^{(\ell)}\big) \log p(x \mid \theta)\; dx \\ &= -H(\tilde{p}) - \frac{1}{L} \sum_{\ell=1}^{L} \log p(x^{(\ell)} \mid \theta) \end{aligned}$$

Because $H(\tilde{p})$ does not depend on θ, the parameters minimizing $D(\tilde{p}\,||\,p_\theta)$ and maximizing the expected log–likelihood coincide, establishing eq. (2.14). The unique characterization of $\hat{\theta}$ via moment–matching (eq. (2.15)) then follows from Prop. 2.1.2. □

In principle, Prop. 2.1.2 and 2.1.3 suggest a straightforward procedure for learning exponential familes: estimate appropriate sufficient statistics, and then find corresponding canonical parameters via convex optimization [6, 15, 36, 52]. In practice, however, significant difficulties may arise. For example, practical applications often require *semi-supervised learning* from partially labeled training data, so that the needed statistics cannot be directly measured. Even when sufficient statistics are available, calculation of the corresponding parameters can be intractable in large, complex models.

These results also have important implications for the selection of appropriate exponential families. In particular, because the chosen statistics are sufficient for parameter estimation, the learned model *cannot* capture aspects of the target distribution neglected by these statistics. These concerns motivate our later development of *nonparametric* methods (see Sec. 2.5) which extend exponential families to learn richer, more flexible models.

Maximum Entropy Models

In the previous section, we argued that certain statistics are sufficient to characterize the best exponential family approximation of a given target density. The following theorem shows that if these statistics are the *only* available information about a target density, then the corresponding exponential family provides a natural model.

Theorem 2.1.1. *Consider a collection of statistics $\{\phi_a \mid a \in \mathcal{A}\}$, whose expectations with respect to some target density $\tilde{p}(x)$ are known:*

$$\int_{\mathcal{X}} \phi_a(x)\, \tilde{p}(x)\, dx = \mu_a \qquad\qquad a \in \mathcal{A} \tag{2.16}$$

The unique distribution $\hat{p}(x)$ maximizing the entropy $H(\hat{p})$, subject to these moment constraints, is then a member of the exponential family of eq. (2.1), with $\nu(x) = 1$ and canonical parameters $\hat{\theta}$ chosen so that $\mathbb{E}_{\hat{\theta}}[\phi_a(x)] = \mu_a$.

Proof. The general form of eq. (2.1) can be motivated by a Lagrangian formulation of this constrained optimization problem. Taking derivatives, the Lagrange multipliers become the exponential family's canonical parameters. Global optimality can then be verified via a bound based on the KL divergence [21, 49]. A related characterization of exponential families with reference measures $\nu(x) \neq 1$ is also possible [21]. □

Note that eq. (2.16) implicitly assumes the existence of *some* distribution satisfying the specified moment constraints. In general, verifying this feasibility can be extremely challenging [311], relating to classic moment inequality [25, 176] and covariance extension [92, 229] problems. Also, given insufficient moment constraints for non–compact continous spaces, the maximizing density may be improper and have infinite entropy.

Recall that the entropy measures the inherent uncertainty in a random variable. Thus, if the sufficient statistics of eq. (2.16) are the only available characterization of a target density, the corresponding exponential family is justified as the model which imposes the fewest additional assumptions about the data generation process.

■ 2.1.2 Learning with Prior Knowledge

The results of the previous sections show how exponential families use sufficient statistics to characterize the *likelihood* of observed training data. Frequently, however, we also have *prior* knowledge about the expected location, scale, concentration, or other features of the process generating the data. When learning from small datasets, consistent incorporation of prior knowledge can dramatically improve the accuracy and robustness of the resulting model.

In this section, we develop Bayesian methods for learning and inference which treat the "parameters" of exponential family densities as random variables. In addition to allowing easy incorporation of prior knowledge, this approach provides natural confidence estimates for models learned from noisy or sparse data. Furthermore, it leads to powerful methods for transferring knowledge among multiple related learning tasks. See Bernardo and Smith [21] for a more formal, comprehensive survey of this topic.

Analysis of Posterior Distributions

Given an exponential family $p(x \mid \theta)$ with canonical parameters θ, Bayesian analysis begins with a *prior distribution* $p(\theta \mid \lambda)$ capturing any available knowledge about the data generation process. This prior distribution is typically itself a member of a family of densities with *hyperparameters* λ. For the moment, we assume these hyperparameters are set to some fixed value based on our prior beliefs.

Given L independent, identically distributed observations $\{x^{(\ell)}\}_{\ell=1}^{L}$, two computations arise frequently in statistical analyses. Using Bayes' rule, the *posterior distribution*

of the canonical parameters can be written as follows:

$$p(\theta \mid x^{(1)}, \ldots, x^{(L)}, \lambda) = \frac{p(x^{(1)}, \ldots, x^{(L)} \mid \theta, \lambda)\, p(\theta \mid \lambda)}{\int_\Theta p(x^{(1)}, \ldots, x^{(L)} \mid \theta, \lambda)\, p(\theta \mid \lambda)\; d\theta} \tag{2.17}$$

$$\propto p(\theta \mid \lambda) \prod_{\ell=1}^{L} p(x^{(\ell)} \mid \theta) \tag{2.18}$$

The proportionality symbol of eq. (2.18) represents the constant needed to ensure integration to unity (in this case, the data likelihood of eq. (2.17)). Recall that, for minimal exponential families, the canonical parameters are uniquely associated with expectations of that family's sufficient statistics (Prop. 2.1.3). The posterior distribution of eq. (2.18) thus captures our knowledge about the statistics likely to be exhibited by future observations.

In many situations, statistical models are used primarily to predict future observations. Given L independent observations as before, the *predictive likelihood* of a new observation $\bar{x}$ equals

$$p(\bar{x} \mid x^{(1)}, \ldots, x^{(L)}, \lambda) = \int_\Theta p(\bar{x} \mid \theta)\, p(\theta \mid x^{(1)}, \ldots, x^{(L)}, \lambda)\; d\theta \tag{2.19}$$

where the posterior distribution over parameters is as in eq. (2.18). By averaging over our posterior uncertainty in the parameters θ, this approach leads to predictions which are typically more robust than those based on a single parameter estimate.

In principle, a fully Bayesian analysis should also place a prior distribution $p(\lambda)$ on the hyperparameters. In practice, however, computational considerations frequently motivate an *empirical Bayesian* approach [21, 75, 107] in which λ is estimated by maximizing the training data's marginal likelihood:

$$\hat{\lambda} = \arg\max_\lambda\, p(x^{(1)}, \ldots, x^{(L)} \mid \lambda) \tag{2.20}$$

$$= \arg\max_\lambda \int_\Theta p(\theta \mid \lambda) \prod_{\ell=1}^{L} p(x^{(\ell)} \mid \theta)\; d\theta \tag{2.21}$$

In situations where this optimization is intractable, cross–validation approaches which optimize the predictive likelihood of a held–out data set are often useful [21].

More generally, the predictive likelihood computation of eq. (2.19) is itself intractable for many practical models. In these cases, the parameters' posterior distribution (eq. (2.18)) is often approximated by a single *maximum a posteriori (MAP)* estimate:

$$\hat{\theta} = \arg\max_\theta\, p(\theta \mid x^{(1)}, \ldots, x^{(L)}, \lambda) \tag{2.22}$$

$$= \arg\max_\theta\, p(\theta \mid \lambda) \prod_{\ell=1}^{L} p(x^{(\ell)} \mid \theta) \tag{2.23}$$

This approach is best justified when the training set size L is very large, so that the posterior distribution of eq. (2.22) is tightly concentrated [21, 107]. Sometimes, however, MAP estimates are used with smaller datasets because they are the only computationally viable option.

Parametric and Predictive Sufficiency

When computing the posterior distributions and predictive likelihoods motivated in the previous section, it is very helpful to have compact ways of characterizing large datasets. For exponential families, the notions of sufficiency introduced in Sec. 2.1.1 can be extended to simplify learning with prior knowledge.

Theorem 2.1.2. *Let $p(x \mid \theta)$ denote an exponential family with canonical parameters θ, and $p(\theta \mid \lambda)$ a corresponding prior density. Given L independent, identically distributed samples $\{x^{(\ell)}\}_{\ell=1}^{L}$, consider the following statistics:*

$$\boldsymbol{\phi}(x^{(1)}, \dots, x^{(L)}) \triangleq \left\{ \frac{1}{L} \sum_{\ell=1}^{L} \phi_a(x^{(\ell)}) \;\middle|\; a \in \mathcal{A} \right\} \tag{2.24}$$

These empirical moments, along with the sample size L, are then said to be parametric sufficient *for the posterior distribution over canonical parameters, so that*

$$p(\theta \mid x^{(1)}, \dots, x^{(L)}, \lambda) = p(\theta \mid \boldsymbol{\phi}(x^{(1)}, \dots, x^{(L)}), L, \lambda) \tag{2.25}$$

Equivalently, they are predictive sufficient *for the likelihood of new data $\bar{x}$:*

$$p(\bar{x} \mid x^{(1)}, \dots, x^{(L)}, \lambda) = p(\bar{x} \mid \boldsymbol{\phi}(x^{(1)}, \dots, x^{(L)}), L, \lambda) \tag{2.26}$$

Proof. Parametric sufficiency follows from the Neyman factorization criterion, which is satisfied by any exponential family. The correspondence between parametric and predictive sufficiency can then be argued from eqs. (2.18, 2.19). For details, see Sec. 4.5 of Bernardo and Smith [21]. □

This theorem makes exponential families particularly attractive when learning from large datasets, due to the often dramatic compression provided by the statistics of eq. (2.24). It also emphasizes the importance of selecting appropriate sufficient statistics, since other features of the data cannot affect subsequent model predictions.

Analysis with Conjugate Priors

Theorem 2.1.2 shows that statistical predictions in exponential families are functions solely of the chosen sufficient statistics. However, it does not provide an explicit characterization of the posterior distribution over model parameters, or guarantee that the predictive likelihood can be computed tractably. In this section, we describe an expressive family of prior distributions which are also analytically tractable.

Let $p(x \mid \theta)$ denote a family of probability densities parameterized by θ. A family of prior densities $p(\theta \mid \lambda)$ is said to be *conjugate* to $p(x \mid \theta)$ if, for any observation x and hyperparameters λ, the posterior distribution $p(\theta \mid x, \lambda)$ remains in that family:

$$p(\theta \mid x, \lambda) \propto p(x \mid \theta)\, p(\theta \mid \lambda) \propto p\big(\theta \mid \bar{\lambda}\big) \tag{2.27}$$

In this case, the posterior distribution is compactly described by an updated set of hyperparameters $\bar{\lambda}$. For exponential families parameterized as in eq. (2.1), conjugate priors [21, 36] take the following general form:

$$p(\theta \mid \lambda) = \exp\left\{\sum_{a \in \mathcal{A}} \theta_a \lambda_0 \lambda_a - \lambda_0 \Phi(\theta) - \Omega(\lambda)\right\} \tag{2.28}$$

While this functional form duplicates the exponential family's, the interpretation is different: the density is over the space of parameters Θ, and determined by hyperparameters λ. The conjugate prior is *proper*, or normalizable, when the hyperparameters take values in the space Λ where the log normalization constant $\Omega(\lambda)$ is finite:

$$\Omega(\lambda) = \log \int_{\Theta} \exp\left\{\sum_{a \in \mathcal{A}} \theta_a \lambda_0 \lambda_a - \lambda_0 \Phi(\theta)\right\} d\theta \tag{2.29}$$

$$\Lambda \triangleq \Big\{\lambda \in \mathbb{R}^{|\mathcal{A}|+1} \mid \Omega(\lambda) < \infty\Big\} \tag{2.30}$$

Note that the dimension of the conjugate family's hyperparameters λ is one larger than the corresponding canonical parameters θ.

The following result verifies that the conjugate family of eq. (2.28) satisfies the definition of eq. (2.27), and provides an intuitive interpretation for the hyperparameters:

Proposition 2.1.4. *Let $p(x \mid \theta)$ denote an exponential family with canonical parameters θ, and $p(\theta \mid \lambda)$ a family of conjugate priors defined as in eq. (2.28). Given L independent samples $\{x^{(\ell)}\}_{\ell=1}^{L}$, the posterior distribution remains in the same family:*

$$p(\theta \mid x^{(1)}, \ldots, x^{(L)}, \lambda) = p\big(\theta \mid \bar{\lambda}\big) \tag{2.31}$$

$$\bar{\lambda}_0 = \lambda_0 + L \qquad \bar{\lambda}_a = \frac{\lambda_0 \lambda_a + \sum_{\ell=1}^{L} \phi_a(x^{(\ell)})}{\lambda_0 + L} \qquad a \in \mathcal{A} \tag{2.32}$$

Integrating over Θ, the log–likelihood of the observations can then be compactly written using the normalization constant of eq. (2.29):

$$\log p(x^{(1)}, \ldots, x^{(L)} \mid \lambda) = \Omega\big(\bar{\lambda}\big) - \Omega(\lambda) + \sum_{\ell=1}^{L} \log \nu(x^{(\ell)}) \tag{2.33}$$

Proof. Expanding the posterior distribution as in eq. (2.18), we have

$$\begin{aligned}
p(\theta \mid x^{(1)}, \ldots, x^{(L)}, \lambda) &\propto p(\theta \mid \lambda) \prod_{\ell=1}^{L} p(x^{(\ell)} \mid \theta) \\
&\propto \exp\left\{ \sum_{a \in \mathcal{A}} \theta_a \lambda_0 \lambda_a - \lambda_0 \Phi(\theta) \right\} \prod_{\ell=1}^{L} \nu(x^{(\ell)}) \exp\left\{ \sum_{a \in \mathcal{A}} \theta_a \phi_a(x^{(\ell)}) - \Phi(\theta) \right\} \\
&\propto \exp\left\{ \sum_{a \in \mathcal{A}} \theta_a \left(\lambda_0 \lambda_a + \sum_{\ell=1}^{L} \phi_a(x^{(\ell)}) \right) - (\lambda_0 + L)\, \Phi(\theta) \right\} \prod_{\ell=1}^{L} \nu(x^{(\ell)}) \\
&\propto \exp\left\{ \sum_{a \in \mathcal{A}} \theta_a \, (\lambda_0 + L) \left(\frac{\lambda_0 \lambda_a + \sum_{\ell=1}^{L} \phi_a(x^{(\ell)})}{\lambda_0 + L} \right) - (\lambda_0 + L)\, \Phi(\theta) \right\}
\end{aligned}$$

Note that the last line absorbs the reference measure terms, which are constant with respect to θ, into the proportionality constant. The posterior hyperparameters of eq. (2.32) can now be verified by comparison with eq. (2.28). Likelihoods are determined by the following integral over Θ:

$$\begin{aligned}
p(x^{(1)}, \ldots, x^{(L)} \mid \lambda) &= \int_{\Theta} p(\theta \mid \lambda) \prod_{\ell=1}^{L} p(x^{(\ell)} \mid \theta)\, d\theta \\
&= \int_{\Theta} \exp\left\{ \sum_{a \in \mathcal{A}} \theta_a \lambda_0 \lambda_a - \lambda_0 \Phi(\theta) - \Omega(\lambda) \right\} \prod_{\ell=1}^{L} \nu(x^{(\ell)}) \exp\left\{ \sum_{a \in \mathcal{A}} \theta_a \phi_a(x^{(\ell)}) - \Phi(\theta) \right\} d\theta \\
&= \exp\{-\Omega(\lambda)\} \int_{\Theta} \exp\left\{ \sum_{a \in \mathcal{A}} \theta_a \left(\lambda_0 \lambda_a + \sum_{\ell=1}^{L} \phi_a(x^{(\ell)}) \right) - (\lambda_0 + L) \Phi(\theta) \right\} d\theta \prod_{\ell=1}^{L} \nu(x^{(\ell)})
\end{aligned}$$

Identifying the second term as an unnormalized conjugate prior, with hyperparameters $\bar{\lambda}$, the log–likelihood of eq. (2.33) then follows from eq. (2.29). □

Note that the predictive likelihood $p(\bar{x} \mid x^{(1)}, \ldots, x^{(L)}, \lambda)$ of eq. (2.19) arises as a special case of Prop. 2.1.4, where eq. (2.33) is used to determine the likelihood of $\bar{x}$ given hyperparameters incorporating previous observations (eq. (2.32)). For many common exponential families, the log normalization constant $\Omega(\lambda)$ can be determined in closed form, and likelihoods are easily computed.

Examining eq. (2.32), we see that the posterior hyperparameters $\bar{\lambda}_a$ are a weighted average of the prior hyperparameters λ_a and the corresponding sufficient statistics of the observations. Conjugate priors are thus effectively described by a set of synthetic *pseudo–observations*, where λ_a is interpreted as the average of $\phi_a(x)$ with respect to this synthetic data. Confidence in these prior statistics is expressed via the effective size $\lambda_0 > 0$ of this synthetic dataset, which need not be integral. This interpretation

often makes it easy to select an appropriate conjugate prior, since hyperparameters correspond to sufficient statistics with intuitive meaning.

When the number of observations L is large relative to λ_0, the posterior distribution of eq. (2.31) is primarily determined by the observed sufficient statistics. Thus, while conjugate families do not always contain truly non–informative *reference priors* [21], sufficiently uninformative, or *vague*, conjugate priors can typically be constructed when desired. More often, however, we find the ability to tractably include informative prior knowledge to be very useful. In cases where conjugate priors cannot adequately capture prior beliefs, mixtures of conjugate priors are often effective [21].

In principle, Prop. 2.1.4 provides a framework for conjugate analysis with any exponential family. In practice, however, canonical parameters may not provide the most convenient, computationally efficient representation. The following sections examine two conjugate families used extensively in this thesis, and develop specialized learning and inference methods with practical advantages.

■ 2.1.3 Dirichlet Analysis of Multinomial Observations

Consider a random variable x taking one of K discrete, categorical values, so that $\mathcal{X} = \{1, \ldots, K\}$. Any probability mass function, or distribution, $p(x)$ is then parameterized by the probabilities $\pi_k \triangleq \Pr[x = k]$ of the K discrete outcomes:

$$p(x \mid \pi_1, \ldots, \pi_K) = \prod_{k=1}^{K} \pi_k^{\delta(x,k)} \qquad \delta(x,k) \triangleq \begin{cases} 1 & x = k \\ 0 & x \neq k \end{cases} \tag{2.34}$$

Given L observations $\{x^{(\ell)}\}_{\ell=1}^{L}$, the *multinomial* distribution [21, 107, 229] gives the total probability of all possible length L discrete sequences taking those values:

$$p(x^{(1)}, \ldots, x^{(L)} \mid \pi_1, \ldots, \pi_K) = \frac{L!}{\prod_k C_k!} \prod_{k=1}^{K} \pi_k^{C_k} \qquad C_k \triangleq \sum_{\ell=1}^{L} \delta(x^{(\ell)}, k) \tag{2.35}$$

When $K = 2$, this is known as the *binomial* distribution. Through comparison with eq. (2.1), we see that multinomial distributions define regular exponential families with sufficient statistics $\phi_k(x) = \delta(x, k)$ and canonical parameters $\theta_k = \log \pi_k$. In a minimal representation, only the first $(K-1)$ statistics are necessary. The multinomial distribution is valid when its parameters lie in the $(K-1)$–simplex:

$$\Pi_{K-1} \triangleq \left\{ (\pi_1, \ldots, \pi_K) \;\middle|\; \pi_k \geq 0,\ \sum_{k=1}^{K} \pi_k = 1 \right\} \tag{2.36}$$

$$= \left\{ (\pi_1, \ldots, \pi_{K-1}, 1 - \textstyle\sum_{k=1}^{K-1} \pi_k) \;\middle|\; \pi_k \geq 0,\ \sum_{k=1}^{K-1} \pi_k \leq 1 \right\} \tag{2.37}$$

Note that the minimal representation of eq. (2.37) implicitly defines π_K as the complement of the probabilities of the other $(K-1)$ categories.

Given L observations as in eq. (2.35), Prop. 2.1.3 shows that the maximum likelihood estimates of the multinomial parameters $\pi = (\pi_1, \ldots, \pi_K)$ equal the empirical frequencies of the discrete categories:

$$\hat{\pi} = \arg\max_{\pi} \sum_{\ell=1}^{L} \log p(x^{(\ell)} \mid \pi) = \left(\frac{C_1}{L}, \ldots, \frac{C_K}{L} \right) \tag{2.38}$$

However, when L is not much larger than K, the ML estimate may assign zero probability to some values, and produce misleading predictions. In the following section, we describe a widely used family of conjugate priors which is useful in these situations.

Dirichlet and Beta Distributions

The *Dirichlet* distribution [21, 107] is the conjugate prior for the multinomial exponential family. Adapting the general form of eq. (2.28), the Dirichlet distribution with hyperparameters $\alpha = (\alpha_1, \ldots, \alpha_K)$ can be written as follows:

$$p(\pi \mid \alpha) = \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \prod_{k=1}^{K} \pi_k^{\alpha_k - 1} \qquad \alpha_k > 0 \tag{2.39}$$

Note that the Dirichlet distribution's normalization constant involves a ratio of gamma functions. By convention, the exponents are defined to equal $(\alpha_k - 1)$ so that the density's mean has the following simple form:

$$\mathbb{E}_{\alpha}[\pi_k] = \frac{\alpha_k}{\alpha_0} \qquad \alpha_0 \triangleq \sum_{k=1}^{K} \alpha_k \tag{2.40}$$

We use $\mathrm{Dir}(\alpha)$ to denote a Dirichlet density with hyperparameters α. Samples can be drawn from a Dirichlet distribution by normalizing a set of K independent gamma random variables [107].

Often, we have no prior knowledge distinguishing the categories, and the K hyperparameters are thus set symmetrically as $\alpha_k = \alpha_0/K$. The variance of the multinomial parameters then equals

$$\mathrm{Var}_{\alpha}[\pi_k] = \frac{K-1}{K^2(\alpha_0 + 1)} \qquad \alpha_k = \frac{\alpha_0}{K} \tag{2.41}$$

See [107] for other moments of the Dirichlet distribution. Because the variance is inversely proportional to α_0, it is known as the *precision* parameter. With a minor abuse of notation, we sometimes use $\mathrm{Dir}(\alpha_0)$ to denote this symmetric prior.

When $K = 2$, the Dirichlet distribution is equivalent to the *beta* distribution [107]. Denoting the beta density's two hyperparameters by α and β, let $\pi \sim \mathrm{Beta}(\alpha, \beta)$ indicate that

$$p(\pi \mid \alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\,\Gamma(\beta)} \pi^{\alpha - 1} (1 - \pi)^{\beta - 1} \qquad \alpha, \beta > 0 \tag{2.42}$$

Note that by convention, samples from the beta density are the probability $\pi \in [0, 1]$ of the first category, while the two–dimensional Dirichlet distribution is equivalently expressed in terms of the probability vector $(\pi, 1 - \pi)$ (see eq. (2.39)). As in eqs. (2.40) and (2.41), the beta density's hyperparameters can be interpreted as setting the prior mean and variance of the binomial parameter π.

In Fig. 2.1, we illustrate several beta distributions. When $\alpha = \beta = 1$, it assigns equal prior probability to all possible binomial parameters π. Larger hyperparameters (smaller variances) lead to unimodal priors concentrated on the chosen mean. We also show examples of Dirichlet distributions on $K = 3$ multinomial categories, using the minimal 2–simplex representation of eq. (2.37). As with the beta density, setting $\alpha_k = 1$ ($\alpha_0 = K$) defines a uniform prior on the simplex, while larger precisions lead to unimodal priors. Interestingly, smaller values of the hyperparameters ($\alpha_k < 1$) favor *sparse* multinomial distributions which assign most of their probability mass to a subset of the categories.

When analyzing multinomial data, it is sometimes useful to consider *aggregate* distributions defined by combining a subset of the categories. If $\pi \sim \mathrm{Dir}(\alpha)$, the multinomial parameters attained by aggregation are also Dirichlet [107]. For example, combining the first two categories, we have

$$(\pi_1 + \pi_2, \pi_3, \ldots, \pi_K) \sim \mathrm{Dir}(\alpha_1 + \alpha_2, \alpha_3, \ldots, \alpha_K) \tag{2.43}$$

More generally, aggregation of any subset of the categories produces a Dirichlet distribution with hyperparameters summed as in eq. (2.43). In particular, the marginal distribution of any single component of a Dirichlet distribution follows a beta density:

$$\pi_k \sim \mathrm{Beta}(\alpha_k, \alpha_0 - \alpha_k) \tag{2.44}$$

This representation leads to an alternative, sequential procedure for drawing random Dirichlet samples [107, 147].

Conjugate Posteriors and Predictions

Consider a set of L observations $\{x^{(\ell)}\}_{\ell=1}^{L}$ from a multinomial distribution $p(x \mid \pi)$, with Dirichlet prior $p(\pi \mid \alpha)$. Via conjugacy, the posterior distribution is also Dirichlet:

$$\begin{aligned} p(\pi \mid x^{(1)}, \ldots, x^{(L)}, \alpha) &\propto p(\pi \mid \alpha)\, p(x^{(1)}, \ldots, x^{(L)} \mid \pi) \\ &\propto \prod_{k=1}^{K} \pi_k^{\alpha_k + C_k - 1} \propto \mathrm{Dir}(\alpha_1 + C_1, \ldots, \alpha_K + C_K) \end{aligned} \tag{2.45}$$

Here, C_k is the number of observations of category k, as in eq. (2.35). If L is sufficiently large, the mean of this posterior distribution (see eq. (2.40)) provides a useful summary statistic. We see that α_k is equivalent to a (possibly non–integral) number of pseudo–observations of category k, and the precision α_0 is the total size of the pseudo–dataset.

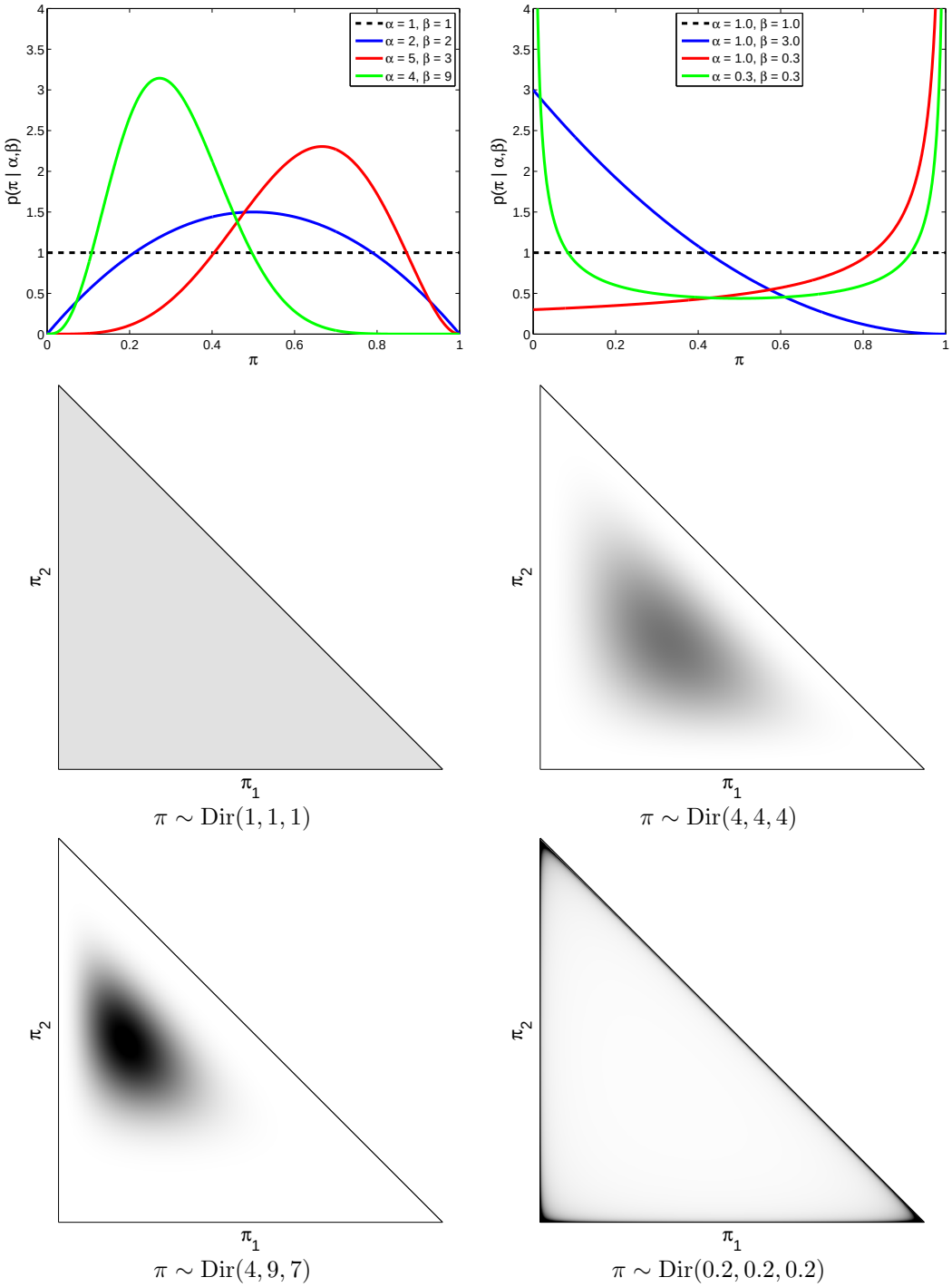


Figure 2.1. Examples of beta and Dirichlet distributions. *Top:* Beta densities with large hyperparameters are unimodal (left), while small values favor biased binomial distributions (right). *Bottom:* Dirichlet densities on $K = 3$ categories, visualized on the simplex $\Pi_2 = (\pi_1, \pi_2, 1 - \pi_1 - \pi_2)$. We show a uniform prior, an unbiased unimodal prior, a biased prior with larger precision α_0, and a prior favoring sparse multinomial distributions. Darker intensities indicate regions with higher probability.

As discussed previously, the predictive likelihood of future observations $\bar{x}$ (as in eq. (2.19)) is often of interest. Using the Dirichlet normalization constant of eq. (2.39) and cancelling terms, it can be shown that

$$p(\bar{x} = k \mid x^{(1)}, \ldots, x^{(L)}, \alpha) = \frac{C_k + \alpha_k}{L + \alpha_0} \tag{2.46}$$

Note that C_k is the number of times category k was observed in the previous L observations (excluding $\bar{x}$). Importantly, these observation counts provide easily updated sufficient statistics which allow rapid predictive likelihood evaluation. Comparing this prediction to that of eq. (2.38), we see that the raw frequencies underlying the ML estimate have been *smoothed* by the pseudo–counts contributed by the Dirichlet prior. More generally, Prop. 2.1.4 can be used to express the likelihood of multiple observations as a ratio gamma functions [123].

■ 2.1.4 Normal–Inverse–Wishart Analysis of Gaussian Observations

Consider a continuous–valued random variable x taking values in d–dimensional Euclidean space $\mathcal{X} = \mathbb{R}^d$. A *Gaussian* or *normal* distribution [21, 107, 229] with mean μ and covariance matrix Λ then has the following form:

$$p(x \mid \mu, \Lambda) = \frac{1}{(2\pi)^{d/2}|\Lambda|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu)^T\Lambda^{-1}(x-\mu)\right\} \tag{2.47}$$

This distribution, which we denote by $\mathcal{N}(\mu, \Lambda)$, is normalizable if and only if Λ is positive definite. Given L independent Gaussian observations $\{x^{(\ell)}\}_{\ell=1}^L$, their joint likelihood is

$$p(x^{(1)}, \ldots, x^{(L)} \mid \mu, \Lambda) \propto |\Lambda|^{-L/2} \exp\left\{-\frac{1}{2}\sum_{\ell=1}^{L}(x^{(\ell)}-\mu)^T\Lambda^{-1}(x^{(\ell)}-\mu)\right\} \tag{2.48}$$

The maximum likelihood estimates of the Gaussian's parameters, based on this data, are the sample mean and covariance:

$$\hat{\mu} = \frac{1}{L}\sum_{\ell=1}^{L} x^{(\ell)} \qquad\qquad \hat{\Lambda} = \frac{1}{L}\sum_{\ell=1}^{L}(x^{(\ell)}-\hat{\mu})(x^{(\ell)}-\hat{\mu})^T \tag{2.49}$$

Expanding the quadratic form of eq. (2.47), we see that Gaussian densities define a regular exponential family, with canonical parameters proportional to the Gaussian's *information* parameterization $(\Lambda^{-1}, \Lambda^{-1}\mu)$. The sample mean and covariance, or equivalently sums of the observations and their outer products, provide sufficient statistics.

Gaussian Inference

Suppose that x and y are two jointly Gaussian random vectors, with distribution

$$\begin{bmatrix} x \\ y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} \Lambda_x & \Lambda_{xy} \\ \Lambda_{yx} & \Lambda_y \end{bmatrix}\right) \tag{2.50}$$

Assuming a fixed covariance, the conjugate prior for a Gaussian's mean is another Gaussian. The conditional distribution of x given y is thus also Gaussian [107, 167, 229], with mean $\widehat{x}$ and covariance $\widehat{\Lambda}_x$ given by the *normal equations*:

$$\widehat{x} = \mu_x + \Lambda_{xy}{\Lambda_y}^{-1}(y - \mu_y) \tag{2.51}$$

$$\widehat{\Lambda}_x = \Lambda_x - \Lambda_{xy}{\Lambda_y}^{-1}\Lambda_{yx} \tag{2.52}$$

The conditional mean $\widehat{x}$ is the *linear least squares estimate* minimizing the mean squared error $\mathbb{E}\big[(x - \widehat{x})^2 \mid y\big]$, while the *error covariance* matrix $\widehat{\Lambda}_x$ measures the reliability of $\widehat{x}$. Note that for Gaussian densities, $\widehat{\Lambda}_x$ is not a function of the observed vector y, but does depend on the joint statistics of x and y.

In many problem domains, the observations y are naturally expressed as a noisy linear function of the latent variables x:

$$y = Cx + v \qquad\qquad v \sim \mathcal{N}(\mu_v, \Lambda_v) \tag{2.53}$$

Assuming x and v are independent, the normal equations then become

$$\widehat{x} = \mu_x + \Lambda_x C^T\big(C\Lambda_x C^T + \Lambda_v\big)^{-1}(y - (C\mu_x + \mu_v)) \tag{2.54}$$

$$\widehat{\Lambda}_x = \Lambda_x - \Lambda_x C^T\big(C\Lambda_x C^T + \Lambda_v\big)^{-1}C\Lambda_x \tag{2.55}$$

Often, these equations are more conveniently expressed in an alternative *information form*. Assuming Λ_x and Λ_v are both positive definite, the matrix inversion lemma [130] allows eqs. (2.54, 2.55) to be rewritten as follows:

$$\widehat{\Lambda}_x^{-1}\widehat{x} = {\Lambda_x}^{-1}\mu_x + C^T{\Lambda_v}^{-1}(y - \mu_v) \tag{2.56}$$

$$\widehat{\Lambda}_x^{-1} = {\Lambda_x}^{-1} + C^T{\Lambda_v}^{-1}C \tag{2.57}$$

This information form plays an important role in the development of tractable computational methods for Gaussian graphical models (see Sec. 2.2.2).

Normal–Inverse–Wishart Distributions

Any distribution satisfying certain spherical symmetries has a representation as a continuous mixture of Gaussian densities, for some prior on that Gaussian's covariance matrix [21, Sec. 4.4]. The conjugate prior for the covariance matrix of a Gaussian distribution with known mean is the *inverse–Wishart* distribution [107], a multivariate generalization of the scaled inverse–χ^2 density. The d–dimensional inverse–Wishart density, with covariance parameter Δ and ν degrees of freedom,[2] equals

$$p(\Lambda \mid \nu, \Delta) \propto |\Lambda|^{-\left(\frac{\nu+d+1}{2}\right)} \exp\left\{-\frac{1}{2}\operatorname{tr}(\nu\Delta\Lambda^{-1})\right\} \tag{2.58}$$

[2] In some texts [107], inverse–Wishart distributions are instead parameterized by a scale matrix $\nu\Delta$.

We denote this density by $\mathcal{W}(\nu, \Delta)$. An inverse–Wishart prior is proper when $\nu > d$, and skewed towards larger covariances, so that its mean and mode equal

$$\mathbb{E}_\nu[\Lambda] = \frac{\nu}{\nu - d - 1}\Delta \qquad \nu > d + 1 \tag{2.59}$$

$$\arg\max_{\Lambda} \mathcal{W}(\Lambda; \nu, \Delta) = \frac{\nu}{\nu + d + 1}\Delta \tag{2.60}$$

The degrees of freedom ν acts as a precision parameter, and can be interpreted as the size of a pseudo–dataset with sample covariance Δ. However, because the inverse–Wishart density is rotationally invariant, it cannot model situations in which the degree of prior knowledge varies across different covariance entries or subspaces. Inverse–Wishart samples can be drawn via appropriate transformations of standard Gaussian random variables [107].

If a multivariate Gaussian's mean and covariance are both uncertain, the *normal–inverse–Wishart* distribution [107] provides an appropriate conjugate prior. Following eq. (2.58), the covariance matrix is assigned an inverse–Wishart prior $\Lambda \sim \mathcal{W}(\nu, \Delta)$. Conditioned on Λ, the mean $\mu \sim \mathcal{N}(\vartheta, \Lambda/\kappa)$. Here, ϑ is the expected mean, for which we have κ pseudo–observations on the scale of observations $x \sim \mathcal{N}(\mu, \Lambda)$. The joint prior distribution, denoted by $\mathcal{NW}(\kappa, \vartheta, \nu, \Delta)$, then takes the following form:

$$p(\mu, \Lambda \mid \kappa, \vartheta, \nu, \Delta) \propto |\Lambda|^{-\left(\frac{\nu+d}{2}+1\right)} \exp\left\{-\frac{1}{2}\operatorname{tr}(\nu\Delta\Lambda^{-1}) - \frac{\kappa}{2}(\mu - \vartheta)^T\Lambda^{-1}(\mu - \vartheta)\right\} \tag{2.61}$$

Fig. 2.2 illustrates a normal–inverse–χ^2 density, the special case arising when $d = 1$. Note that the mean and variance are dependent, so that there is greater uncertainty in the mean value for larger underlying variances. This scaling is often, but not always, appropriate, and is necessary if conjugacy is desired [107]. Fig. 2.2 also shows several Gaussian distributions drawn from a two–dimensional normal–inverse–Wishart prior.

Conjugate Posteriors and Predictions

Consider a set of L observations $\{x^{(\ell)}\}_{\ell=1}^{L}$ from a multivariate Gaussian distribution $\mathcal{N}(\mu, \Lambda)$ with normal–inverse–Wishart prior $\mathcal{NW}(\kappa, \vartheta, \nu, \Delta)$. Via conjugacy, the posterior distribution $p\big(\mu, \Lambda \mid x^{(1)}, \ldots, x^{(\ell)}, \kappa, \vartheta, \nu, \Delta\big)$ is also normal–inverse–Wishart, and thus compactly described by a set of updated hyperparameters $\mathcal{NW}\big(\bar{\kappa}, \bar{\vartheta}, \bar{\nu}, \bar{\Delta}\big)$. Through manipulation of the quadratic form in eq. (2.61), it can be shown [107] that these posterior hyperparameters equal

$$\bar{\kappa}\bar{\vartheta} = \kappa\vartheta + \sum_{\ell=1}^{L} x^{(\ell)} \qquad \bar{\kappa} = \kappa + L \tag{2.62}$$

$$\bar{\nu}\bar{\Delta} = \nu\Delta + \sum_{\ell=1}^{L} x^{(\ell)}x^{(\ell)^T} + \kappa\vartheta\vartheta^T - \bar{\kappa}\bar{\vartheta}\bar{\vartheta}^T \qquad \bar{\nu} = \nu + L \tag{2.63}$$

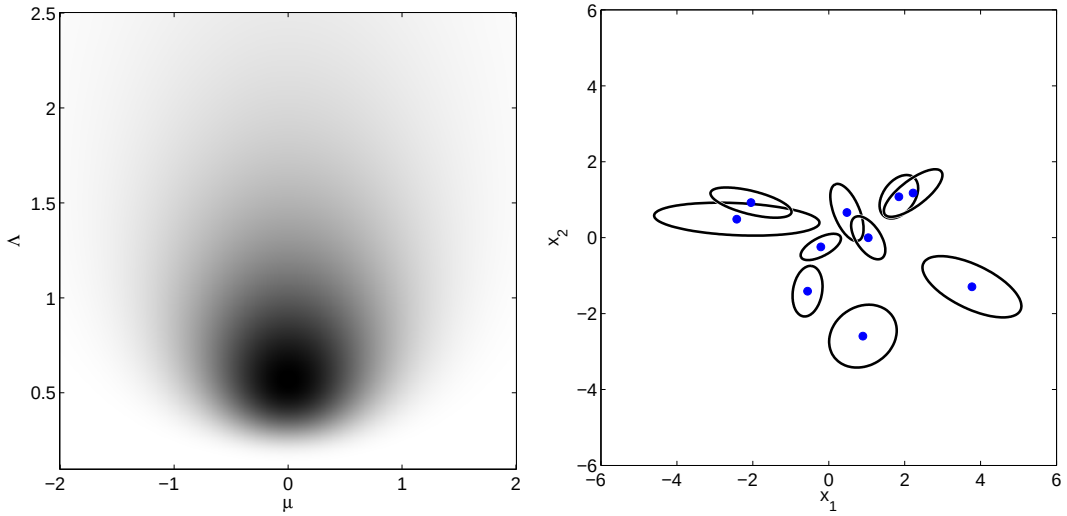


Figure 2.2. Examples of normal–inverse–Wishart distributions. *Left:* Joint probability density of a scalar normal–inverse–χ^2 distribution $(\mu, \Lambda) \sim \mathcal{NW}(2, 0, 4, 1)$. *Right:* Covariance ellipses corresponding to ten samples from a two–dimensional normal–inverse–Wishart distribution $(\mu, \Lambda) \sim \mathcal{NW}(0.3, 0, 4, I_2)$.

To efficiently represent these posterior parameters, we can cache the observations' sum (eq. (2.62)), and the *Cholesky decomposition* [63, 118] of the sum of observation outer products (eq. (2.63)). Cholesky decompositions are numerically robust, can be recursively updated as observations are added or removed, and allow fast likelihood evaluation through the solution of triangulated linear systems.

Integrating over the parameters of the normal–inverse–Wishart posterior distribution, the predictive likelihood of a new observation $\bar{x}$ is multivariate Student–t with $(\bar{\nu} - d + 1)$ degrees of freedom [107]. Assuming $\bar{\nu} > (d + 1)$, this posterior density has finite covariance, and can be approximated by a moment–matched Gaussian:

$$p(\bar{x} \mid x^{(1)}, \ldots, x^{(L)}, \kappa, \vartheta, \nu, \Delta) \approx \mathcal{N}\left(\bar{x}; \bar{\vartheta}, \frac{(\bar{\kappa} + 1)\bar{\nu}}{\bar{\kappa}(\bar{\nu} - d - 1)} \bar{\Delta}\right) \tag{2.64}$$

As illustrated in Fig. 2.3, Student–t distributions have heavier tails than Gaussians, due to integration over uncertainty in the true covariance. However, the KL divergence plot of Fig. 2.3 shows that, for small d, the Gaussian approximation is accurate unless $\bar{\nu}$ is very small. Examining eqs. (2.62, 2.63), we see that the predictive likelihood depends on regularized estimates of the mean and covariance of previous observations.

■ 2.2 Graphical Models

Many practical applications, including the computer vision tasks investigated in this thesis, involve very large collections of random variables. In these situations, direct application of the classic exponential families introduced in the previous section is typically infeasible. For example, a multinomial model of the joint distribution of

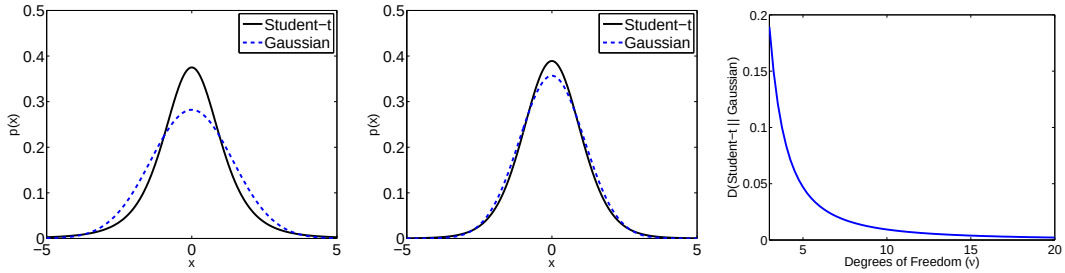


Figure 2.3. Approximation of Student–t predictive distributions by a Gaussian with moments matched as in eq. (2.64). We compare one–dimensional Gaussian and heavier–tailed Student–t densities with $\nu = 4$ (left) and $\nu = 10$ (center) degrees of freedom. For moderate ν, the Gaussian approximation becomes very accurate (see plot of KL divergence versus ν, right).

100 binary variables has $2^{100} \approx 10^{30}$ parameters. Even if such a density could be stored and manipulated, reliable parameter estimation would require an unrealistically massive dataset. Similarly, in fields such as image processing [85, 95, 189, 285] and oceanography [86, 330], estimation of random fields containing millions of continuous variables is not uncommon. However, explicit computations with large, unstructured covariance matrices are extremely difficult [63], typically requiring specialized, parallel hardware.

Probabilistic *graphical models* provide a powerful, flexible framework which addresses these concerns [40, 50, 159, 177, 231, 249, 311, 339]. Graphs are used to decompose multivariate, joint distributions into a set of local interactions among small subsets of variables. These local relationships produce conditional independencies which lead to efficient learning and inference algorithms. Moreover, their modular structure provides an intuitive language for expressing domain–specific knowledge about variable relationships, and facilitates the transfer of modeling advances to new applications.

In the following sections, we introduce and compare several different families of graphical models, including directed Bayesian networks, undirected Markov random fields, and factor graphs. We then relate these models to classic notions of exchangeability, motivating a family of *hierarchical* models used extensively in this thesis.

2.2.1 Brief Review of Graph Theory

We begin by reviewing definitions from graph theory which are useful in describing graphical models. For more detailed surveys of these concepts, see [50, 177].

A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consists of a set of *nodes* or *vertices* $\mathcal{V}$, and a corresponding set of *edges* $\mathcal{E}$. Each edge $(i, j) \in \mathcal{E}$ connects two distinct nodes $i, j \in \mathcal{V}$. For *directed graphs*, an edge (i, j) connects a *parent* vertex i to its *child* j, and is pictorially represented by an arrow (see Fig. 2.4(a)). The set of all parents $\Gamma(j)$ of node j is then given by

$$\Gamma(j) \triangleq \{i \in \mathcal{V} \mid (i, j) \in \mathcal{E}\} \tag{2.65}$$

In *undirected graphs*, an edge $(i, j) \in \mathcal{E}$ if and only if $(j, i) \in \mathcal{E}$, as depicted by an

arrowless line (see Fig. 2.4(c)). For such graphs, $\Gamma(j)$ are known as the *neighbors* of node j, since $i \in \Gamma(j)$ whenever $j \in \Gamma(i)$. It is also possible to define *chain graphs* which mix undirected and directed edges [37, 50, 177], but we do not use them in this thesis. Within any graph, a *clique* is a set of nodes for which all pairs are connected by an edge. If the entire graph forms a clique, it is said to be *complete*.

When describing the statistical properties of graphical models, the structural properties of the underlying graph play an important role. A *path* between nodes $i_0 \neq i_T$ is a sequence of distinct nodes $(i_0, i_1, \ldots, i_T)$ such that $(i_{\ell-1}, i_\ell) \in \mathcal{E}$ for $\ell = 1, \ldots, T$. A *cycle*, or loop,[3] is a path which starts and ends with the same node $i_0 = i_T$, and for which all internal nodes $(i_1, \ldots, i_{T-1})$ are distinct. If there is a path (in either direction) between every pair of nodes, $\mathcal{G}$ is *connected*. If an edge joins two non–consecutive vertices within some cycle, it is called a *chord*. When the undirected version of $\mathcal{G}$ (obtained by replacing all directed edges with undirected ones) has no cycles, the graph is *tree–structured*. Within any tree, a *leaf* node has at most one neighbor. Note that it is easy to construct acyclic, directed graphs which are *not* trees. For any graph, the *diameter* equals the number of edges in the longest path between any two nodes.

Hypergraphs extend graphs by introducing *hyperedges* connecting subsets with more than two vertices [177]. We denote a hypergraph by $\mathcal{H} = (\mathcal{V}, \mathcal{F})$, where $\mathcal{V}$ are vertices as before, and each hyperedge $f \in \mathcal{F}$ is some subset of those vertices ($f \subset \mathcal{V}$). Pictorially, we represent hypergraphs by *bipartite graphs* with circular nodes for each vertex $i \in \mathcal{V}$, and square nodes for each hyperedge $f \in \mathcal{F}$ (see Fig. 2.4(b)). Lines are then used to connect hyperedge nodes to their associated vertex set [175].

■ 2.2.2 Undirected Graphical Models

Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ or hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{F})$, graphical models represent probability distributions by associating each node $i \in \mathcal{V}$ with a random variable $x_i \in \mathcal{X}_i$. The structure of the joint distribution $p(x)$, where $x \triangleq \{x_i \mid i \in \mathcal{V}\}$ takes values in the joint sample space $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_N$, is then determined by the corresponding (hyper)edges. In this section, we introduce three closely related families of graphical models which use edges to encode local, probabilistic *constraints*.

Factor Graphs

Hypergraphs $\mathcal{H} = (\mathcal{V}, \mathcal{F})$ provide an intuitive means of describing probability distributions $p(x)$. For any $f \in \mathcal{F}$, let $x_f \triangleq \{x_i \mid i \in f\}$ denote the corresponding set of random variables. A *factor graph* then defines the joint distribution as a normalized product of local *potential functions* defined on these hyperedges:

$$p(x) \propto \prod_{f \in \mathcal{F}} \psi_f(x_f) \tag{2.66}$$

[3] In graph theoretic terminology, a loop is an edge connecting a node to itself [177]. However, as graphical models do not have self–connections, in this thesis we use the terms loop and cycle interchangeably, as is standard in the graphical inference literature [219, 319].

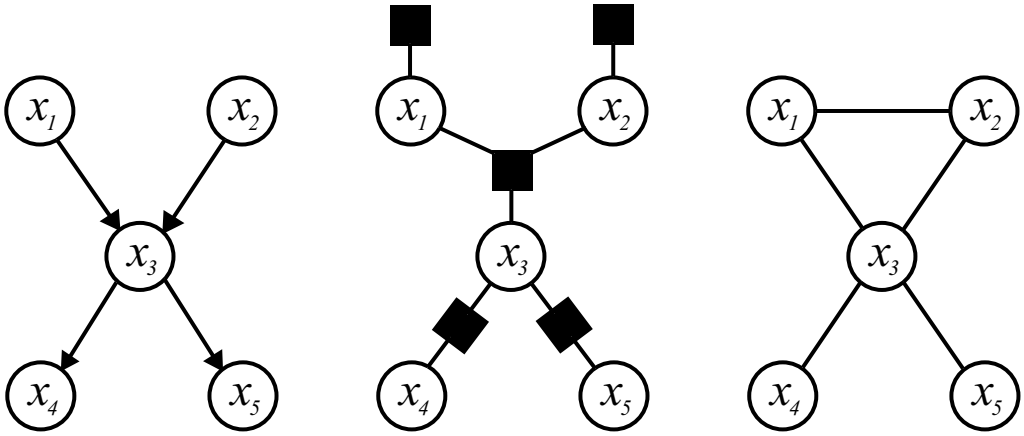


Figure 2.4. Three graphical representations of a distribution over five random variables (see [175]). (a) Directed graph $\mathcal{G}$ depicting a causal, generative process. (b) Factor graph expressing the factorization underlying $\mathcal{G}$. (c) A "moralized" undirected graph capturing the Markov structure of $\mathcal{G}$.

For example, in the factor graph of Fig. 2.5(c), there are 5 variable nodes, and the joint distribution has one potential for each of the 3 hyperedges:

$$p(x) \propto \psi_{123}(x_1, x_2, x_3)\, \psi_{234}(x_2, x_3, x_4)\, \psi_{35}(x_3, x_5)$$

Often, these potentials can be interpreted as local dependencies or constraints. Note, however, that $\psi_f(x_f)$ does *not* typically correspond to the marginal distribution $p_f(x_f)$, due to interactions with the graph's other potentials.

In many applications, factor graphs are used to impose structure on an exponential family of densities. In particular, suppose that each potential function is described by the following unnormalized exponential form:

$$\psi_f(x_f \mid \theta_f) = \nu_f(x_f) \exp\left\{ \sum_{a \in \mathcal{A}_f} \theta_{fa} \phi_{fa}(x_f) \right\} \tag{2.67}$$

Here, $\theta_f \triangleq \{\theta_{fa} \mid a \in \mathcal{A}_f\}$ are the canonical parameters of the *local* exponential family for hyperedge f. From eq. (2.66), the joint distribution can then be written as

$$p(x \mid \theta) = \left(\prod_{f \in \mathcal{F}} \nu_f(x_f) \right) \exp\left\{ \sum_{f \in \mathcal{F}} \sum_{a \in \mathcal{A}_f} \theta_{fa} \phi_{fa}(x_f) - \Phi(\theta) \right\} \tag{2.68}$$

Comparing to eq. (2.1), we see that factor graphs define regular exponential families [104, 311], with parameters $\theta = \{\theta_f \mid f \in \mathcal{F}\}$, whenever local potentials are chosen from such families. The results of Sec. 2.1 then show that *local* statistics, computed over the support of each hyperedge, are sufficient for learning from training data. This

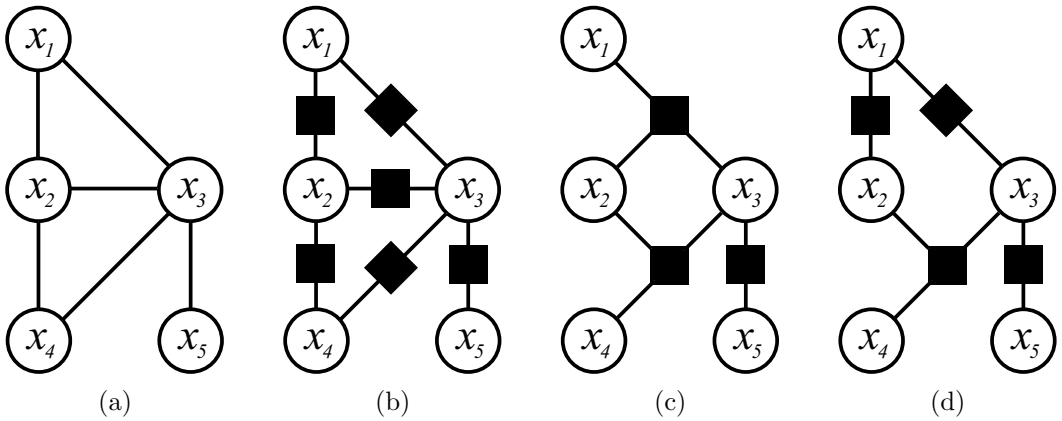


Figure 2.5. An undirected graphical model, and three factor graphs with equivalent Markov properties. (a) Undirected graph $\mathcal{G}$ representing five random variables. (b) Factor graph interpreting $\mathcal{G}$ as a pairwise MRF. (c) Factor graph corresponding to the maximal cliques of $\mathcal{G}$. (d) Another possible factorization which is Markov with respect to $\mathcal{G}$. In all cases, single-node factors are omitted for clarity.

guarantee can be extremely useful for large graphs with many variables. Note, however, that interactions among overlapping potential functions induce global dependencies in the parameters. Thus, as we discuss in more detail in Sec. 2.3, learning can be computationally difficult even when the potentials take simple forms.

Many widely used graphical models correspond to a particular choice of the exponential families in eq. (2.68). For example, any distribution on discrete spaces $\mathcal{X}_i = \{1, \dots, K_i\}$ can be expressed in terms of a set of *indicator* potential functions which enumerate all possible configurations of the variables within each factor [311]. Alternatively, jointly Gaussian random fields take potentials to be local quadratic functions. The graph structure of these *covariance selection* models is then expressed via an inverse covariance matrix which is *sparse*, with many entries equaling zero [64, 177, 268, 276].

The exponential family representation of eq. (2.68) is convenient for learning and parameter estimation. In many applications, however, a model has already been determined (perhaps via MAP estimation as in Sec. 2.1.2), and we are instead interested in *inference* problems. In such cases, we prefer the representation of eq. (2.66), since it highlights the factorization underlying efficient computational methods.

Markov Random Fields

Undirected graphical models, or *Markov random fields (MRFs)*, characterize distributions $p(x)$ via a set of implied conditional independencies. In this section, we describe these Markov properties, and relate them to an algebraic factorization similar to that underlying factor graphs.

Given an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, let f, g and h denote three disjoint subsets of $\mathcal{V}$. Set h is said to *separate* sets f and g if every path between f and g passes through some node in h. A stochastic process x is *globally Markov* with respect to $\mathcal{G}$ if x_f and

x_g are independent conditioned on the variables x_h in any separating set:

$$p(x_f, x_g \mid x_h) = p(x_f \mid x_h)\, p(x_g \mid x_h) \qquad \text{if } h \text{ separates } f \text{ from } g \tag{2.69}$$

This property generalizes temporal Markov processes, for which the past and future are independent conditioned on the present. For example, the undirected graph of Fig. 2.5(a) implies the following conditional independencies, among others:

$$p(x_1, x_2, x_5 \mid x_3, x_4) = p(x_1, x_2 \mid x_3, x_4)\, p(x_5 \mid x_3)$$
$$p(x_1, x_4, x_5 \mid x_2, x_3) = p(x_1 \mid x_2, x_3)\, p(x_4 \mid x_2, x_3)\, p(x_5 \mid x_3)$$

An important special case of eq. (2.69) guarantees that conditioned on its immediate neighbors, the random variable at any node is independent of the rest of the process:

$$p\big(x_i \mid x_{\mathcal{V}\setminus i}\big) = p\big(x_i \mid x_{\Gamma(i)}\big) \tag{2.70}$$

As we discuss in later sections, this *local Markov property* plays an important role in the design of efficient learning and inference algorithms.

The following theorem, due to Hammersley and Clifford, shows that Markov random fields are naturally parameterized via potential functions defined on the cliques of the corresponding undirected graph.

Theorem 2.2.1 (Hammersley-Clifford). *Let $\mathcal{C}$ denote the set of cliques of an undirected graph $\mathcal{G}$. A probability distribution defined as a normalized product of nonnegative potential functions on those cliques is then always Markov with respect to $\mathcal{G}$:*

$$p(x) \propto \prod_{c \in \mathcal{C}} \psi_c(x_c) \tag{2.71}$$

Conversely, any strictly positive density ($p(x) > 0$ for all x) which is Markov with respect to $\mathcal{G}$ can be represented in this factored form.

Proof. There are a variety of ways to prove this result; see [26, 35, 43] for examples and further discussion. For a degenerate Markov distribution which cannot be factored as in eq. (2.71), see Lauritzen [177]. □

Comparing eq. (2.71) to eq. (2.66), we see that Markov random fields can always be represented by a factor graph with one hyperedge for each of the graph's cliques [175, 339]. This representation is also known as the *clique hypergraph* corresponding to $\mathcal{G}$ [177]. Note that it is possible, but not necessary, to restrict this factorization to *maximal cliques* which are not a strict subset of any other clique (see Fig. 2.5(c)).

In practice, Markov properties are used in two complementary ways. If a stochastic process is known to satisfy certain conditional independencies, the Hammersley–Clifford Theorem then motivates models parameterized by local sufficient statistics. Conversely,

given any graphical model, the implied Markov properties can be exploited to design more efficient learning and inference algorithms.

While undirected graphs fully specify a probability density's Markov structure, they do not unambigously determine that density's factorization into potential functions. For example, in Fig. 2.5 we show three different factor graphs, all of which are Markov with respect to the same undirected graph. Because the differences among these factorizations have implications for learning and inference, the more detailed factor graph representation is often preferable [96, 98, 175].

Pairwise Markov Random Fields

In many applications, it is convenient to consider a restricted class of *pairwise Markov random fields*. Given an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a pairwise MRF expresses the joint distribution as a product of potential functions defined on that graph's edges:

$$p(x) \propto \prod_{(i,j)\in\mathcal{E}} \psi_{ij}(x_i, x_j) \prod_{i\in\mathcal{V}} \psi_i(x_i) \tag{2.72}$$

Because pairs of neighboring nodes always define cliques, the Hammersley–Clifford Theorem guarantees that pairwise MRFs are Markov with respect to $\mathcal{G}$. The inclusion of single–node potentials $\psi_i(x_i)$ is not strictly necessary, but is often convenient. Pairwise MRFs containing only binary variables are known as *Ising models* in the statistical physics literature [337].

Fig. 2.5(b) shows the factor graph corresponding to a pairwise MRF, and contrasts it with models incorporating higher order cliques. To avoid ambiguities, in this thesis we only use undirected graphs to depict pairwise MRFs. For graphical models containing interactions among three or more variables, we instead use a factor graph representation which explicitly reveals the underlying factorization.

Many inference tasks can be posed as the estimation of a set of *latent* or *hidden* variables x based on noisy observations y. In such cases, pairwise MRFs are sometimes used to express the internal structure of the desired posterior distribution:

$$p(x \mid y) = \frac{p(x,y)}{p(y)} \propto \prod_{(i,j)\in\mathcal{E}} \psi_{ij}(x_i, x_j) \prod_{i\in\mathcal{V}} \psi_i(x_i, y) \tag{2.73}$$

Frequently, observations also decompose into local measurements $y = \{y_i \mid i \in \mathcal{V}\}$, so that $\psi_i(x_i, y) = \psi_i(x_i, y_i)$. Fig. 2.6 shows two examples of pairwise MRFs used widely in practice: a multiscale tree–structured graph [34, 41, 85, 86, 189, 330], and a nearest–neighbor grid [26, 95, 108, 196, 285]. In both cases, shaded nodes represent noisy local observations y_i.

■ 2.2.3 Directed Bayesian Networks

We now introduce a different family of graphical models derived from directed graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. As before, *Bayesian networks* associate each node $i \in \mathcal{V}$ with a random

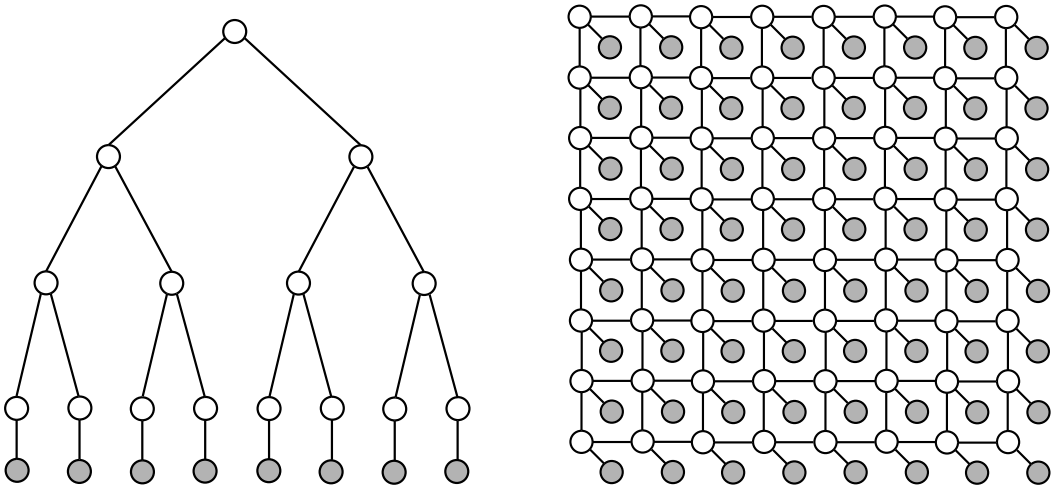


Figure 2.6. Sample pairwise Markov random fields, where open nodes represent hidden variables x_i and shaded nodes are local observations y_i. *Left:* A multiscale tree–structured graph, in which *coarse scale* nodes capture dependencies among an observed *fine scale* process. *Right:* A nearest–neighbor grid in which each hidden variable is connected to its four closest spatial neighbors.

variable x_i. However, in place of potential functions, directed models decompose $p(x)$ via the conditional density of each child node i given its parents $\Gamma(i)$:

$$p(x) = \prod_{i \in \mathcal{V}} p\big(x_i \mid x_{\Gamma(i)}\big) \tag{2.74}$$

For nodes i without parents ($\Gamma(i) = \emptyset$), we define $p\big(x_i \mid x_{\Gamma(i)}\big) = p(x_i)$. This factorization is consistent whenever $\mathcal{G}$ is a directed *acyclic* graph, so that its edges specify a valid partial ordering of the random variables [50, 128, 231]. For example, the directed graph of Fig. 2.4(a) implies the following conditional densities:

$$p(x) = p(x_1)\, p(x_2)\, p(x_3 \mid x_1, x_2)\, p(x_4 \mid x_3)\, p(x_5 \mid x_3)$$

Bayesian networks effectively define a *causal* generative process, beginning with nodes without parents and proceeding from parent to child throughout the graph. In contrast, direct sampling from undirected graphical models is often intractable (see Sec. 2.4).

The Markov properties of directed Bayesian networks are slightly different from those of undirected graphical models. In particular, a random variable x_i is conditionally independent of the remaining process given its parents $x_{\Gamma(i)}$, children $\{x_j \mid i \in \Gamma(j)\}$, and its *children's parents*. These relationships are captured by a corresponding *moral graph* in which parents are connected ("married") by additional undirected edges [50], as in Fig. 2.4(c). Although factor graphs can express this Markov structure (see Fig. 2.4(b)), doing so obscures the underlying causal, generative process. A directed generalization of factor graphs has been proposed [96, 98], but we focus on simpler

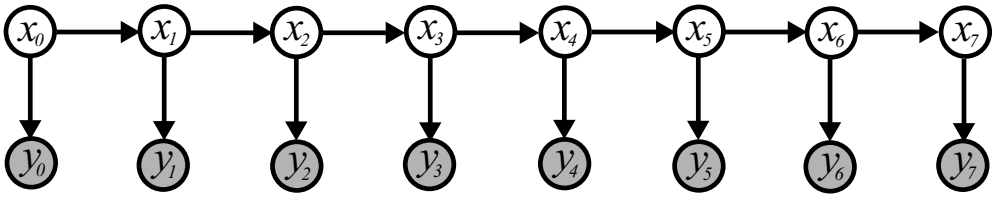


Figure 2.7. Directed graphical representation of a hidden Markov model (HMM) for $T = 8$ samples of a temporal process. The hidden states x_t capture dependencies among the observations y_t.

Bayesian network representations. For a discussion of transformations allowing conversion between undirected, directed, and factor graphs, see [175, 339].

In many applications, exponential families provide convenient parameterizations of the conditional densities composing a Bayesian network. In general, directed models define *curved* exponential families [74], because conditional densities with multiple parents may impose constraints on the set of achievable canonical parameters [104]. However, this subtlety does not arise in the particular models considered by this thesis.

Hidden Markov Models

Directed graphical models provide the basis for a family of *hidden Markov models (HMMs)* which are widely used to model temporal stochastic processes [8, 70, 163, 235]. Let $y = \{y_t\}_{t=0}^{T-1}$ denote observations of a temporal process collected at T discrete time points. We assume that each observation y_t is independently sampled conditioned on an underlying *hidden state* x_t. If we further assume that these states $x = \{x_t\}_{t=0}^{T-1}$ evolve according to a *first–order* temporal Markov process, the joint distribution equals

$$p(x, y) = p(x_0)\, p(y_0 \mid x_0) \prod_{t=1}^{T-1} p(x_t \mid x_{t-1})\, p(y_t \mid x_t) \tag{2.75}$$

Fig. 2.7 shows a directed graphical representation of this density. In later chapters, we extend this model to develop methods for visual tracking of articulated objects.

Historically, models equivalent to HMMs were independently developed in several different domains. For example, in speech recognition the hidden states typically take values on some finite, discrete set, and statistical methods are used to learn dynamics from speech waveforms [235]. In contrast, control theorists often use continuous *state space models* to characterize the position, velocity, and other properties of physical systems [8, 163]. Graphical models unify these disparate approaches, and allow advances in learning and inference methods to be transferred between domains [50, 159, 249].

■ 2.2.4 Model Specification via Exchangeability

In some applications, a graphical model's structure is determined by the physical data generation process. For example, HMMs (see Fig. 2.7) are often derived from a known

dynamical system, while grid–structured MRFs (see Fig. 2.6) can arise from the discretization of stochastic partial differential equations. For other learning tasks, however, the generative process may be unknown, or too complex to characterize explicitly. In this section, we show how simple assumptions about the *indistinguishability* of different observations lead naturally to a family of hierarchical, directed graphical models.

Consider a set of N random variables $\{x_i\}_{i=1}^N$. These variables are said to be *exchangeable* if every permutation, or reordering, of their indices has equal probability:

$$p(x_1, \ldots, x_N) = p\big(x_{\tau(1)}, \ldots, x_{\tau(N)}\big) \qquad \text{for any permutation } \tau(\cdot) \tag{2.76}$$

This expression formalizes the concept of an unordered collection of random variables, for which the chosen indices are purely notational. When no auxiliary information is available, this assumption is usually reasonable. Extending this definition, a sequence $\{x_i\}_{i=1}^\infty$ is *infinitely exchangeable* if every finite subsequence is exchangeable [21, 107].

As shown by the following theorem, exchangeable observations can *always* be represented via a prior distribution over some latent parameter space.

Theorem 2.2.2 (De Finetti). *For any infinitely exchangeable sequence of random variables $\{x_i\}_{i=1}^\infty$, $x_i \in \mathcal{X}$, there exists some space Θ, and corresponding density $p(\theta)$, such that the joint probability of any N observations has a mixture representation:*

$$p(x_1, x_2, \ldots, x_N) = \int_\Theta p(\theta) \prod_{i=1}^N p(x_i \mid \theta)\, d\theta \tag{2.77}$$

When $\mathcal{X}$ is a K–dimensional discrete space, Θ may be chosen as the $(K-1)$–simplex. For Euclidean $\mathcal{X}$, Θ is an infinite–dimensional space of probability measures.

Proof. De Finetti's original proof for binary $\mathcal{X}$ dates to the 1930's; see [127] for a simpler proof of that case, and [21, Sec. 4.5] for generalizations and additional references. □

Technically, the representation of eq. (2.77) is only guaranteed to exist when $\{x_i\}_{i=1}^N$ are part of an infinitely exchangeable sequence. However, for moderate N, the distortion induced by assuming infinite exchangeability, when only finite exchangeability is guaranteed, cannot be significant [21, Prop. 4.19].

De Finetti's theorem is often taken as a justification for Bayesian methods, since the infinite mixture representation of eq. (2.77) corresponds precisely with the marginal likelihood of eq. (2.21). We see that exchangeability does not imply independence of the observations, but *conditional independence* given a set of latent parameters θ. Note also that for continuous sample spaces, these parameters are infinite–dimensional, since there is no finite parameterization for the space of continuous densities. This motivates a class of nonparametric methods which we discuss further in Sec. 2.5.

When applying the representation of eq. (2.77), it is common to assume some family of prior distributions with hyperparameters λ, so that

$$p(x_1, \ldots, x_N, \theta \mid \lambda) = p(\theta \mid \lambda) \prod_{i=1}^N p(x_i \mid \theta) \tag{2.78}$$

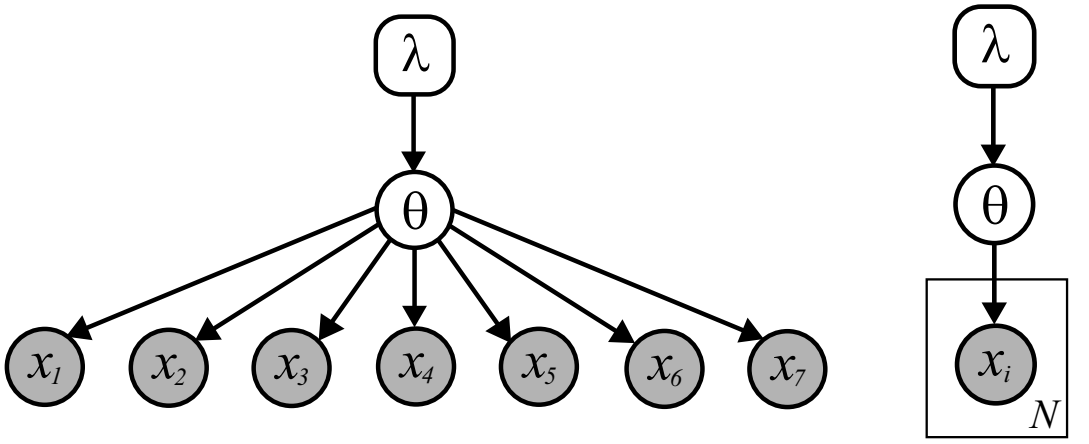


Figure 2.8. De Finetti's representation of N exchangeable random variables $\{x_i\}_{i=1}^N$ as a hierarchical model. Each observation is independently sampled from a density with parameters θ, which are in turn assigned a prior distribution with hyperparameters λ. *Left:* Explicit model for $N = 7$ variables. *Right:* Compact plate representation of the N–fold replication of the observations x_i.

This generative process can be described by the Bayesian network of Fig. 2.8, where *plates* are used to compactly denote replicated variables [37, 159]. In Bayesian statistics, this is known as a *hierarchical model* [21, 107] due to the layering by which observations depend on parameters, which are in turn related to hyperparameters. Note that we have explicitly included the parameters θ and hyperparameters λ (depicted by a rounded box) in the graphical structure. While not strictly necessary, this approach is often useful in learning problems where the parameters are of particular interest [50].

Finite Exponential Family Mixtures

Standard exponential family densities can be too inflexible to accurately describe many complex, multimodal datasets. In these situations, data are often modeled via a finite *mixture distribution* [107, 203, 239, 249]. A K component mixture model takes the following general form:

$$p(x \mid \pi, \theta_1, \ldots, \theta_K) = \sum_{k=1}^{K} \pi_k f(x \mid \theta_k) \qquad \pi \in \Pi_{K-1} \tag{2.79}$$

Each mixture component, or *cluster*, belongs to a parameterized family of probability densities $f(x \mid \theta)$, whose distribution we equivalently denote by $F(\theta)$. Each data point x_i is generated by independently selecting one of K clusters according to the multinomial distribution π, and then sampling from the chosen cluster's data distribution:

$$\begin{aligned} z_i &\sim \pi \\ x_i &\sim F(\theta_{z_i}) \end{aligned} \tag{2.80}$$

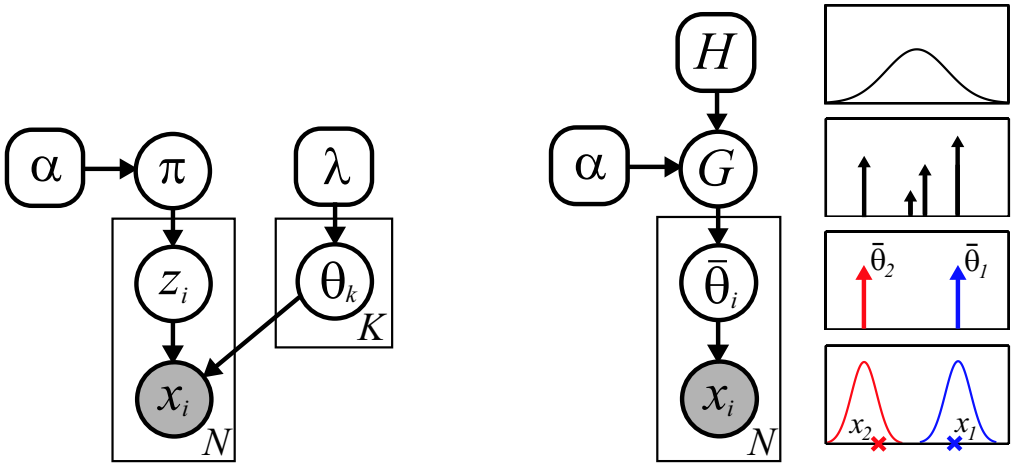


Figure 2.9. Directed graphical representations of a K component mixture model. Mixture weights $\pi \sim \mathrm{Dir}(\alpha)$, while cluster parameters are assigned independent priors $\theta_k \sim H(\lambda)$. *Left:* Indicator variable representation, in which $z_i \sim \pi$ is the cluster that generates $x_i \sim F(\theta_{z_i})$. *Right:* Alternative distributional form, in which G is a discrete distribution on Θ taking K distinct values. $\bar{\theta}_i \sim G$ are the parameters of the cluster that generates $x_i \sim F(\bar{\theta}_i)$. We illustrate with a mixture of $K = 4$ Gaussians, where cluster variances are known (bottom) and $H(\lambda)$ is a Gaussian prior on cluster means (top). Sampled cluster means $\bar{\theta}_1, \bar{\theta}_2$, and corresponding Gaussians, are shown for two observations x_1, x_2.

The unobserved *indicator variable* $z_i \in \{1, \ldots, K\}$ specifies the unique cluster associated with x_i. Mixture models are widely used for *unsupervised learning*, where clusters are used to discover subsets of the data with common attributes.

In most applications of mixture models, $f(x \mid \theta_k)$ is chosen to be an appropriate exponential family. For example, Euclidean observations are often modeled via Gaussian mixtures, so that the parameters $\theta_k = (\mu_k, \Lambda_k)$ specify each cluster's mean μ_k and covariance Λ_k. When learning mixtures from data, it is often useful to place an independent conjugate prior H, with hyperparameters λ, on each cluster's parameters:

$$\theta_k \sim H(\lambda) \qquad\qquad k = 1, \ldots, K \tag{2.81}$$

Similarly, in the absence of prior knowledge distinguishing the clusters, the mixture weights π can be assigned a symmetric Dirichlet prior with precision α:

$$\pi \sim \mathrm{Dir}\Big(\frac{\alpha}{K}, \ldots, \frac{\alpha}{K}\Big) \tag{2.82}$$

Fig. 2.9 shows a directed graphical model summarizing this generative process. As in Fig. 2.8, plates are used to compactly denote the K cluster parameters $\{\theta_k\}_{k=1}^K$ and N data points $\{x_i\}_{i=1}^N$. In Fig. 2.10, we illustrate several two–dimensional Gaussian mixtures sampled from a conjugate, normal–inverse–Wishart prior.

Mixture models can equivalently be expressed in terms of a discrete distribution G

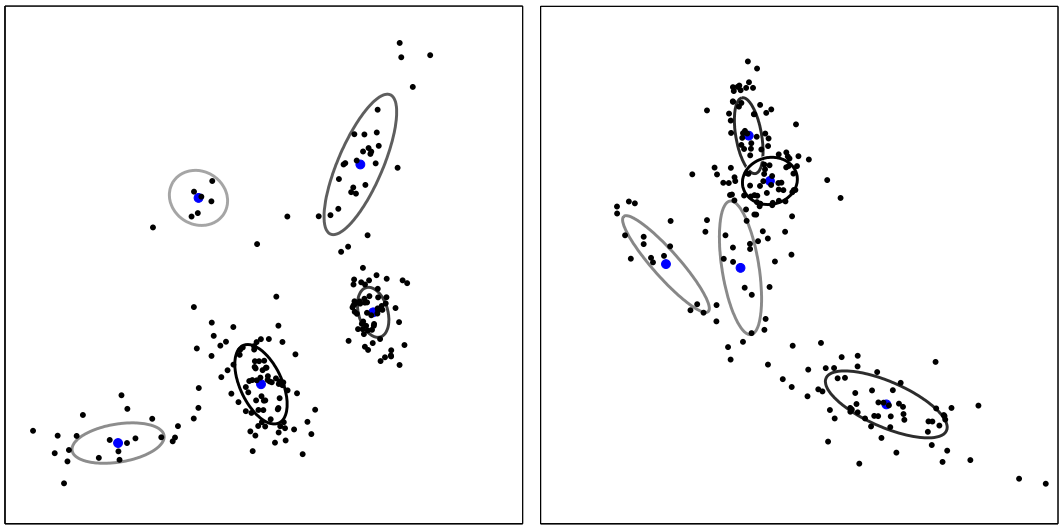


Figure 2.10. Two randomly sampled mixtures of $K = 5$ two–dimensional Gaussians. Mixture parameters are generated from conjugate, normal–inverse–Wishart priors. For each mixture, we plot one standard deviation covariance ellipses Λ_k with intensity proportional to their probability $\pi \sim \mathrm{Dir}(\alpha_0)$, $\alpha_0 = 10$. In each case, we also show $N = 200$ randomly sampled observations.

on the space Θ of cluster parameters:

$$G(\theta) = \sum_{k=1}^{K} \pi_k \delta(\theta, \theta_k) \qquad \begin{aligned} \pi &\sim \mathrm{Dir}(\alpha) \\ \theta_k &\sim H(\lambda) \qquad k = 1, \ldots, K \end{aligned} \tag{2.83}$$

We generate each data point x_i by sampling a set of parameters $\bar{\theta}_i$ from G:

$$\begin{aligned} \bar{\theta}_i &\sim G \\ x_i &\sim F\big(\bar{\theta}_i\big) \end{aligned} \tag{2.84}$$

This representation, which is statistically equivalent to the indicator variables used in eq. (2.80), plays an important role in later hierarchical extensions. Note that G can be seen as a discrete, K component approximation to the infinite–dimensional measure arising in De Finetti's Theorem. Fig. 2.9 shows a graphical representation of this alternative form, and illustrates the generative process for a simple one–dimensional Gaussian mixture with known variance.

The mixture models of Fig. 2.9 assume the number of clusters K to be a fixed, known constant. In general, determining an appropriate mixture size is a difficult problem, which has motivated a wide range of model selection procedures [46, 87, 203, 314]. In Sec. 2.5, we discuss an alternative nonparametric approach which controls complexity by placing prior distributions on *infinite* mixtures.

Analysis of Grouped Data: Latent Dirichlet Allocation

In many domains, there are several *groups* of data which are thought to be produced by related, but distinct, generative processes. For example, medical studies often combine data collected at multiple sites, which examine a common treatment but may have location–specific idiosyncrasies [75, 107]. In text analysis, the words composing a text corpus are typically separated into different documents [31, 123, 140, 289]. Similarly, computer vision systems like those developed in this thesis learn appearance models from visual features detected in different training images [14, 79, 81, 266, 280, 282].

While it is simplest to analyze each group independently, doing so neglects critical information when groups are individually ambiguous. Conversely, combining groups in a single exchangeable dataset may lead to inappropriately biased estimates, and obscures features distinguishing particular groups. By *sharing* random parameters among groups, hierarchical Bayesian models provide an elegant compromise [21, 107, 216]. Posterior dependencies between parameters then effectively transfer information between related experiments, documents, or objects. Estimates based on these distributions are "shrunk" together, so that groups share the strength of other datasets while retaining distinctive features. For example, the classic *James–Stein estimator*, which uniformly dominates the ML estimate of a multivariate Gaussian's mean, can be derived via an empirical Bayesian analysis of a particular hierarchical model [75].

Latent Dirichlet allocation (LDA) [31] extends mixture models (as in Fig. 2.9) to learn clusters describing several related sets of observations. Given J groups of data, let $\mathbf{x}_j = (x_{j1}, \ldots, x_{jN_j})$ denote the N_j data points in group j, and $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_J)$. LDA assumes that the data within each group are exchangeable, and independently sampled from one of K latent clusters with parameters $\{\theta_k\}_{k=1}^K$. Letting $\boldsymbol{\pi}_j \in \Pi_{K-1}$ denote the mixture weights for the j^{th} group, we have

$$p(x_{ji} \mid \boldsymbol{\pi}_j, \theta_1, \ldots, \theta_K) = \sum_{k=1}^{K} \pi_{jk} f(x_{ji} \mid \theta_k) \qquad i = 1, \ldots, N_j \tag{2.85}$$

Comparing to eq. (2.79), we see that for individual groups LDA is equivalent to a finite mixture model. LDA extends standard mixture models by sharing a common set of clusters among several related groups. These shared parameters θ_k allow learning algorithms to transfer information, while distinct mixture weights $\boldsymbol{\pi}_j$ capture the particular features of each group. Because the data within each group are always observed, an explicit generative model for N_j is unnecessary.

To complete this hierarchical model, we must assign a distribution to the mixture weights $\{\boldsymbol{\pi}_j\}_{j=1}^J$. LDA assumes groups have no distinguishing features beyond the data they contain, and are thus exchangeable. De Finetti's Theorem then implies that these mixture weights are independently sampled from some common prior distribution. For computational simplicity, LDA chooses a conjugate Dirichlet prior:

$$\boldsymbol{\pi}_j \sim \mathrm{Dir}(\alpha) \qquad j = 1, \ldots, J \tag{2.86}$$

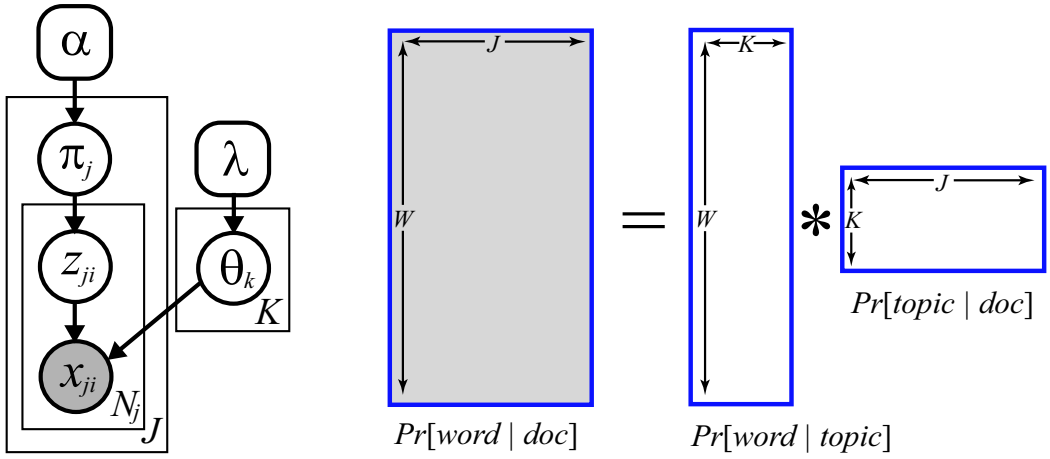


Figure 2.11. The latent Dirichlet allocation (LDA) model for sharing K clusters θ_k among J groups of exchangeable data $\mathbf{x}_j = (x_{j1}, \ldots, x_{jN_j})$. *Left:* LDA as a directed, hierarchical model. Each group's mixture weights $\boldsymbol{\pi}_j \sim \mathrm{Dir}(\alpha)$, while cluster parameters are assigned independent priors $\theta_k \sim H(\lambda)$. $z_{ji} \sim \boldsymbol{\pi}_j$ indicates the shared cluster that generates $x_{ji} \sim F(\theta_{z_{ji}})$. *Right:* When observations are one of W discrete words, LDA can be seen as a probabilistically constrained factorization of the matrix describing the bag of words composing each group, or document. The number K of latent clusters, or topics, determines the factorization's rank. The hyperparameters λ and α define Dirichlet priors for the columns of the word and topic distribution matrices, respectively.

The resulting hierarchical model is illustrated in Fig. 2.11. The Dirichlet hyperparameters α may be either chosen symmetrically (as in eq. (2.41)) to encode prior knowledge [123], or learned from training data in an empirical Bayesian fashion [31]. Often, robustness is improved by assigning conjugate priors $\theta_k \sim H(\lambda)$ to the cluster parameters, as in standard mixture models (see eq. (2.81)). The resulting model is said to be *partially exchangeable* [21], since observations are distinguished only by their associated group. As we demonstrate later, hierarchical graphical models provide a powerful framework for describing dependencies within richly structured datasets.

LDA was originally used to analyze text corpora, by associating groups with documents and data x_{ji} with individual words. The exchangeability assumption treats each document as a "bag of words," incorrectly ignoring the true sentence structure. By doing so, however, LDA leads to tractable algorithms which automatically learn *topics* (clusters) from large, unlabeled document collections [31, 123, 211]. These topics are alternatively known as *aspects*, and LDA as the generative aspect model [211].

For discrete data, LDA effectively determines a low–rank *factorization* of the matrix containing the frequency of each word in each document (see Fig. 2.11). As discussed in detail by Blei et. al. [31], LDA's globally consistent generative model provides conceptual and practical advantages over earlier factorization methods such as *latent semantic analysis* [140]. Importantly, however, LDA can also be generalized to continuous data by associating clusters with appropriate exponential families $F(\theta)$. For example, in later sections of this thesis we use Gaussian "topics" to model spatial data.

As with finite mixture models, the number of clusters or topics K used by LDA is a fixed constant. In practice, learning algorithms are sensitive to this parameter [31, 123], and computationally expensive cross–validation schemes are often needed. Motivated by this issue, Sec. 2.5.4 discusses the hierarchical Dirichlet process [289], a nonparametric generalization of LDA which automatically infers the number of topics needed to explain a given training corpus.

■ 2.2.5 Learning and Inference in Graphical Models

In most applications of graphical models, inference and learning can be posed in terms of a few canonical computational tasks. We divide the random variables composing the graphical model into three sets: observations y, latent or hidden variables x, and parameters θ. While the form of this parameterization differs for directed and undirected graphs, the objectives outlined below arise in both cases.

Inference Given Known Parameters

We begin by assuming the graph's parameters θ are fixed to known, constant values via some previous modeling procedure. The posterior distribution $p(x \mid y, \theta)$ then fully captures available information about the hidden variables x. However, for most realistic graphs the joint sample space $\mathcal{X}$ is far too large to characterize explicitly. For example, given N binary hidden variables, $|\mathcal{X}| = 2^N$. We must thus develop efficient methods to *infer* statistics summarizing this posterior density.

Given global observations y, the joint density $p(x \mid y, \theta)$ is often effectively summarized by the following *posterior marginal distributions:*

$$p(x_i \mid y, \theta) = \int_{\mathcal{X}_{\mathcal{V}\setminus i}} p(x \mid y, \theta)\, dx_{\mathcal{V}\setminus i} \qquad i \in \mathcal{V} \tag{2.87}$$

Here, $\mathcal{V} \setminus i$ denotes all nodes except that corresponding to x_i. The mean of this conditional density is the *Bayes' least squares estimate* [167, 229], while its mode is the *maximizer of the posterior marginals (MPM)* [196] minimizing the expected number of misclassified variables. In addition, the variance or entropy of $p(x_i \mid y, \theta)$ measure the posterior *uncertainty* in these estimates, which can be critical in practical applications [98, 231, 285, 330].

In some cases, hidden variables are instead inferred via a global MAP estimate:

$$\hat{x} = \arg\max_x p(x \mid y, \theta) \tag{2.88}$$

While MAP estimates desirably optimize the joint posterior probability [108], they do not directly provide confidence measures. Furthermore, when observations are noisy or ambiguous, MAP estimation is often less robust than the MPM criterion [196]. For these reasons, we focus primarily on the computation of posterior marginals.

Learning with Hidden Variables

Criteria for learning in graphical models directly generalize those proposed for exponential families in Sec. 2.1.2. Let $p(\theta \mid \lambda)$ denote a prior distribution, with hyperparameters λ, on the graphical model's parameters. In the simplest case, we use the given observations y to determine a single MAP parameter estimate:

$$\hat{\theta} = \arg\max_{\theta} p(\theta \mid y, \lambda) \tag{2.89}$$

$$= \arg\max_{\theta} p(\theta \mid \lambda) \int_{\mathcal{X}} p(x, y \mid \theta)\, dx \tag{2.90}$$

This optimization is complicated by a marginalization over hidden variables x, a difficulty which did not arise with fully observed exponential families (see eq. (2.23)). Inference problems analogous to the posterior marginal computation of eq. (2.87) thus also play a role when learning with hidden variables.

In many situations, the parameters themselves are of interest, and characterizations of their posterior uncertainty are useful. Given some decomposition $\theta = \{\theta_a \mid a \in \mathcal{A}\}$ of the joint parameter space, the posterior marginal distributions of these parameters, and the corresponding hidden variables, equal

$$p(\theta_a \mid y, \lambda) = \int_{\mathcal{X}} \int_{\Theta_{\mathcal{A}\setminus a}} p(x \mid y, \theta)\, p(\theta \mid y, \lambda)\, d\theta_{\mathcal{A}\setminus a}\, dx \qquad a \in \mathcal{A} \tag{2.91}$$

$$p(x_i \mid y, \lambda) = \int_{\Theta} \int_{\mathcal{X}_{\mathcal{V}\setminus i}} p(x \mid y, \theta)\, p(\theta \mid y, \lambda)\, dx_{\mathcal{V}\setminus i}\, d\theta \qquad i \in \mathcal{V} \tag{2.92}$$

Here, θ_a typically parameterizes an individual potential function in undirected graphs, or the conditional distribution of a single variable in directed graphs. Integrating over all parameters and hidden variables, we recover the observations' *marginal likelihood:*

$$p(y \mid \lambda) = \int_{\mathcal{X}} \int_{\Theta} p(x, y \mid \theta)\, p(\theta \mid \lambda)\, d\theta\, dx \tag{2.93}$$

The marginal likelihood is central to Bayesian approaches to model selection, where integration over parameters provides a form of Occam's razor penalizing overly complex models [154, 238]. It also arises in classification problems, for which posterior probabilities are used to determine the most likely explanation of the given observations. Furthermore, maximizing eq. (2.93) with respect to hyperparameters λ provides an empirical Bayesian estimate of the prior distribution (see eq. (2.21)).

Computational Issues

Unfortunately, for many graphical models arising in practice, exact solution of these learning and inference tasks is computationally intractable. Consider, for example, the posterior marginal computation of eq. (2.87). Given N variables, each taking one of K discrete states, this expression leads to a summation containing K^{N-1} terms, which

for arbitrary graphs is NP hard [45]. Optimization of the MAP criterion (eq. (2.88)) is equally challenging [258]. For continuous $\mathcal{X}$, we face a high–dimensional integration which is usually also intractable. A notable exception occurs when all variables are jointly Gaussian, so that linear algebraic connections allow exact inference in $\mathcal{O}(N^3)$ operations [63, 118]. However, even this computation may be extremely difficult for large graphs [285, 330]. Typically, learning problems are no more tractable, since they involve integrations like those arising in inference.

In the following sections, we discuss two general frameworks which provide approximate solutions to learning and inference tasks. We begin in Sec. 2.3 by outlining variational methods which pose these computations as deterministic optimization problems. In Sec. 2.4, we then describe a complementary family of Monte Carlo methods which explore posterior distributions via efficient numerical simulations.

■ 2.3 Variational Methods and Message Passing Algorithms

In this section, we introduce a class of deterministic approximations to the problems of learning and inference posed in Sec. 2.2.5. A *variational method* [98, 161, 251, 311] begins by expressing a statistical inference task as the solution to a mathematical optimization problem. By approximating or relaxing this objective function, one can derive computationally tractable algorithms which bound or approximate the statistics of interest. Often, these algorithms inherit the graphical model's local structure, and can be implemented via the calculation of *messages* passed between neighboring nodes.

We begin our development by considering the marginal log–likelihood of the observed variables y, integrating over hidden states x and parameters θ (see eq. (2.93)). Let $q(x, \theta)$ denote some approximation to the joint posterior density $p(x, \theta \mid y, \lambda)$. Via Jensen's inequality (see eq. (2.9)), any such approximation then provides a lower bound on the marginal likelihood:

$$\begin{aligned}
\log p(y \mid \lambda) &= \log \int_\Theta \int_{\mathcal{X}} p(x, y, \theta \mid \lambda)\, dx\, d\theta \\
&= \log \int_\Theta \int_{\mathcal{X}} q(x, \theta) \frac{p(x, y, \theta \mid \lambda)}{q(x, \theta)}\, dx\, d\theta \\
&\geq \int_\Theta \int_{\mathcal{X}} q(x, \theta) \log \frac{p(x, y, \theta \mid \lambda)}{q(x, \theta)}\, dx\, d\theta \qquad (2.94) \\
&= -D(q(x, \theta) \,||\, p(x, \theta \mid y, \lambda)) + \log p(y \mid \lambda) \qquad (2.95)
\end{aligned}$$

The final equality follows by using Bayes' rule to decompose $p(x, y, \theta \mid \lambda)$. Given some family of approximating densities $\mathcal{Q}$, the best lower bound is achieved by the distribution minimizing the KL divergence from the true posterior:

$$\hat{q}(x, \theta) = \arg\min_{q \in \mathcal{Q}} D(q(x, \theta) \,||\, p(x, \theta \mid y, \lambda)) \qquad (2.96)$$

Of course, if $\mathcal{Q}$ is unrestricted the optimum is trivially $\hat{q}(x, \theta) = p(x, \theta \mid y, \lambda)$. Variational methods instead choose $\mathcal{Q}$ to be a simpler density representation for which

computations are *tractable*.

The following sections explore two classes of variational methods. In Sec. 2.3.1, we discuss *mean field* methods which use tractable families $\mathcal{Q}$ to derive a simplifying decomposition of $D(q\,||\,p)$. This representation is used to develop iterative methods guaranteed to converge to a local optimum of eq. (2.96). Sec. 2.3.2 then describes *loopy belief propagation (BP)*, which uses properties of tree–structured graphical models to motivate intuitive approximations of $\mathcal{Q}$ and $D(q\,||\,p)$. While loopy BP leads to approximations, rather than bounds, on the marginal likelihood, it is often more accurate in practice. Importantly, for either method the optimizing density $\hat{q}(x, \theta)$ provides estimates of the posterior marginal densities motivated in Sec. 2.2.5.

For simplicity, we focus on algorithms which infer conditional marginal densities in pairwise Markov random fields. However, similar variational methods may also be derived for directed [161] and factor [98, 324] graphs. In Sec. 2.3.3, we then show how the *expectation–maximization (EM)* algorithm extends inference methods to learn parameters from partially labeled data.

■ 2.3.1 Mean Field Approximations

Given some fixed, undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider a pairwise Markov random field as introduced in Sec. 2.2.2:

$$p(x \mid y) = \frac{1}{Z} \prod_{(i,j)\in\mathcal{E}} \psi_{ij}(x_i, x_j) \prod_{i\in\mathcal{V}} \psi_i(x_i, y) \tag{2.97}$$

$$= \exp\left\{ - \sum_{(i,j)\in\mathcal{E}} \phi_{ij}(x_i, x_j) - \sum_{i\in\mathcal{V}} \phi_i(x_i, y) - \Phi \right\} \tag{2.98}$$

Here, $\Phi = \log Z$ is the log partition function, and eq. (2.98) expresses the joint density via the negative logarithms of the potential functions:

$$\phi_{ij}(x_i, x_j) \triangleq -\log \psi_{ij}(x_i, x_j) \qquad \phi_i(x_i, y) \triangleq -\log \psi_i(x_i, y) \tag{2.99}$$

This representation is related to *Boltzmann's law* from statistical mechanics [337], which says that for a system in equilibrium at temperature T, a state x with energy $\phi(x)$ has probability $p(x) \propto \exp\{-\phi(x)/T\}$. For a pairwise MRF, the *energy* thus equals

$$\phi(x) = \sum_{(i,j)\in\mathcal{E}} \phi_{ij}(x_i, x_j) + \sum_{i\in\mathcal{V}} \phi_i(x_i, y) \tag{2.100}$$

We assume that the parameters θ defining the graph's potentials have been fixed by some previous modeling procedure, and do not denote them explicitly. Instead, we focus on estimating the posterior marginal densities $p(x_i \mid y)$ for all nodes $i \in \mathcal{V}$.

To develop the *mean field* method, we decompose the KL divergence (see eq. (2.96))

between an approximate posterior $q(x)$ and the target pairwise MRF as follows:

$$D(q \,||\, p) = \int_{\mathcal{X}} q(x) \log q(x) \; dx - \int_{\mathcal{X}} q(x) \log p(x \mid y) \; dx \tag{2.101}$$

$$= -H(q) + \int_{\mathcal{X}} \phi(x) \, q(x) \; dx + \Phi \tag{2.102}$$

The first term of this decomposition is the *negative entropy*, while by analogy with Boltzmann's law the second term is known as the *average energy*. Excluding the log partition function Φ, which is constant assuming fixed parameters, eq. (2.102) is sometimes called the *Gibbs free energy* [337]. Minimizing this free energy with respect to $q(x)$, we recover the true posterior of eq. (2.98). For an alternative interpretation of this relationship, in which the negative entropy arises as the conjugate dual of the log partition function, see [161, 311].

Naive Mean Field

Mean field methods are derived by choosing a restricted family of approximating densities $\mathcal{Q}$ for which minimization of eq. (2.102) is tractable. By appropriately parameterizing $\mathcal{Q}$, fixed points of this minimization also give estimates $q_i(x_i) \approx p(x_i \mid y)$ of the desired marginals. In the simplest case, the so–called *naive mean field* [98, 161, 311, 337] approximation takes $\mathcal{Q}$ to be the set of fully factorized densities:

$$q(x) = \prod_{i \in \mathcal{V}} q_i(x_i) \tag{2.103}$$

Recall that the joint entropy of a set of independent random variables equals the sum of their individual entropies [49]. Inserting the factorization of eq. (2.103) into the free energy of eq. (2.102) and simplifying, we then have

$$D(q \,||\, p) = -\sum_{i \in \mathcal{V}} H(q_i) + \sum_{i \in \mathcal{V}} \int_{\mathcal{X}_i} \phi_i(x_i, y) \, q_i(x_i) \; dx_i$$

$$\cdots + \sum_{(i,j) \in \mathcal{E}} \int_{\mathcal{X}_i} \int_{\mathcal{X}_j} \phi_{ij}(x_i, x_j) \, q_i(x_i) \, q_j(x_j) \; dx_j \, dx_i + \Phi \tag{2.104}$$

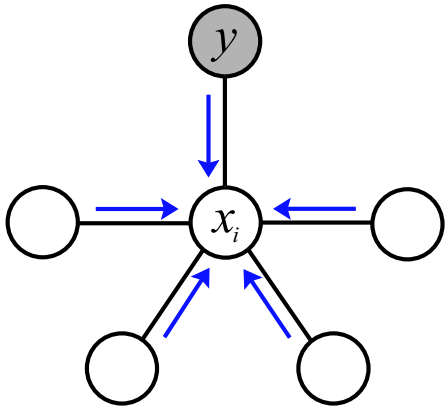
Here, we have used eq. (2.100) to decompose the average energy according to the pairwise MRF's graphical structure.

To minimize the mean field free energy of eq. (2.104), we construct a Lagrangian constraining each approximating marginal distribution to integrate to one:

$$\mathcal{L}(q, \gamma) = D(q \,||\, p) + \sum_{i \in \mathcal{V}} \gamma_i \left(1 - \int_{\mathcal{X}_i} q_i(x_i) \; dx_i \right) \tag{2.105}$$

Differentiating $\mathcal{L}(q, \gamma)$ with respect to $q_i(x_i)$ and simplifying, we find that the optimal marginals are related by the following fixed point equations:

$$\log q_i(x_i) = -\phi_i(x_i, y) - \sum_{j \in \Gamma(i)} \int_{\mathcal{X}_j} \phi_{ij}(x_i, x_j) \, q_j(x_j) \; dx_j + \bar{\gamma}_i \qquad i \in \mathcal{V} \tag{2.106}$$



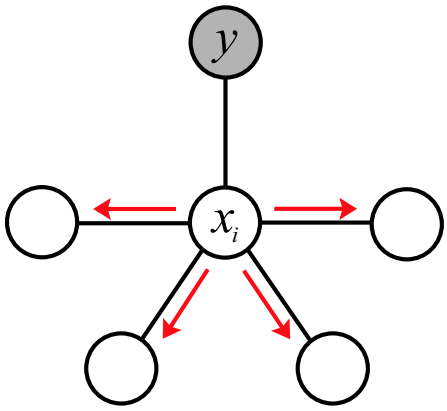


Figure 2.12. Message passing implementation of the naive mean field method. *Left:* Approximate marginal densities are determined from the normalized product of the local observation potential with messages sent from neighboring nodes. *Right:* Given an updated marginal estimate, new messages are calculated and transmitted to all neighbors.

Here, $\bar{\gamma}_i$ is a constant chosen to satisfy the marginalization constraint. Due to the pairwise relationships in the free energy of eq. (2.104), the marginal $q_i(x_i)$ at node i depends directly on the corresponding marginals at neighboring nodes $\Gamma(i)$. Thus, even though $\mathcal{Q}$ is fully factorized, the corresponding mean field solution desirably propagates information from local potentials throughout the graph.

To implement the mean field method, we must have a tractable representation for the marginal densities $q_i(x_i)$, and a corresponding algorithm for updating these marginals. Consider the following decomposition of the mean field fixed point equation (eq. (2.106)):

$$q_i(x_i) \propto \psi_i(x_i, y) \prod_{j \in \Gamma(i)} m_{ji}(x_i) \qquad i \in \mathcal{V} \tag{2.107}$$

$$m_{ji}(x_i) \propto \exp\left\{ -\int_{\mathcal{X}_j} \phi_{ij}(x_i, x_j)\, q_j(x_j)\, dx_j \right\} \qquad j \in \Gamma(i) \tag{2.108}$$

We interpret $m_{ji}(x_i)$ as a *message* sent from j to its neighboring node i. As illustrated in Fig. 2.12, mean field algorithms alternate between updating a local marginal estimate (eq. (2.107)), and using this new marginal to calculate an updated message for each neighbor (eq. (2.108)). If marginals are updated sequentially, the mean field algorithm is a form of coordinate descent which converges to a local minimum of the free energy (eq. (2.104)). Parallel updates are also possible, but do not guarantee convergence.

If $\mathcal{X}_i$ takes K discrete values, we can represent messages and marginals by K–dimensional vectors. The integration of eq. (2.108) then becomes a summation, allowing direct message computation in $\mathcal{O}(K^2)$ operations. For hidden variables defined on

continuous spaces $\mathcal{X}_i$, implementation of the mean field method is more complicated. In jointly Gaussian random fields, the integral message updates can be rewritten in terms of the posterior means [311], leading to an algorithm equivalent to the classic Gauss–Seidel iteration for linear systems [63]. More generally, for directed or undirected graphs where all potentials are defined by exponential families, the mean field marginals are finitely parameterized by the corresponding sufficient statistics [110]. From eq. (2.108), we see that messages then become exponentiated expectations of these statistics with respect to neighboring nodes. This approach can be extended to infer approximate marginal distributions for parameters θ_a (see eq. (2.91)) when all priors $p(\theta_a \mid \lambda_a)$ are conjugate [110, 331]. The VIBES software package exploits this flexibility, along with the local structure of message–passing updates, to automatically generate mean field inference code for directed graphical models [331].

While exponential families are somewhat flexible, many applications involve more complex, continuous potentials which lack sufficient statistics. In such cases, there is no finite representation for the marginal densities $q_i(x_i)$, and message updates are typically intractable. Sometimes, however, the mean field algorithm can be reasonably approximated by Monte Carlo methods which represent $q_i(x_i)$ via a collection of random samples [332]. We discuss these methods in more detail in Sec. 2.4.

Information Theoretic Interpretations

In information theory, the KL divergence $D(p \,||\, q)$ arises as a measure of the asymptotic inefficiency, or information loss [49], incurred by assuming that a stochastic process x has distribution $q(x)$ when its true distribution is $p(x \mid y)$. From this perspective, given an approximating family $\mathcal{Q}$, it seems more appropriate to minimize $D(p \,||\, q)$ over $q \in \mathcal{Q}$ rather than the "backwards" divergence $D(q \,||\, p)$ underlying mean field methods. Indeed, for fully factorized $\mathcal{Q}$ as in eq. (2.103), $D(p \,||\, q)$ has an intuitive form:

$$\begin{aligned} D(p \,||\, q) &= \int_{\mathcal{X}} p(x \mid y) \log p(x \mid y)\, dx - \int_{\mathcal{X}} p(x \mid y) \log \prod_{i \in \mathcal{V}} q_i(x_i)\, dx \\ &= -H(p) - \sum_{i \in \mathcal{V}} \int_{\mathcal{X}_i} p(x_i \mid y) \log q_i(x_i)\, dx_i \\ &= \sum_{i \in \mathcal{V}} H(p_i) - H(p) + \sum_{i \in \mathcal{V}} D(p_i \,||\, q_i) \end{aligned} \tag{2.109}$$

The first two terms, which do not depend on $q(x)$, capture the fundamental information loss incurred by *any* approximation neglecting depencies among the hidden variables. The last term is uniquely minimized by taking $q_i(x_i) = p(x_i \mid y)$, so that the true posterior marginals are exactly recovered. Interestingly, mean field methods can also be derived via a first–order Taylor series expansion of this divergence [166].

While the decomposition of eq. (2.109) shows that the marginals $p(x_i \mid y)$ provide an appropriate summary of $p(x \mid y)$, it does not provide a computational method for determining these marginals. Conversely, while mean field methods do *not* generally

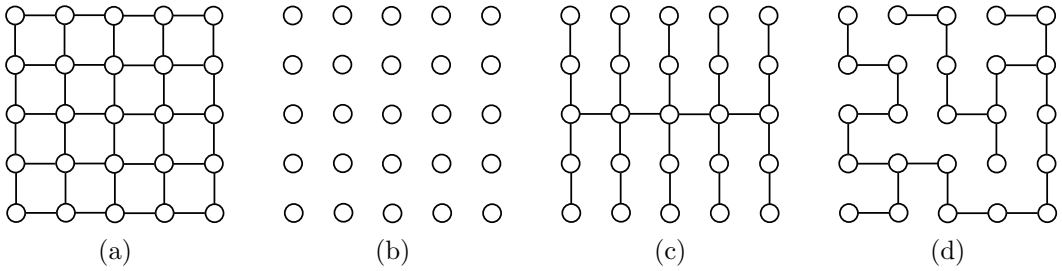


Figure 2.13. Tractable subgraphs underlying different variational methods for approximate inference. (a) Original nearest–neighbor grid (observation nodes not shown). (b) Fully factored model employed by the naive mean field method. (c) An embedded tree, as might be exploited by a structured mean field method. (d) Another of this grid's many embedded trees.

recover the true posterior marginals, minimization of $D(q \,||\, p)$ leads to tractable algorithms providing potentially useful approximations. Indeed, as we discuss in later sections, this variational approach provides a flexible framework for developing richer approximations with increased accuracy. See [161, 311] for an alternative motivation of mean field methods based on conjugate duality.

Structured Mean Field

Results from the statistical physics literature guarantee that, for certain densely connected models with sufficiently homogeneous potentials, the naive mean field approximation becomes exact as the number of variables N approaches infinity [337]. However, for sparse, irregular graphs like those considered by this thesis, its marginal estimates $q_i(x_i)$ can be extremely *overconfident*, underestimating the uncertainty of the true posterior $p(x_i \mid y)$. In addition, the mean field iteration of eqs. (2.107, 2.108) often gets stuck in local optima which differ substantially from the true posterior [98, 320]. Geometrically, these local optima arise because the set of pairwise marginals achievable via fully factorized densities is not convex [311].

Motivated by these issues, researchers have developed a variety of variational methods which extend and improve the naive mean field approximation [98, 161, 251, 311]. In particular, fully factorized approximations effectively remove all of the target graphical model's edges. However, one can also consider *structured mean field* methods based on approximating families which directly capture more of the original graph's structure (see Fig. 2.13). Optimization of these approximations is possible assuming exact inference in the chosen subgraphs is tractable [111, 252, 327, 335]. As we show in the following section, Markov chains and trees allow fast, exact recursive inference algorithms which form the basis for a variety of higher–order variational methods.

■ 2.3.2 Belief Propagation

As discussed in Sec. 2.2.5, direct solution of learning and inference problems arising in graphical models is typically intractable. Sometimes, however, global inference tasks

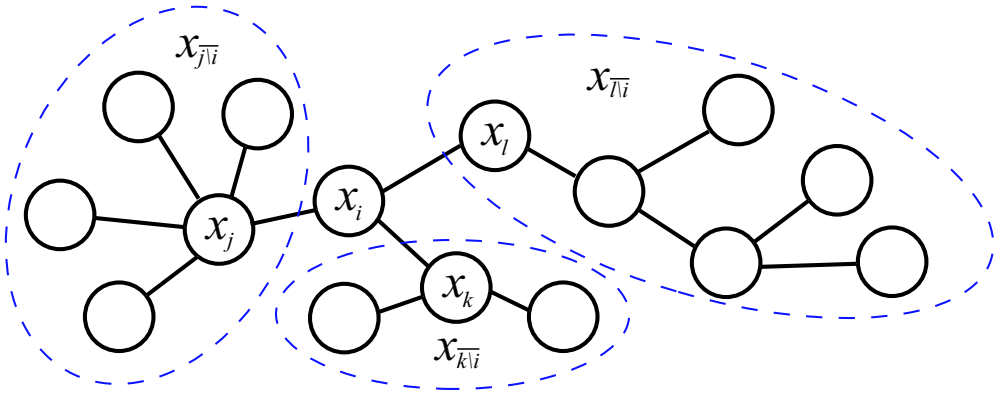


Figure 2.14. For a tree–structured graph, each node i partitions the graph into $|\Gamma(i)|$ disjoint subtrees. Conditioned on x_i, the variables $x_{\overline{j\setminus i}}$ in these subtrees are independent.

can be efficiently decomposed into a set of simpler, local computations. In particular, for tree–structured graphical models a generalization of dynamic programming known as *belief propagation (BP)* [178, 231, 255] recursively computes exact posterior marginals in linear time. In the following sections, we provide a brief derivation of BP, and discuss issues arising in its implementation. We then present a variational interpretation of BP which justifies extensions to graphs with cycles.

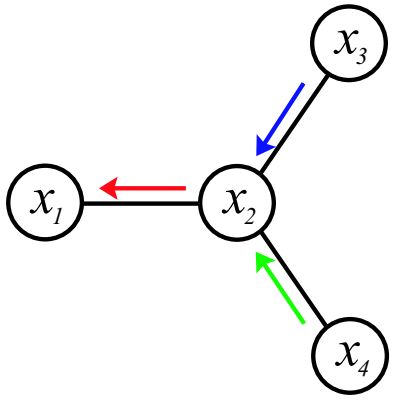
Message Passing in Trees

Consider a pairwise MRF, parameterized as in Sec. 2.3.1, whose underlying graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is tree–structured. As shown in Fig. 2.14, any node $i \in \mathcal{V}$ divides such a tree into $|\Gamma(i)|$ disjoint subsets:

$$\overline{j \setminus i} \triangleq \{j\} \cup \{k \in \mathcal{V} \mid \text{no path from } k \to j \text{ intersects } i\} \tag{2.110}$$

By the Markov properties of $\mathcal{G}$, the variables $x_{\overline{j\setminus i}}$ in these sub–trees are conditionally independent given x_i. The BP algorithm exploits this structure to recursively decompose the computation of $p(x_i \mid y)$ into a series of simpler, local calculations.

From the Hammersley–Clifford Theorem, Markov properties are expressed through the *algebraic* structure of the pairwise MRF's factorization into clique potentials. As illustrated in Fig. 2.15, tree–structured graphs allow multi–dimensional integrals (or summations) to be decomposed into a series of simpler, one–dimensional integrals. As in dynamic programming [24, 90, 303], the overall integral can then be computed via a recursion involving messages sent between neighboring nodes. This decomposition is an instance of the same *distributive law* underlying a variety of other algorithms [4, 50, 255], including the fast Fourier transform. Critically, because messages are shared among similar decompositions associated with different nodes, BP efficiently and *simultaneously* computes the desired marginals for all nodes in the graph.



$$
\begin{aligned}
p(x_1) &\propto \iiint \psi_1(x_1)\psi_{12}(x_1,x_2)\psi_2(x_2)\psi_{23}(x_2,x_3)\psi_3(x_3)\psi_{24}(x_2,x_4)\psi_4(x_4)\,dx_4\,dx_3\,dx_2 \\
&\propto \psi_1(x_1)\iiint \psi_{12}(x_1,x_2)\psi_2(x_2)\psi_{23}(x_2,x_3)\psi_3(x_3)\psi_{24}(x_2,x_4)\psi_4(x_4)\,dx_4\,dx_3\,dx_2 \\
&\propto \psi_1(x_1)\int \psi_{12}(x_1,x_2)\psi_2(x_2)\left[\iint \psi_{23}(x_2,x_3)\psi_3(x_3)\psi_{24}(x_2,x_4)\psi_4(x_4)\,dx_4\,dx_3\right]dx_2 \\
&\propto \psi_1(x_1)\int \psi_{12}(x_1,x_2)\psi_2(x_2)\underbrace{\left[\int \psi_{23}(x_2,x_3)\psi_3(x_3)\,dx_3\right]}_{m_{32}(x_2)}\cdot\underbrace{\left[\int \psi_{24}(x_2,x_4)\psi_4(x_4)\,dx_4\right]}_{m_{42}(x_2)}dx_2 \\
&\qquad\qquad m_{21}(x_1) \propto \int \psi_{12}(x_1,x_2)\psi_2(x_2)m_{32}(x_2)\,m_{42}(x_2)\;dx_2
\end{aligned}
$$

Figure 2.15. Example derivation of the BP message passing recursion through repeated application of the distributive law. Because the joint distribution $p(x)$ factorizes as a product of pairwise clique potentials, the joint integral can be decomposed via messages $m_{ji}(x_i)$ sent between neighboring nodes.

To derive the BP algorithm, we begin by considering the clique potentials corresponding to particular subsets of the full graph:

$$\Psi_{\mathcal{A}}(x_{\mathcal{A}}) \triangleq \prod_{(i,j)\in\mathcal{E}(\mathcal{A})} \psi_{ij}(x_i,x_j) \prod_{i\in\mathcal{A}} \psi_i(x_i,y) \qquad \mathcal{A}\subset\mathcal{V} \tag{2.111}$$

Here, $\mathcal{E}(\mathcal{A}) \triangleq \{(i,j)\in\mathcal{E} \mid i,j\in\mathcal{A}\}$ are the edges contained in the *node–induced subgraph* [50] corresponding to $\mathcal{A}$. Using the partitions illustrated in Fig. 2.14, we can then write the marginal distribution of any node as follows:

$$p(x_i \mid y) \propto \int_{\mathcal{X}_{\mathcal{V}\setminus i}} \psi_i(x_i,y) \prod_{j\in\Gamma(i)} \psi_{ij}(x_i,x_j)\,\Psi_{\overline{j\setminus i}}(x_{\overline{j\setminus i}})\;dx_{\mathcal{V}\setminus i} \tag{2.112}$$

$$\propto \psi_i(x_i,y) \prod_{j\in\Gamma(i)} \int_{\mathcal{X}_{\overline{j\setminus i}}} \psi_{ij}(x_i,x_j)\,\Psi_{\overline{j\setminus i}}(x_{\overline{j\setminus i}})\;dx_{\overline{j\setminus i}} \tag{2.113}$$

To verify eq. (2.112), note that it simply regroups the pairwise MRF's potentials according to Fig. 2.14. Because the variables in the subgraphs separated by node i share no potentials, the joint integral then decomposes accordingly. Interpreting the integrals in eq. (2.113) as *messages* $m_{ji}(x_i)$ sent to node i from each of its neighbors, we have

$$p(x_i \mid y) \propto \psi_i(x_i, y) \prod_{j \in \Gamma(i)} m_{ji}(x_i) \tag{2.114}$$

The message $m_{ji}(x_i)$ is a function providing the value of the corresponding integral for each possible $x_i \in \mathcal{X}_i$. Note that in a graph with cycles, node i would not necessarily disjointly partition the potentials, so the decomposition of eqs. (2.112, 2.113) is invalid.

In some applications, the joint distributions $p(x_i, x_j \mid y)$ of pairs of nodes are also of interest [324]. In tree–structured graphs, neighboring nodes $(i, j) \in \mathcal{E}$ partition the global set of clique potentials as follows:

$$p(x \mid y) \propto \psi_{ij}(x_i, x_j)\, \psi_i(x_i, y)\, \psi_j(x_j, y) \prod_{\ell \in \Gamma(i) \backslash j} \Psi_{\overline{\ell \backslash i}}(x_{\overline{\ell \backslash i}}) \prod_{k \in \Gamma(j) \backslash i} \Psi_{\overline{k \backslash j}}(x_{\overline{k \backslash j}}) \tag{2.115}$$

The corresponding subgraphs are illustrated in Fig. 2.16. Applying this decomposition as in eq. (2.112), and integrating over all variables except x_i and x_j, we then have

$$p(x_i, x_j \mid y) \propto \psi_{ij}(x_i, x_j)\, \psi_i(x_i, y)\, \psi_j(x_j, y) \prod_{\ell \in \Gamma(i) \backslash j} m_{\ell i}(x_i) \prod_{k \in \Gamma(j) \backslash i} m_{kj}(x_j) \tag{2.116}$$

The messages decomposing this pairwise marginal density are defined *identically* to those used in eq. (2.114) to compute single–node marginals.

As defined in eq. (2.113), the messages may still be complex functions of large groups of variables. To derive an efficient recursive decomposition, we consider the marginalization constraint relating the single–node and pairwise marginal distributions:

$$p(x_i \mid y) = \int_{\mathcal{X}_j} p(x_i, x_j \mid y)\, dx_j \tag{2.117}$$

$$\begin{aligned} \psi_i(x_i, y) \prod_{\ell \in \Gamma(i)} m_{\ell i}(x_i) \propto \psi_i(x_i, y) \prod_{\ell \in \Gamma(i) \backslash j} m_{\ell i}(x_i) \\ \cdots \times \int_{\mathcal{X}_j} \psi_{ij}(x_i, x_j)\, \psi_j(x_j, y) \prod_{k \in \Gamma(j) \backslash i} m_{kj}(x_j)\, dx_j \end{aligned} \tag{2.118}$$

Note that all but one of the terms on the left hand side of eq. (2.118) have identical functions of x_i on the right hand side. Cancelling these terms, as illustrated graphically in Fig. 2.16 (see [339]), the marginalization constraint is always satisfied when the remaining message $m_{ji}(x_i)$ is defined as follows:

$$m_{ji}(x_i) \propto \int_{\mathcal{X}_j} \psi_{ij}(x_i, x_j)\, \psi_j(x_j, y) \prod_{k \in \Gamma(j) \backslash i} m_{kj}(x_j)\, dx_j \tag{2.119}$$

This recursion expresses one outgoing message from node j in terms of the other $(|\Gamma(j)| - 1)$ incoming messages. At the leaves of the tree, eq. (2.119) and our initial message definition (eq. (2.113)) coincide:

$$m_{ji}(x_i) \propto \int_{\mathcal{X}_j} \psi_{ij}(x_i, x_j)\, \psi_j(x_j, y)\, dx_j \qquad \Gamma(j) = \{i\} \tag{2.120}$$

Thus, by recursively computing the messages along every edge according to eq. (2.119), we may then easily find any single–node (eq. (2.114)) or pairwise (eq. (2.116)) marginal of interest. For more formal derivations of this algorithm, see [4, 255].

Fig. 2.16 summarizes the BP message update recursion, and the corresponding message products which provide marginal densities. These posterior marginals are sometimes called *beliefs*, by analogy with expert systems developed in the artificial intelligence community [50, 178, 231]. Anticipating later extensions of BP which only provide approximate posterior marginals, we denote the beliefs for individual and pairs of nodes by $q_i(x_i)$ and $q_{ij}(x_i, x_j)$, respectively. This form of the BP algorithm is due to Shafer and Shenoy [255], who emphasized the central role of factorization in recursive inference. Several other variants of BP have been proposed [50, 158, 306], including versions adapted to directed Bayesian networks [178, 231] and factor graphs [175, 324].

To implement the BP algorithm, a *schedule* by which the messages are updated must be selected. In tree–structured graphs, an appropriate ordering of these updates requires each message to be computed only once, so that all N marginals may be determined in $\mathcal{O}(N)$ operations. One possible efficient schedule chooses some node as the *root* of the tree. This induces a partial ordering of the nodes in *scale* according to their distance from the root (see Fig. 2.6). Messages are then computed in two stages: an upward sweep proceeding from leaves to the root, followed by a downward sweep propagating information from the root throughout the graph [34, 41, 330]. Alternatively, an efficient *decentralized* schedule begins by passing outward messages from all leaf nodes. Internal message $m_{ji}(x_i)$ is then computed once node j has received messages from all $(|\Gamma(j)| - 1)$ of its other neighbors [175].

One can also consider a *parallel* form of the BP algorithm, in which every node recomputes all outgoing messages at each iteration, based on messages received from its neighbors in the previous iteration [231]. After T iterations, local marginal estimates will then optimally incorporate information from all nodes within distance T [4]. Convergence to the optimal posterior marginals occurs once the number of iterations equals the tree's diameter (at most $(N-1)$). While parallel BP updates are typically inefficient on a serial computer, they are useful in distributed implementations [100, 245].

Representing and Updating Beliefs

As with the mean field algorithm, implementations of BP require a tractable representation of the beliefs, and corresponding computational methods for the message updates of eq. (2.119). In the simplest case, where each variable x_i takes one of K discrete values ($|\mathcal{X}_i| = K$), messages and marginals can be represented by K–dimensional vectors.

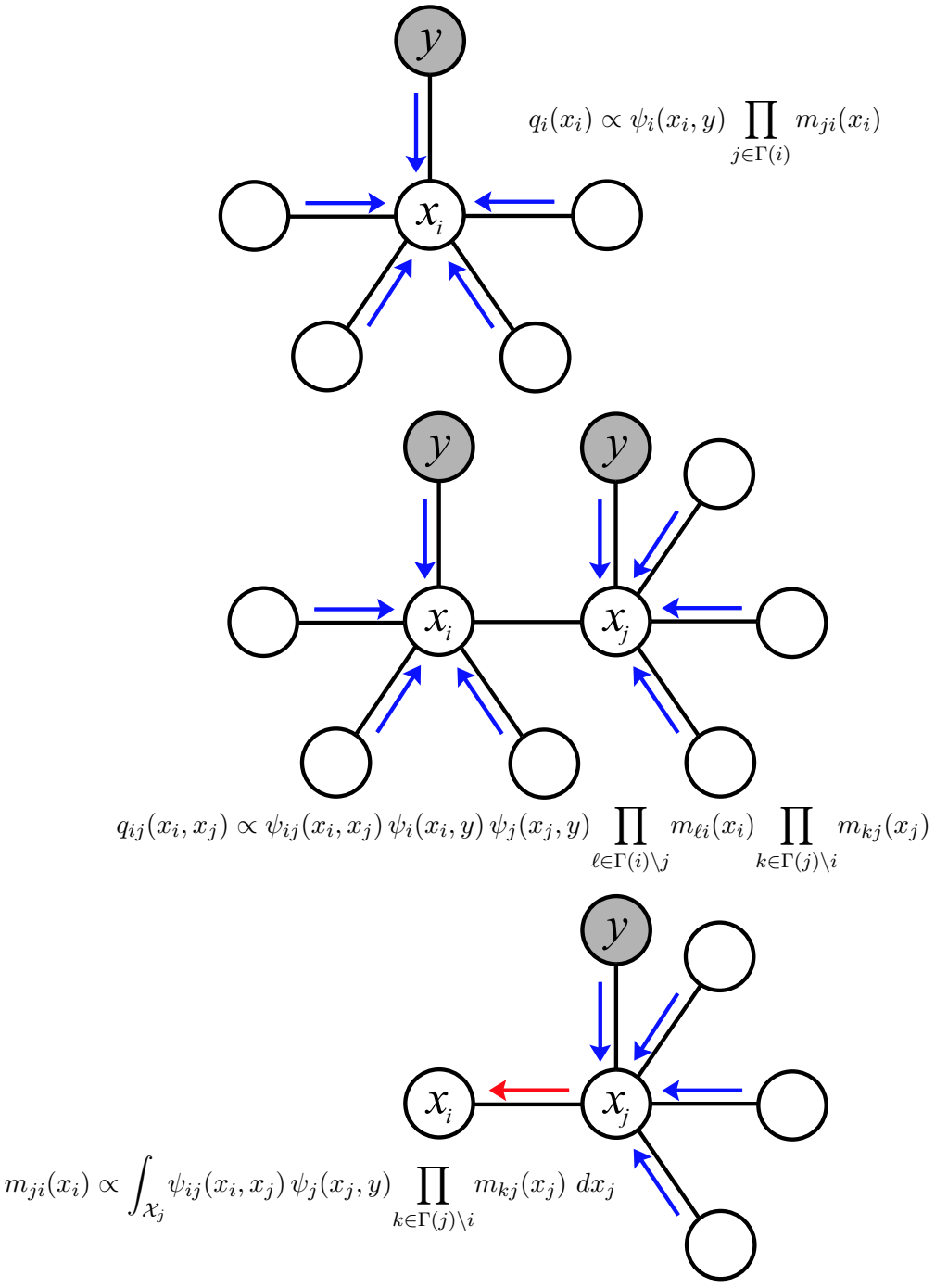


Figure 2.16. Message passing recursions underlying the BP algorithm. *Top:* Approximate marginal densities are determined from the normalized product of the local observation potential with messages sent from neighboring nodes. *Middle:* Pairwise marginal densities are derived from a similar message product. *Bottom:* A new outgoing message (red) is computed from all other incoming messages (blue).

The message update integral then becomes a matrix–vector product, which in general requires $\mathcal{O}(K^2)$ operations:

$$m_{ji}(x_i) \propto \sum_{x_j \in \mathcal{X}_j} \psi_{ij}(x_i, x_j)\, \psi_j(x_j, y) \prod_{k \in \Gamma(j) \setminus i} m_{kj}(x_j) \tag{2.121}$$

For an N node tree, BP can then compute all marginals in $\mathcal{O}(NK^2)$ operations, a dramatic savings versus the $\mathcal{O}(K^N)$ cost of brute–force summation. When the pairwise potentials $\psi_{ij}(x_i, x_j)$ are sufficiently regular, techniques such as FFTs can further reduce costs to $\mathcal{O}(K \log K)$, or $\mathcal{O}(K)$ with additional approximations [80]. By analogy with the form of eq. (2.121), BP is sometimes called the *sum–product* algorithm [175]. Specializing discrete BP to temporal HMMs (see Fig. 2.7), we recover the *forward–backward* algorithm, which is widely used for speech processing [235]. More generally, recursions equivalent to BP are often applied to multiscale discrete–state quadtree models arising in image processing [34, 330].

Inference in HMMs with continuous hidden variables has been extensively studied in the context of state space representations for dynamical systems [8, 164]. For linear systems with Gaussian dynamics and observation noise, the posterior distribution of the states is jointly Gaussian, and marginals are thus determined by their mean and covariance. In such models, BP is equivalent to *fixed–interval smoothing* algorithms which combine the Kalman filter with a complementary reverse–time recursion [8, 163, 164, 249]. These algorithms are readily generalized to any tree–structured graphical model with Gaussian potentials [41, 330]. In undirected Gaussian MRFs, BP messages are most easily updated in information form, via inverse covariance matrices [276, 321].

In contrast to the Gaussian case, continuous state space models containing non–linear or non–Gaussian interactions typically lead to message updates which lack a closed analytic form [8, 153]. Even in cases where all potentials are drawn from exponential families, the corresponding posterior densities may not have finite–dimensional sufficient statistics [326]. These difficulties have motivated a wide range of methods which approximate the true posterior by a tractable analytic form. For example, the *extended Kalman filter* fits a Gaussian posterior via a gradient–based linearization [8, 153], while the *unscented Kalman filter* uses a more accurate quadrature method [162]. More generally, given any exponential family, *expectation propagation (EP)* [135, 213] uses the moment matching conditions of Sec. 2.1.1 to approximate the beliefs produced by each message update. Note, however, that determining the sufficient statistics for such projections can itself be a challenging problem [344].

For many graphical models, the true posterior marginals are multimodal, or exhibit other features poorly approximated by standard exponential families. In some cases, a fixed K–point discretization leads to an effective histogram approximation of the true continuous beliefs [11, 80, 95, 169]. However, as K must in general grow exponentially with the dimension of $\mathcal{X}_i$, computation of the discrete messages underlying this approach can be extremely demanding. This has motivated approaches which use online message computations to *dynamically* discretize the belief space. In some cases,

deterministic rules are used to prune discretization grids [47, 48] or Gaussian mixture approximations [5, 94, 267]. Alternatively, Monte Carlo methods can be used to iteratively improve stochastic approximations to the true beliefs [9, 197, 224]. In particular, Chap. 3 describes and extends a family of *particle filters* [11, 70] which approximate messages and beliefs by a set of weighted samples.

Message Passing in Graphs with Cycles

Our earlier derivation of the BP algorithm assumed a tree–structured graph. The *junction tree* algorithm extends BP to allow exact inference in arbitrary graphs [178, 255]. Let $\mathcal{G}$ be an undirected graph (directed graphs are first moralized as in Fig. 2.4(c)). In the first of three stages, $\mathcal{G}$ is *triangulated* by adding edges so that all cycles of length four or greater contain a chord. Then, a tree is formed from the maximal cliques of the triangulated graph. Finally, a variant of BP performs exact inference on the resulting junction tree (for more details, see [4, 50, 158, 177]). The triangulation step ensures that any variables shared by two cliques are also members of other cliques along their connecting path. This *running intersection property* must be satisfied for local junction tree computations to produce globally consistent estimates. For many graphs, however, triangulation greatly increases the size of the resulting cliques. In such cases, the number of states associated with these cliques grows exponentially, and inference in the junction tree can become intractable [45].

For graphs in which exact inference is infeasible, we can still use the BP algorithm to develop improved variational methods. As mentioned in Sec. 2.3.1, one approach uses *embedded* trees (as in Fig. 2.13) to develop structured mean field bounds with increased accuracy [111, 252, 327]. In this thesis, we focus on an alternative method known as *loopy belief propagation* [231]. As summarized in Fig. 2.16, the BP algorithm proceeds entirely via a series of *local* message updates. Given a graph with cycles, loopy BP iterates a parallel form of these message updates. Remarkably, in many applications this seemingly heuristic method converges to beliefs which very closely approximate the true posterior marginals [101, 219].

The traditional dynamic programming derivation of BP provides no justification for loopy BP, other than the vague intuition that it should work well for graphs whose cycles are "long enough." In the following section, we provide a variational interpretation which places loopy BP on firmer conceptual ground. We then briefly survey known theoretical results and extensions.

Loopy BP and the Bethe Free Energy

Unsurprisingly, variational analyses of loopy BP are closely related to the Markov structure of tree–structured graphical models. The following proposition provides a local factorization which is valid for any tree–structured joint distribution, and derives a corresponding entropy decomposition.

Proposition 2.3.1. *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a tree–structured undirected graph. Any joint distribution $p(x)$ which is Markov with respect to $\mathcal{G}$ factorizes according to marginal distributions defined on the graph's nodes and edges:*

$$p(x) = \prod_{(i,j) \in \mathcal{E}} \frac{p_{ij}(x_i, x_j)}{p_i(x_i) p_j(x_j)} \prod_{i \in \mathcal{V}} p_i(x_i) \tag{2.122}$$

The joint entropy $H(p)$ then decomposes according to the graphical structure:

$$H(p) = \sum_{i \in \mathcal{V}} H(p_i) - \sum_{(i,j) \in \mathcal{E}} I(p_{ij}) \tag{2.123}$$

Proof. The factorization of eq. (2.122) is a special case of the junction tree decomposition, and can be formally verified using an induction argument [50, 177, 178]. In Markov chains, for example, it is easily derived from the standard representation via one–step transition probabilities. The entropy decomposition of eq. (2.123) then follows directly from the definitions of entropy (eq. (2.7)) and mutual information (eq. (2.11)). □

Interestingly, eq. (2.122) shows that the marginal distributions of tree–structured graphs can be inferred via a *reparameterization* operation which transforms arbitrary clique potentials (as in eq. (2.97)) to this particular canonical form [306].

Given any tree–structured undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consider a pairwise MRF $p(x \mid y)$ parameterized as in eq. (2.98). Using the entropy decomposition of eq. (2.123), the KL divergence $D(q \,||\, p)$ from any tree–structured approximation $q(x)$ equals

$$\begin{aligned} D(q \,||\, p) = -\sum_{i \in \mathcal{V}} H(q_i) + \sum_{(i,j) \in \mathcal{E}} I(q_{ij}) + \sum_{i \in \mathcal{V}} \int_{\mathcal{X}_i} \phi_i(x_i, y)\, q_i(x_i)\, dx_i \\ \cdots + \sum_{(i,j) \in \mathcal{E}} \int_{\mathcal{X}_i} \int_{\mathcal{X}_j} \phi_{ij}(x_i, x_j)\, q_{ij}(x_i, x_j)\, dx_j\, dx_i + \Phi \end{aligned} \tag{2.124}$$

This divergence depends solely on the pairwise marginals $q_{ij}(x_i, x_j)$, not on other non–local aspects of $q(x)$. To arrive at the loopy BP algorithm, we assume that the KL divergence of eq. (2.124) is *approximately* correct even for graphs with cycles. The beliefs $q_i(x_i)$ and $q_{ij}(x_i, x_j)$ are then *pseudo–marginals*, which differ from the true marginals of $p(x \mid y)$. In statistical physics, this approximation is known as the *Bethe free energy* [337, 340]. Note that for pairwise MRFs, the average energy term can be exactly written in terms of pairwise marginals. The approximation thus involves incorrectly applying the tree–based entropy of eq. (2.123) to cyclic graphs.

As with our earlier mean field derivation, loopy BP is derived by using Lagrangian methods to minimize the Bethe free energy of eq. (2.124). First, each edge $(i, j) \in \mathcal{E}$ is associated with a set of Lagrange multipliers constraining $q_{ij}(x_i, x_j)$ to consistently marginalize to $q_i(x_i)$:

$$q_i(x_i) = \int_{\mathcal{X}_j} q_{ij}(x_i, x_j)\, dx_j \qquad \text{for all } x_i \in \mathcal{X}_i \tag{2.125}$$

Adding additional normalization constraints (as in eq. (2.105)) and taking derivatives, we recover a set of fixed point equations relating Lagrange multipliers and beliefs. Finally, as derived in detail by [340], the BP equations of Fig. 2.16 are *exactly* recovered by identifying messages as particular monotonic transformations of Lagrange multipliers.

The correspondence between loopy BP and the Bethe free energy has several important implications. First, the derivation sketched above shows that loopy BP fixed points correspond to stationary points of the Bethe free energy.[4] A more refined analysis shows that *stable* BP fixed points must be local minima [132]. Furthermore, because the Bethe free energy is bounded below, every graphical model has at least one BP fixed point [338, 340].

In general, the Bethe free energy is not convex, so there may be multiple BP solutions, and convergence is not guaranteed. However, for single cycles [133, 319] or graphs with sufficiently weak potentials [133, 143, 286], BP is guaranteed to have a single, unique global fixed point. In models where loopy BP exhibits instability, message schedules which pass messages along embedded chains or trees (as in Fig. 2.13), or step–size rules which damp message updates, can improve convergence [306]. Convergence dynamics are sometimes analyzed via the *computation tree* corresponding to the chosen message schedule [143, 155, 286, 319, 321]. Alternatively, *double–loop* algorithms have been developed which directly minimize the Bethe free energy at greater computational cost [134, 290, 341].

This derivation of loopy BP approximates the variational objective of eq. (2.96) in two ways. First, as mentioned earlier, the Bethe free energy (eq. (2.124)) uses an entropy approximation which is incorrect on graphs with cycles, and thus does not strictly bound the marginal likelihood. Second, the marginalization constraints of eq. (2.125) are insufficient to ensure that the estimated pseudo–marginals $\{q_{ij}(x_i, x_j) \mid (i,j) \in \mathcal{E}\}$ correspond to some valid global $q(x)$. For example, the constraint that every joint distribution has a positive definite covariance matrix is in general *not* implied by these marginalization conditions [311, 312]. Nevertheless, in many practical applications loopy BP produces accurate, effective belief estimates [101, 219, 320].

Theoretical Guarantees and Extensions

In the artificial intelligence community, the loopy BP algorithm was originally suggested by Pearl [231] (see [219] for a historical discussion). Then in 1993, *turbo codes* were independently discovered to achieve outstanding error–correction performance by coupling two randomly interleaved convolutional codes with an iterative decoder [23]. In the following years, the equivalence of this iterative approach and loopy BP was recognized [101, 201]. Graphical representations were then used to extend turbo (or sum–product) decoding to many other code families [175], rediscovering a class of *low density parity check (LDPC)* codes proposed in Gallager's 1960 doctoral thesis [102]. Subsequent refinements have led to long block–length codes which practically achieve

[4] Note that subtleties can arise with free energy analyses in graphical models containing hard constraints, for which potentials are not strictly positive [340].

the capacity of memoryless channels [22, 42]. This performance is theoretically understood through results which show that loopy BP becomes exact as cycles become arbitrarily long, and a corresponding *density evolution* algorithm which computes capacity thresholds for random code ensembles [244].

Inspired by its successes in iterative decoding, researchers have successfully applied loopy BP to a wide range of challenging learning and inference tasks [48, 95, 99, 219, 245, 283, 336]. Concurrently, the variational interpretation provided by the Bethe free energy has led to several important theoretical results and extensions. In particular, BP can be seen as a *reparameterization* algorithm which attempts to transform the given clique potentials into the canonical form of Prop. 2.3.1 [306]. Except in certain degenerate cases, this is impossible for graphs with cycles, and loopy BP will thus *not* provide exact posterior marginals. Interestingly, however, any loopy BP fixed point is consistent with respect to *every* tree embedded in the original graph (for examples, see Fig. 2.13). This analysis can be extended to bound the error in BP's approximate marginals [305, 306]. These results are stronger than those available for the mean field method, and support the empirical observation that loopy BP is typically more accurate and less prone to local optima [320].

Additional performance guarantees are available for Gaussian MRFs. If Gaussian BP converges, several different techniques can be used to guarantee exactness of the posterior means [155, 250, 306, 321]. However, the estimated variances are incorrect because correlations due to the graph's cycles are neglected. Intuitively, when all potentials are positively correlated or *attractive*, these variance estimates are over–confident [321]. Furthermore, convergence is guaranteed for a wide class of *walk–summable* models [155], or equivalently any graph whose pairwise potentials are normalizable.

More generally, variational interpretations of BP have led to the development of several extensions with improved accuracy. For example, the Bethe entropy of eq. (2.124) can be seen as the first terms of an expansion based on the Möbius inversion formula [125, 248]. Higher order terms directly account for relationships among larger groups of variables. Exploiting this, a *region graph* framework has been proposed which leads to better entropy approximations, and a corresponding family of *generalized belief propagation* algorithms [202, 338, 339, 340]. This approach generalizes the *Kikuchi free energies* [337] developed in the statistical physics community. The *expectation propagation* algorithm [135, 212, 213] provides a closely related method of incorporating higher–order dependencies (see [305] and [323] for unifying comparisons). In addition, a family of robust *reweighted* belief propagation algorithms have been derived from convex upper bounds on the log partition function [307, 310, 328, 329].

Finally, we note that the distributive structure underlying the BP algorithm can be generalized to any commutative semiring [4, 50, 255, 303]. In particular, a *max–product* variant of BP generalizes the Viterbi algorithm [90, 235] to efficiently compute optimal MAP estimates in tree–structured graphs [175, 231]. For graphs with cycles, there are some guarantees on max–product's accuracy [308, 322], and a reweighted extension can sometimes assure an optimal MAP solution [172, 309]. See [311] for an introduction

emphasizing variational interpretations of these methods.

■ 2.3.3 The Expectation Maximization Algorithm

In this section, we consider the MAP parameter estimation criterion motivated in Sec. 2.2.5. Given a model with parameters θ, and prior distribution $p(\theta \mid \lambda)$, we seek

$$\hat{\theta} = \arg\max_{\theta} p(\theta \mid y, \lambda) = \arg\max_{\theta} p(\theta \mid \lambda) \int_{\mathcal{X}} p(x, y \mid \theta)\ dx \tag{2.126}$$

As before, y are observations and x are latent variables. The *Expectation Maximization (EM)* algorithm [65] is an iterative parameter estimation scheme which tractably handles hidden or *missing* data x. We derive EM using the previously introduced variational framework, and discuss its application to learning in graphical models. For other introductions to the EM algorithm, see [98, 107, 161, 225].

As with other variational methods, the EM algorithm uses a distribution $q(x)$ over hidden variables to bound an otherwise intractable integral. Using Bayes' rule to expand the posterior distribution of eq. (2.126), we have

$$\log p(\theta \mid y, \lambda) = \log \int_{\mathcal{X}} p(x, y \mid \theta)\ dx + \log p(\theta \mid \lambda) - \log p(y \mid \lambda) \tag{2.127}$$

$$\geq \int_{\mathcal{X}} q(x) \log \frac{p(x, y \mid \theta)}{q(x)}\ dx + \log p(\theta \mid \lambda) - \log p(y \mid \lambda) \tag{2.128}$$

Here, we have applied Jensen's inequality as in our earlier variational bound on the marginal likelihood (eq. (2.94)). Regrouping terms and neglecting the final normalization constant, which does not depend on θ, we arrive at the following functional:

$$\mathcal{L}(q, \theta) = H(q) + \int_{\mathcal{X}} q(x) \log p(x, y \mid \theta)\ dx + \log p(\theta \mid \lambda) \tag{2.129}$$

Comparing to eq. (2.102), we see that $\mathcal{L}(q, \theta)$ equals a negative free energy [225] plus another term incorporating prior knowledge about the unknown parameters [107].

As in [225, 227], we derive the EM algorithm as a coordinate ascent iteration on $\mathcal{L}(q, \theta)$. In the expectation or *E–step*, the parameters θ are fixed and the optimal variational distribution $q(x)$ is determined. Then in the maximization or *M–step*, the lower bound defined by $q(x)$ is maximized with respect to the parameters:

$$q^{(t)} = \arg\max_{q} \mathcal{L}(q, \theta^{(t-1)}) \tag{2.130}$$

$$\theta^{(t)} = \arg\max_{\theta} \mathcal{L}(q^{(t)}, \theta) \tag{2.131}$$

It can be shown that the posterior probability of eq. (2.126) increases monotonically with each EM iteration, converging to some local maximum [65, 107, 225]. In the following sections, we discuss the implementation of these steps in greater detail.

Expectation Step

Fixing the parameters to some value $\theta^{(t-1)}$, provided either by the previous M–step or an initialization $\theta^{(0)}$, the E–step objective of eq. (2.130) becomes

$$q^{(t)} = \arg\max_q \left[H(q) + \int_{\mathcal{X}} q(x) \log p(x, y \mid \theta^{(t-1)}) \, dx \right] \tag{2.132}$$

Note the similarity of this equation to the variational objective underlying the mean field method (eq. (2.102)). Adding a Lagrange multiplier ensuring that $q(x)$ is properly normalized (as in eq. (2.105)) and taking derivatives, it is easily shown that

$$q^{(t)}(x) = p(x \mid y, \theta^{(t-1)}) \tag{2.133}$$

See [225] for a detailed derivation. We see that the E–step simply infers the posterior distribution of the hidden variables given the current parameters.

If $p(x, y \mid \theta)$ defines an exponential family, the expected values of that family's statistics are sufficient for the subsequent M–step. In graphical models, the E–step thus reduces to the problem of computing the posterior marginal distribution of each hidden variable (see Sec. 2.2.5). The variational derivation of the EM algorithm also justifies *incremental* E–steps, in which the expectations of only some variables are updated at each iteration [225]. In graphs where exact inference is intractable, mean field methods are commonly used to further bound the log–likelihood [161, 311, 331]. It is also tempting to use higher order variational methods, such as loopy BP, as approximate E–steps [98, 136]. In such cases, however, $\mathcal{L}(q, \theta)$ no longer strictly bounds the true posterior probability [311], and the resulting iteration may be unstable or inaccurate.

Maximization Step

Given the posterior distribution $q^{(t)}(x)$ determined in the previous E–step, the M–step objective of eq. (2.131) equals

$$\theta^{(t)} = \arg\max_\theta \left[\log p(\theta \mid \lambda) + \int_{\mathcal{X}} q^{(t)}(x) \log p(x, y \mid \theta) \, dx \right] \tag{2.134}$$

Up to an additive constant independent of θ, the likelihood term in eq. (2.134) equals $-D\big(q^{(t)} \,||\, p_\theta\big)$. If θ parameterizes an exponential family and the prior distribution is uninformative, Prop. 2.1.2 then shows that $\theta^{(t)}$ should be chosen to match the appropriate sufficient statistics of $q^{(t)}$. Similarly, conjugate priors $p(\theta \mid \lambda)$ are easily handled by appropriately biasing these statistics (see Prop. 2.1.4). More generally, partial M–steps can be used which increase, but do not maximize, the current likelihood bound [107, 225].

In directed Bayesian networks, the M–step can often be computed in closed form [37, 50, 98, 128]. Consider the following directed factorization:

$$p(x \mid \theta) = \prod_{i \in \mathcal{V}} p\big(x_i \mid x_{\Gamma(i)}, \theta_i\big) \tag{2.135}$$

Here, θ_i parameterizes the transition distribution for the i^{th} node, and we have not explicitly indicated which nodes correspond to observations y. If each transition is assigned a *meta independent* [50, 59] prior $p(\theta_i \mid \lambda_i)$, the objective of eq. (2.134) equals

$$\theta^{(t)} = \arg\max_{\theta} \sum_{i \in \mathcal{V}} \iint q^{(t)}(x_i, x_{\Gamma(i)}) \log p\big(x_i \mid x_{\Gamma(i)}, \theta_i\big)\, dx_i\, dx_{\Gamma(i)} + \log p(\theta_i \mid \lambda_i) \tag{2.136}$$

The parameters associated with different nodes are thus *decoupled*, and can be estimated independently. This optimization is similarly tractable for many models in which parameters are shared among multiple transition densities [235].

In undirected graphical models, parameter estimation is more challenging. Consider a factor graph parameterized as in eq. (2.68), and assume for simplicity that the reference measure $\nu(x) = 1$. Then, if each clique potential is assigned a meta independent prior $p(\theta_f \mid \lambda_f)$, the M–step objective equals

$$\theta^{(t)} = \arg\max_{\theta} \sum_{f \in \mathcal{F}} \left[\sum_{a \in \mathcal{A}_f} \theta_{fa} \int q^{(t)}(x_f) \phi_{fa}(x_f)\, dx_f + \log p(\theta_f \mid \lambda_f) \right] - \Phi(\theta) \tag{2.137}$$

In contrast with eq. (2.136), the log partition function $\Phi(\theta)$ induces non–local dependencies among the parameters. When the corresponding graph is decomposable or triangulated, junction tree representations can be used to efficiently estimate parameters [59, 177]. Otherwise, computationally demanding numerical methods are required, often implemented via one of several *iterative scaling* algorithms [53, 56, 62, 177, 227, 268, 290]. A recently proposed family of convex *upper* bounds on the log partition function can be used for approximate undirected parameter estimation [307, 310].

■ 2.4 Monte Carlo Methods

By using random samples to simulate probabilistic models, *Monte Carlo methods* [9, 107, 192] provide complementary solutions to the learning and inference tasks described in Sec. 2.2.5. In contrast with variational approaches, they are guaranteed to give arbitrarily precise estimates with sufficient computation. In practice, however, care must be taken to design efficient algorithms so that reliable, accurate estimates can be obtained at a tractable computational cost.

Let $p(x)$ denote some target density with sample space $\mathcal{X}$. Many inference tasks, including the calculation of marginal densities and sufficient statistics, can be expressed as the expected value $\mathbb{E}_p[f(x)]$ of an appropriately chosen function [9, 192]. Suppose that $p(x)$ is difficult to analyze explicitly, but that L independent samples $\{x^{(\ell)}\}_{\ell=1}^{L}$ are

available. The desired statistic can then be approximated as follows:

$$\mathbb{E}_p[f(x)] = \int_{\mathcal{X}} f(x)p(x)\,dx \tag{2.138}$$

$$\approx \frac{1}{L}\sum_{\ell=1}^{L} f(x^{(\ell)}) = \mathbb{E}_{\tilde{p}}[f(x)] \tag{2.139}$$

Here, $\tilde{p}(x)$ is the empirical density (see eq. (2.13)) corresponding to the L samples, as illustrated in Fig. 2.17(a). This estimate is unbiased, and converges to $\mathbb{E}_p[f(x)]$ almost surely as $L \to \infty$. Furthermore, its error is asymptotically Gaussian, with variance determined by $\mathbb{E}_p\left[f^2(x)\right]$ rather than the dimensionality of the sample space [9].

In graphical models, exact samples can be drawn from the posterior distribution $p(x \mid y)$ using a variant of the junction tree algorithm (see Sec. 2.3.2). First, some clique is chosen as the tree's root, and a sample is drawn from its corresponding marginal. The values of neighboring cliques are then recursively sampled from the appropriate conditional densities [50]. For many graphs, however, the junction tree's cliques are too large, and exact sampling is intractable. The following sections describe several Monte Carlo methods which allow approximate samples to be drawn more efficiently.

■ 2.4.1 Importance Sampling

Importance sampling provides an alternative to direct Monte Carlo approximation in cases where sampling from $p(x)$ is difficult. We assume that it is possible to evaluate $p(x) = \bar{p}(x)/Z$ up to some normalization constant Z. Let $q(x)$ denote a *proposal distribution* which is absolutely continuous with respect to $p(x)$, so that $p(\bar{x}) = 0$ whenever $q(\bar{x}) = 0$. The expectation of eq. (2.138) can then be rewritten as follows:

$$\mathbb{E}_p[f(x)] = \frac{\int_{\mathcal{X}} f(x)w(x)q(x)\,dx}{\int_{\mathcal{X}} w(x)q(x)\,dx} \qquad w(x) = \frac{\bar{p}(x)}{q(x)} \tag{2.140}$$

The denominator of eq. (2.140) implicitly defines the unknown normalization constant via the *weight function* $w(x)$. Given L independent samples $\{x^{(\ell)}\}_{\ell=1}^{L}$ from the proposal density $q(x)$, we approximate this expectation as

$$\mathbb{E}_p[f(x)] \approx \sum_{\ell=1}^{L} w^{(\ell)} f(x^{(\ell)}) \qquad w^{(\ell)} \triangleq \frac{w(x^{(\ell)})}{\sum_{m=1}^{L} w(x^{(m)})} \tag{2.141}$$

Importance sampling thus estimates the target expectation via a collection of *weighted* samples $\{(x^{(\ell)}, w^{(\ell)})\}_{\ell=1}^{L}$ from the proposal density $q(x)$. Under mild assumptions, this estimate is asymptotically consistent [9], and its variance is smallest when the proposal density $q(x) \propto |f(x)|p(x)$. Fig. 2.17 illustrates weighted samples drawn from two different importance approximations to a bimodal target distribution.

The practical effectiveness of importance sampling critically depends on the chosen importance density. When $q(x)$ assigns low probability to likely regions of the target sample space, importance estimates can be extremely inaccurate. For example,

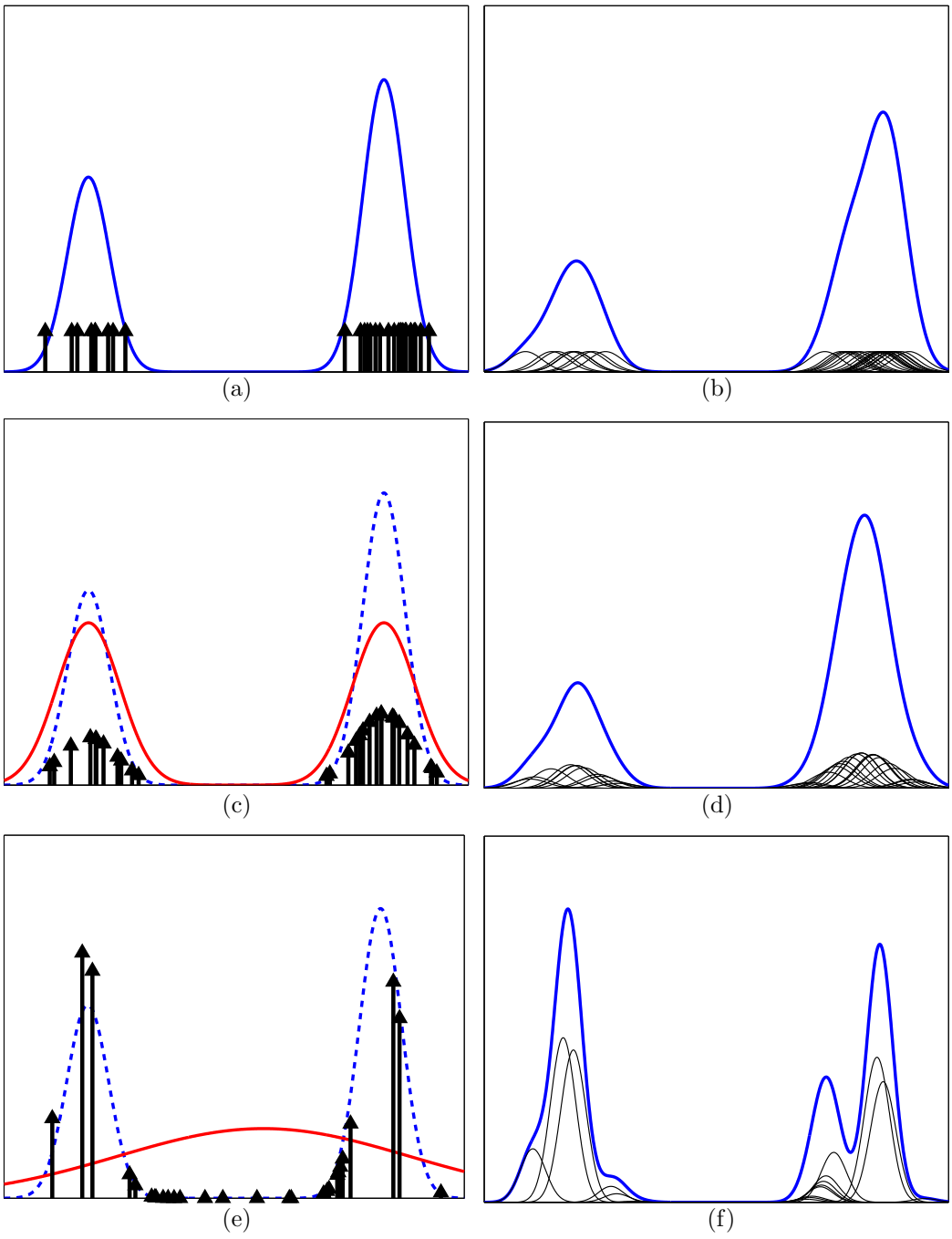


Figure 2.17. Monte Carlo estimates based on 30 samples (arrows) from one–dimensional proposal distributions (left column), and corresponding kernel density estimates (right column) constructed via likelihood cross–validation. (a) Target density (solid), and unweighted direct samples. (b) Kernel density (thick blue line) estimated from Gaussian kernels (thin black lines). (c) A mixture proposal distribution (solid) closely matched to the target density (dashed), and importance weighted samples. (d) Kernel density estimated from weighted Gaussian kernels. (e) A Gaussian proposal distribution (solid) with mean and variance matching the target density (dashed), and weighted samples. (f) Kernel density with artifacts from the Gaussian proposal's widely varying importance weights.

the poorly matched proposal distribution of Fig. 2.17(e) causes many samples to have negligible weight, greatly reducing the *effective* sample size. Heavy–tailed proposal distributions, which are more dispersed than the target density, typically provide greater robustness [107, 192]. For high–dimensional problems, however, designing good proposals is extremely challenging, since even minor discrepancies can produce widely varying importance weights. In graphical models, importance sampling is thus typically used as a building block within more sophisticated Monte Carlo methods.

■ 2.4.2 Kernel Density Estimation

In some applications of Monte Carlo methods, an explicit estimate $\hat{p}(x)$ of the target density $p(x)$ is desired, rather than a summary statistic as in eq. (2.138). *Nonparametric* density estimators avoid choosing a particular form for $\hat{p}(x)$, and allow the complexity of the estimated density to grow as more samples are observed. Given L independent samples $\{x^{(\ell)}\}_{\ell=1}^{L}$, the corresponding *kernel* or *Parzen window* density estimate [230, 263] can be written as follows:

$$\hat{p}(x) = \sum_{\ell=1}^{L} w^{(\ell)} \mathcal{N}(x; x^{(\ell)}, \Lambda) \tag{2.142}$$

This estimator uses a Gaussian kernel function to smooth the raw sample set, intuitively placing more probability mass in regions with many samples. Other kernel functions may also be considered [263], but we focus on the Gaussian case. If these samples are drawn from the target density $p(x)$, the weights are set uniformly to $w^{(\ell)} = 1/L$. More generally, they could come from an importance sampling scheme [220] as in eq. (2.141).

The kernel density estimate of eq. (2.142) depends on the bandwidth or covariance Λ of the Gaussian kernel function. There is an extensive literature on methods for automatic bandwidth selection [263]. For example, the simple "rule of thumb" method combines a robust covariance estimate with an asymptotic formula which assumes the target density is Gaussian. While fast to compute, it often oversmooths multimodal distributions. In such cases, more sophisticated cross–validation schemes can improve performance [263]. Fig. 2.17 illustrates kernel density estimates constructed from three different proposal distributions, with bandwidth automatically selected via likelihood cross–validation. Note that inaccurate importance densities produce less reliable density estimators (compare Fig. 2.17(d) and Fig. 2.17(f)).

■ 2.4.3 Gibbs Sampling

We now describe a family of iterative, *Markov chain Monte Carlo (MCMC)* methods which draw samples from an otherwise intractable target density $p(x)$. Starting from some initial global configuration $x^{(0)} \in \mathcal{X}$, subsequent states are determined via a first–order Markov process:

$$x^{(t)} \sim q(x \mid x^{(t-1)}) \qquad\qquad t = 1, 2, \ldots \tag{2.143}$$

The transition distribution $q(\cdot \mid \cdot)$ is designed so that the resulting Markov chain is irreducible and aperiodic, with $p(x)$ as its unique equilibrium distribution [9]. Thus, after many iterations T the state will be approximately distributed as $x^{(T)} \sim p(x)$, providing a sample from the desired target density.

The *Metropolis–Hastings* algorithm [9, 107] provides a flexible, general framework for constructing Markov chains with a desired equilibrium distribution $p(x)$. In this section, we describe the *Gibbs sampler* [106, 108, 196], a special case that is particularly well suited to state spaces with internal structure. Let $x = (x_1, \ldots, x_N)$ denote a decomposition of the joint sample space into N variables. Gibbs samplers assume that it is tractable to sample from the conditional distribution of one of these variables given the other $(N-1)$. At iteration t, a particular variable $i(t)$ is selected for resampling, and the rest are held constant:

$$x_i^{(t)} \sim p(x_i \mid x_j^{(t-1)}, j \neq i) \qquad i = i(t) \tag{2.144}$$

$$x_j^{(t)} = x_j^{(t-1)} \qquad j \neq i(t) \tag{2.145}$$

If these sampling updates are iterated so that all variables are resampled infinitely often, mild conditions ensure $x^{(t)}$ will converge to a sample from $p(x)$ as $t \to \infty$ [9, 108, 186]. Randomly permuting the order in which variables are resampled, rather than repeating a single fixed order, often improves the rate of convergence [246].

Although there exist polynomial bounds on the time required for some MCMC methods to *mix* to the target equilibrium distribution [9, 186], it can be difficult to guarantee or diagnose convergence in high–dimensional models [192]. In practice, it is often useful to run the sampler from several random initializations, and compare problem–dependent summary statistics. If slow mixing is observed, one can consider *blocked Gibbs samplers* which, rather than sampling individual variables, jointly resample small groups of variables which are thought to be strongly correlated [9, 185, 246].

For some models, Gibbs samplers are best implemented via *auxiliary variable* methods [9]. These algorithms are based on a joint distribution $p(x, z)$ which is designed to marginalize to the target density $p(x)$. In the simplest case, auxiliary variables z are chosen so that the following conditional densities are tractable:

$$x^{(t)} \sim p(x \mid z^{(t-1)}) \tag{2.146}$$

$$z^{(t)} \sim p(z \mid x^{(t)}) \tag{2.147}$$

More generally, eq. (2.146) may be replaced by several Gibbs sampling steps as in eqs. (2.144, 2.145). Any joint sample $(x^{(T)}, z^{(T)})$ from the resulting Markov chain then also provides an approximate sample $x^{(T)}$ from the target density of interest. Some auxiliary variable methods, such as the hybrid Monte Carlo algorithm [9, 107, 192], are designed to improve the convergence rate of the resulting Markov chain. Alternatively, auxiliary variable methods sometimes lead to tractable Gibbs samplers for models in which direct conditional densities lack simple forms [222]. Several algorithms developed in this thesis exploit this technique.

Sampling in Graphical Models

The Gibbs sampler's use of partitioned state spaces is ideally suited for inference in graphical models [98, 108, 196, 231]. For example, consider a pairwise MRF $p(x \mid y)$ parameterized as in eq. (2.97). By the Markov properties discussed in Sec. 2.2.2, the posterior distribution of x_i depends only on the values at neighboring nodes:

$$p\big(x_i \mid x_{\mathcal{V}\setminus i}, y\big) = p\big(x_i \mid x_{\Gamma(i)}, y\big) \propto \psi_i(x_i, y) \prod_{j \in \Gamma(i)} \psi_{ij}(x_i, x_j) \tag{2.148}$$

When the clique potentials are drawn from exponential families, it is typically easy to sample from this conditional density. Iterating such resampling as in eqs. (2.144, 2.145), we obtain a Gibbs sampler providing Monte Carlo estimates of the posterior marginals motivated in Sec. 2.2.5. Alternatively, the related *simulated annealing* method [9, 108] can be used to search for approximate MAP estimates.

Gibbs sampling is also used to estimate posterior distributions for model parameters θ (see eq. (2.91)). First, hidden variables are sampled given fixed parameters as in eq. (2.148). Then, conditioned on these hidden variables, conjugate priors $p(\theta \mid \lambda)$ typically allow individual parameters to be tractably resampled [37, 50, 106, 128]. Alternating between sampling $x^{(t)} \sim p\big(x \mid \theta^{(t-1)}, y\big)$ and $\theta^{(t)} \sim p\big(\theta \mid x^{(t)}, y, \lambda\big)$, we can estimate statistics of the joint posterior $p(x, \theta \mid y, \lambda)$. The BUGS software package uses this method to do Bayesian learning and inference in directed graphical models [115].

Gibbs Sampling for Finite Mixtures

To illustrate the Gibbs sampler, we consider a K–component exponential family mixture model, as introduced in Sec. 2.2.4 (see Fig. 2.9). While the data $x = \{x_i\}_{i=1}^N$ are directly observed, the latent cluster $z_i \in \{1, \dots, K\}$ associated with each data point is unknown. The simplest mixture model Gibbs sampler thus alternates between sampling cluster indicators $z = \{z_i\}_{i=1}^N$, mixture weights π, and cluster parameters $\{\theta_k\}_{k=1}^K$. We assume the hyperparameters α and λ are set to fixed, known constants.

Given fixed cluster weights and parameters, the indicator variables are conditionally independent. Let $z_{\setminus i}$ denote the set of all cluster assignments excluding z_i. Applying Bayes' rule to the generative model of eq. (2.79), we then have

$$p(z_i = k \mid z_{\setminus i}, x, \pi, \theta_1, \dots, \theta_K) = p(z_i = k \mid x_i, \pi, \theta_1, \dots, \theta_K) \tag{2.149}$$

$$\propto \pi_k f(x_i \mid \theta_k) \tag{2.150}$$

Here, the simplification of eq. (2.149) follows from the Markov properties of the directed graph in Fig. 2.9. By evaluating the likelihood of x_i with respect to each current cluster, we may thus resample z_i in $\mathcal{O}(K)$ operations.

As discussed in detail by [96], the mixture weights π and parameters $\{\theta_k\}_{k=1}^K$ are mutually independent conditioned on the indicator variables z:

$$p(\pi, \theta_1, \dots, \theta_K \mid z, x, \alpha, \lambda) = p(\pi \mid z, \alpha) \prod_{k=1}^{K} p(\theta_k \mid \{x_i \mid z_i = k\}, \lambda) \tag{2.151}$$

Given mixture weights $\pi^{(t-1)}$ and cluster parameters $\{\theta_k^{(t-1)}\}_{k=1}^K$ from the previous iteration, sample a new set of mixture parameters as follows:

1. Independently assign each of the N data points x_i to one of the K clusters by sampling the indicator variables $z = \{z_i\}_{i=1}^N$ from the following multinomial distributions:
$$z_i^{(t)} \sim \frac{1}{Z_i} \sum_{k=1}^K \pi_k^{(t-1)} f(x_i \mid \theta_k^{(t-1)})\, \delta(z_i, k) \qquad Z_i = \sum_{k=1}^K \pi_k^{(t-1)} f(x_i \mid \theta_k^{(t-1)})$$
2. Sample new mixture weights according to the following Dirichlet distribution:
$$\pi^{(t)} \sim \mathrm{Dir}(N_1 + \alpha/K, \ldots, N_K + \alpha/K) \qquad N_k = \sum_{i=1}^N \delta(z_i^{(t)}, k)$$
3. For each of the K clusters, independently sample new parameters from the conditional distribution implied by those observations currently assigned to that cluster:
$$\theta_k^{(t)} \sim p(\theta_k \mid \{x_i \mid z_i^{(t)} = k\}, \lambda)$$
When λ defines a conjugate prior, this posterior distribution is given by Prop. 2.1.4.

Algorithm 2.1. Direct Gibbs sampler for a K component exponential family mixture model, as defined in Fig. 2.9. Each iteration resamples the cluster assignments for all N observations $x = \{x_i\}_{i=1}^N$ once, and uses these updated assignments to choose new mixture parameters.

Assuming α is the precision of a symmetric Dirichlet prior, the posterior distribution of the mixture weights π is also Dirichlet (see eq. (2.45)), with hyperparameters determined by the number of observations N_k currently assigned to each cluster:

$$p(\pi \mid z, \alpha) = \mathrm{Dir}(N_1 + \alpha/K, \ldots, N_K + \alpha/K) \qquad N_k = \sum_{i=1}^N \delta(z_i, k) \tag{2.152}$$

Standard methods may then be used to sample new cluster weights [107]. Intuitively, eq. (2.151) shows that the posterior distribution of the k^{th} cluster's parameters θ_k depends only on those observations currently assigned to it. If λ parameterizes a conjugate prior, Prop. 2.1.4 provides a closed form for this posterior. For example, when clusters are Gaussian, $\theta_k = (\mu_k, \Lambda_k)$ follows a normal–inverse–Wishart density (see Sec. 2.1.4).

Algorithm 2.1 summarizes the Gibbs sampler implied by these conditional distributions. We initialize the mixture parameters according to their priors $\pi^{(0)} \sim \mathrm{Dir}(\alpha)$, $\theta_k^{(0)} \sim H(\lambda)$. At each iteration, $\mathcal{O}(NK)$ operations are needed to resample all N indicator variables. Note that because these indicators are mutually independent given known parameters, the order of this resampling is unimportant. To allow fast parameter resampling, we cache sufficient statistics (as in Thm. 2.1.2) of the data assigned to each cluster, and recursively update these statistics as assignments change.

In Fig. 2.18, we use the Gibbs sampler of Alg. 2.1 to fit a mixture of $K = 4$ two–dimensional Gaussians to $N = 300$ observations. Each Gaussian cluster is assigned a weakly informative normal–inverse–Wishart prior, so that the posterior distribution of $\theta_k = (\mu_k, \Lambda_k)$ can be determined as described in Sec. 2.1.4. The columns of Fig. 2.18

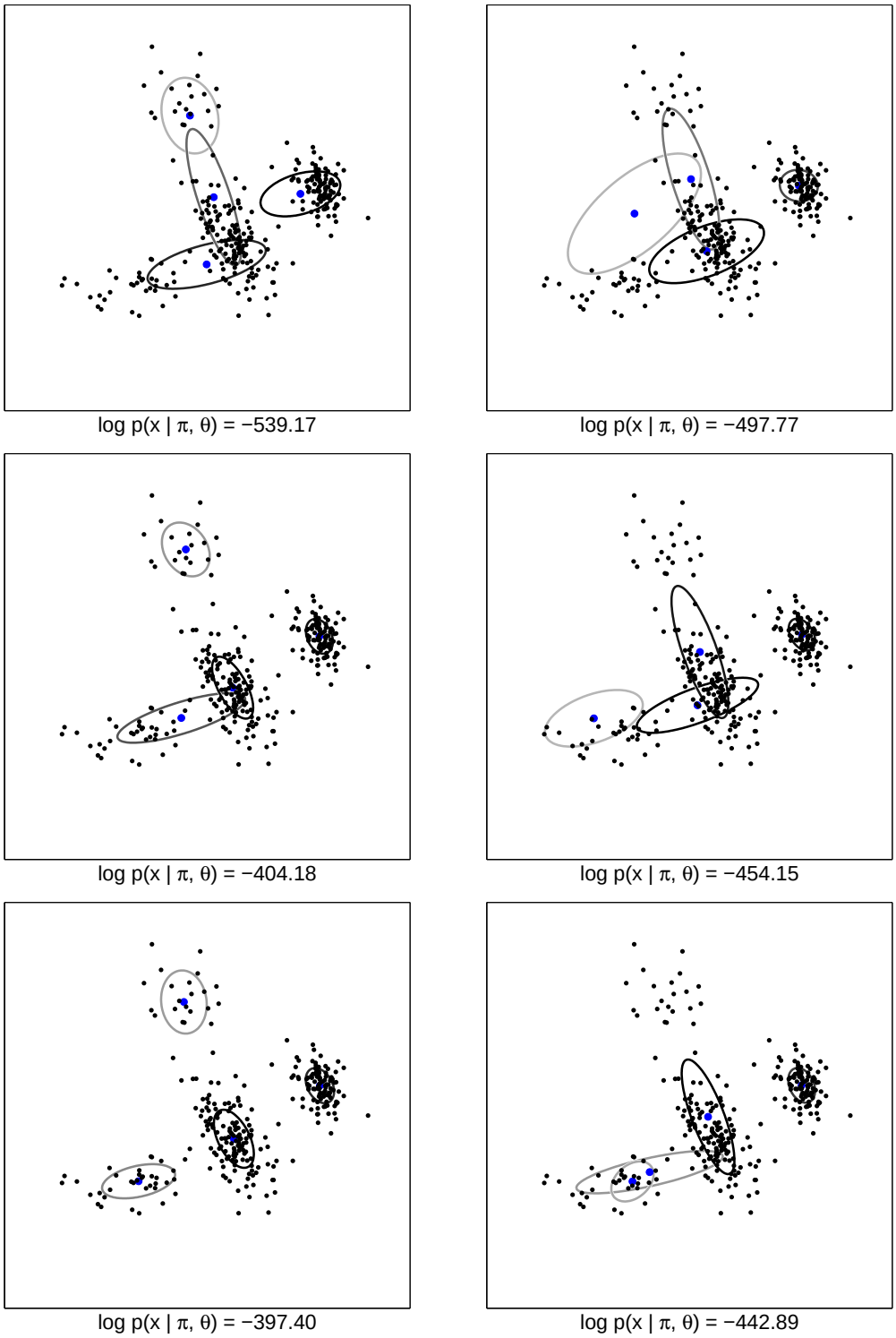


Figure 2.18. Learning a mixture of $K = 4$ Gaussians using the Gibbs sampler of Alg. 2.1. Columns show the current parameters after T=2 (top), T=10 (middle), and T=50 (bottom) iterations from two random initializations. Each plot is labeled by the current data log–likelihood.

compare two different random initializations. Because we use vague priors, the data log–likelihood provides a reasonable convergence measure:

$$\log p(x \mid \pi, \theta_1, \ldots, \theta_K) = \sum_{i=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k f(x_i \mid \theta_k) \right) \tag{2.153}$$

We see that the Gibbs sampler effectively implements a random walk, which gradually moves towards parameters with higher posterior probability. Although the induced Markov chain may converge quickly (left column), it sometimes remains trapped in locally optimal regions of the parameter space for many iterations (right column). Fig. 2.20 compares this behavior to a more sophisticated Rao–Blackwellized sampler developed in the following section.

■ 2.4.4 Rao–Blackwellized Sampling Schemes

In models which impose structured dependencies on multiple latent variables, we can often construct tractable Monte Carlo procedures which improve on the basic estimator of eq. (2.139). Let $p(x, z)$ denote a target distribution on two random variables $x \in \mathcal{X}$, $z \in \mathcal{Z}$. Given L independent samples $\{(x^{(\ell)}, z^{(\ell)})\}_{\ell=1}^{L}$ from this joint distribution, the simplest approximation of a statistic $f(x, z)$ equals

$$\mathbb{E}_p[f(x,z)] = \int_{\mathcal{Z}} \int_{\mathcal{X}} f(x,z) p(x,z)\, dx\, dz \tag{2.154}$$

$$\approx \frac{1}{L} \sum_{\ell=1}^{L} f(x^{(\ell)}, z^{(\ell)}) = \mathbb{E}_{\tilde{p}}[f(x,z)] \tag{2.155}$$

Suppose, however, that the conditional density $p(x \mid z)$ has a tractable analytic form. In this case, we can consider the following alternative estimator:

$$\mathbb{E}_p[f(x,z)] = \int_{\mathcal{Z}} \int_{\mathcal{X}} f(x,z) p(x \mid z)\, p(z)\, dx\, dz \tag{2.156}$$

$$= \int_{\mathcal{Z}} \left[\int_{\mathcal{X}} f(x,z) p(x \mid z)\ dx \right] p(z)\, dz \tag{2.157}$$

$$\approx \frac{1}{L} \sum_{\ell=1}^{L} \int_{\mathcal{X}} f(x, z^{(\ell)}) p(x \mid z^{(\ell)})\ dx = \mathbb{E}_{\tilde{p}}[\mathbb{E}_p[f(x,z) \mid z]] \tag{2.158}$$

The estimators of eqs. (2.155) and (2.158) are both unbiased, and converge to $\mathbb{E}_p[f(x, z)]$ almost surely as $L \to \infty$. Intuitively, however, the marginalized estimate of eq. (2.158) should be more reliable [9, 39, 106], because the underlying sample space $\mathcal{Z}$ is smaller than the original space $\mathcal{X} \times \mathcal{Z}$.

In classical statistics, the *Rao–Blackwell Theorem* [167, 242] establishes the importance of sufficient statistics in parameter estimation. In particular, it allows *minimum variance unbiased estimators* to be designed by conditioning simpler estimators with

respect to appropriate statistics. The Rao–Blackwell Theorem is derived from the following relationship between conditional and unconditional variance, which is also more broadly applicable.

Theorem 2.4.1 (Rao-Blackwell). *Let x and z be dependent random variables, and $f(x, z)$ a scalar statistic. Consider the marginalized statistic $\mathbb{E}_x[f(x, z) \mid z]$, which is a function solely of z. The unconditional variance $\mathrm{Var}_{xz}[f(x, z)]$ is then related to the variance of the marginalized statistic as follows:*

$$\mathrm{Var}_{xz}[f(x,z)] = \mathrm{Var}_z[\mathbb{E}_x[f(x,z) \mid z]] + \mathbb{E}_z[\mathrm{Var}_x[f(x,z) \mid z]] \tag{2.159}$$

$$\geq \mathrm{Var}_z[\mathbb{E}_x[f(x,z) \mid z]] \tag{2.160}$$

Proof. Using the iterated expectations [229, 242] induced by the conditional factorization $p(x, z) = p(x \mid z)\, p(z)$, the unconditional variance of $f(x, z)$ equals

$$\begin{aligned}\mathrm{Var}_{xz}[f(x,z)] &= \mathbb{E}_{xz}\left[f(x,z)^2\right] - \mathbb{E}_{xz}[f(x,z)]^2 \\ &= \mathbb{E}_z\left[\mathbb{E}_x\left[f(x,z)^2 \mid z\right]\right] - \mathbb{E}_z[\mathbb{E}_x[f(x,z) \mid z]]^2\end{aligned}$$

Subtracting and adding $\mathbb{E}_z[\mathbb{E}_x[f(x, z) \mid z]^2]$ and regrouping terms, we may then verify eq. (2.159). Equation (2.160) follows from the non–negativity of $\mathrm{Var}_x[f(x, z) \mid z]$. □

As established by eq. (2.160), analytic marginalization of some variables from a joint distribution *always* reduces the variance of later estimates. Applying this result, the so–called *Rao–Blackwellized* Monte Carlo estimator [9, 39] of eq. (2.158) has lower variance than the direct estimator of eq. (2.155). Intuitively, eq. (2.159) shows that marginalization of x is most useful when the average conditional variance of x is large.

Rao–Blackwellization also plays an important role in other, more sophisticated Monte Carlo methods. In particular, the variance inequality of Thm. 2.4.1 can be generalized to bound the variance of marginalized importance estimators (see Sec. 2.4.1). As we discuss in Chap. 3, this approach has been used to design Rao–Blackwellized improvements of standard particle filters [71, 73]. Similarly, Rao–Blackwellization may dramatically improve the efficiency and accuracy of Gibbs samplers [39, 106, 185]. In particular, for hierarchical models based on conjugate priors, Prop. 2.1.4 can often be used to integrate over latent parameters in closed form. Importantly, the variance reduction guaranteed by Thm. 2.4.1 generalizes to estimates based on the correlated samples produced by a Gibbs sampler [185].

Rao–Blackwellized Gibbs Sampling for Finite Mixtures

To illustrate the design of Rao–Blackwellized samplers, we revisit the mixture model Gibbs sampler summarized in Alg. 2.1. Given fixed cluster indicators z, we show that conjugate priors allow mixture weights π and parameters $\{\theta_k\}_{k=1}^K$ to be analytically marginalized. We may then directly determine the predictive distribution of z_i given the other cluster assignments $z_{\setminus i}$, and construct a more efficient sampler.

Consider the K–component exponential family mixture model of Fig. 2.9, and assume $H(\lambda)$ specifies a conjugate prior for the clusters θ_k. Integrating over the parameters π and $\{\theta_k\}_{k=1}^K$, the model's Markov structure implies the following factorization:

$$p(z_i \mid z_{\setminus i}, x, \alpha, \lambda) \propto p(z_i \mid z_{\setminus i}, \alpha)\, p(x_i \mid z, x_{\setminus i}, \lambda) \tag{2.161}$$

The first term arises from the marginalization of the mixture weights π. Because these weights have a symmetric Dirichlet prior, this predictive distribution is given by eq. (2.46) of Sec. 2.1.3, so that

$$p(z_i = k \mid z_{\setminus i}, \alpha) = \frac{N_k^{-i} + \alpha/K}{N - 1 + \alpha} \qquad N_k^{-i} = \sum_{j \neq i} \delta(z_j, k) \tag{2.162}$$

Note that N_k^{-i} counts the number of observations currently assigned to the k^{th} cluster excluding x_i, the datum whose assignment z_i is being resampled. Similarly, the likelihood term of eq. (2.161) depends on the current assignments $z_{\setminus i}$ as follows:

$$p(x_i \mid z_i = k, z_{\setminus i}, x_{\setminus i}, \lambda) = p(x_i \mid \{x_j \mid z_j = k, j \neq i\}, \lambda) \tag{2.163}$$

For each of the K possible values of z_i, eq. (2.163) equals the predictive likelihood (as in eq. (2.19)) of x_i given the other data currently assigned to that cluster. Because $H(\lambda)$ is conjugate to θ_k, these likelihoods can be analytically determined from Prop. 2.1.4. For example, Gaussian clusters lead to Student–t predictive distributions (see Sec. 2.1.4), which can usually be approximated by the moment–matched Gaussian of eq. (2.64).

Algorithm 2.2 provides one possible Rao–Blackwellized Gibbs sampler based on these predictive distributions. As with the direct Gibbs sampler of Alg. 2.1, $\mathcal{O}(NK)$ operations are required to resample N cluster assignments. To improve the Markov chain's convergence rate, each iteration resamples indicator variables in a different, randomly chosen order [246]. Fast predictive likelihood evaluation is achieved by caching the sufficient statistics $\boldsymbol{\phi}(x)$ (as in Thm. 2.1.2) associated with each cluster. When an observation x_i is reassigned, these statistics are easily updated by subtracting $\boldsymbol{\phi}(x_i)$ from the previous cluster $z_i^{(t-1)}$, and adding $\boldsymbol{\phi}(x_i)$ to the newly chosen cluster $z_i^{(t)}$. We initialize the sampler by sequentially choosing $z_i^{(0)}$ conditioned on $\{z_1^{(0)}, \ldots, z_{i-1}^{(0)}\}$.

In Fig. 2.19, we use the Rao–Blackwellized Gibbs sampler of Alg. 2.2 to fit a mixture of $K = 4$ two–dimensional Gaussians to $N = 300$ observations. Compared to the direct Gibbs sampler of Alg. 2.1 (tested on identical data in Fig. 2.18), the Rao–Blackwellized sampler has less random variation from iteration to iteration. Fig. 2.20 compares the data log–likelihoods (eq. (2.153)) produced by these two algorithms from 100 different random initializations. Typically, the Rao–Blackwellized sampler much more rapidly reaches parameters with high posterior probability. Intuitively, this happens because marginalized, predictive likelihoods implicitly update the model's parameters after every indicator reassignment, rather than once per iteration as in Alg. 2.1. However, the two samplers have similar worst case performance, and may occasionally remain in local

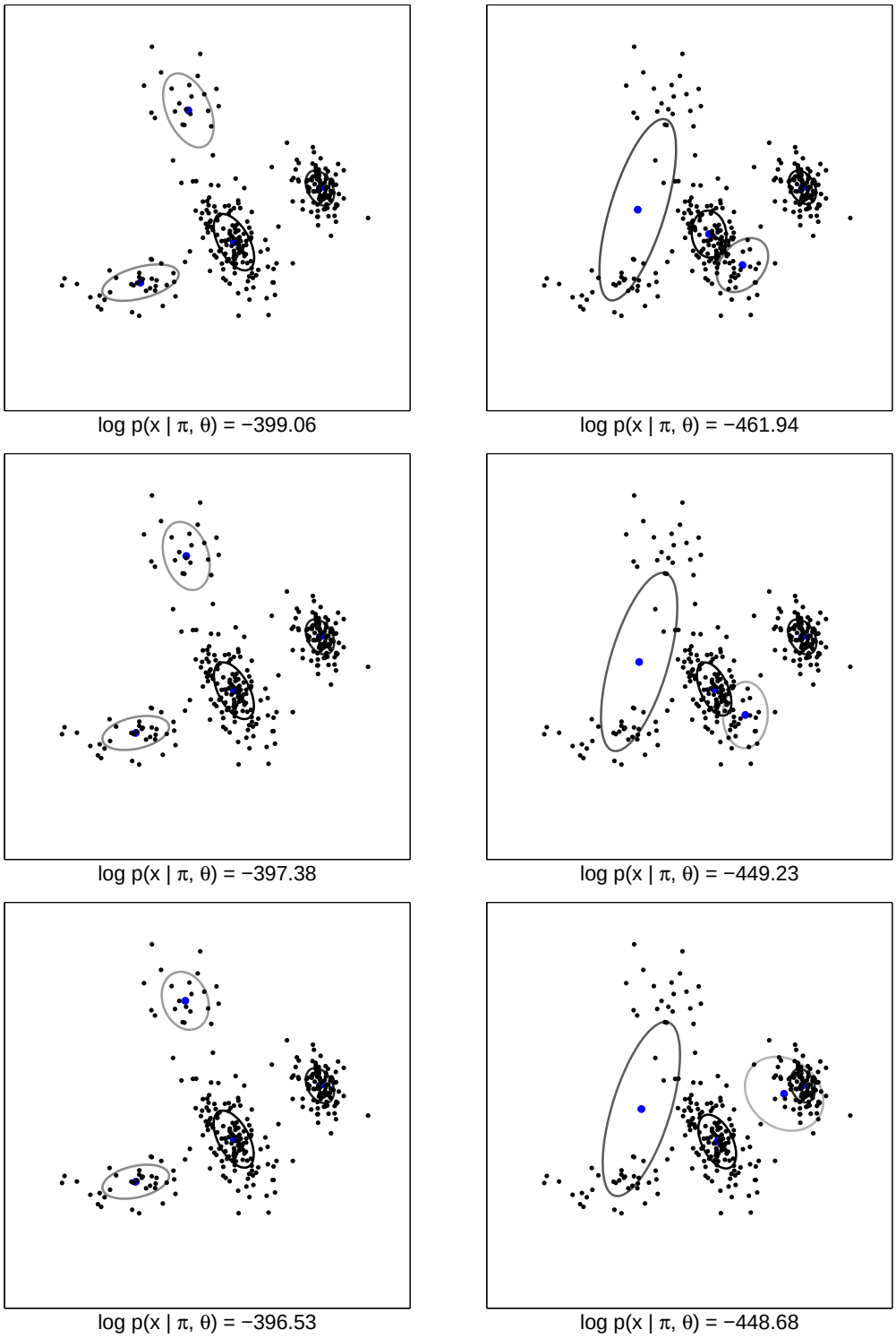


Figure 2.19. Learning a mixture of $K = 4$ Gaussians using the Rao–Blackwellized Gibbs sampler of Alg. 2.2. Columns show the current parameters after T=2 (top), T=10 (middle), and T=50 (bottom) iterations from two random initializations. Each plot is labeled by the current data log–likelihood.

Given previous cluster assignments $z^{(t-1)}$, sequentially sample new assignments as follows:

1. Sample a random permutation $\tau(\cdot)$ of the integers $\{1, \ldots, N\}$.
2. Set $z = z^{(t-1)}$. For each $i \in \{\tau(1), \ldots, \tau(N)\}$, sequentially resample z_i as follows:
   (a) For each of the K clusters, determine the predictive likelihood
   $$f_k(x_i) = p(x_i \mid \{x_j \mid z_j = k, j \neq i\}, \lambda)$$
   This likelihood can be computed from cached sufficient statistics via Prop. 2.1.4.
   (b) Sample a new cluster assignment z_i from the following multinomial distribution:
   $$z_i \sim \frac{1}{Z_i} \sum_{k=1}^{K} (N_k^{-i} + \alpha/K) f_k(x_i) \delta(z_i, k) \qquad Z_i = \sum_{k=1}^{K} (N_k^{-i} + \alpha/K) f_k(x_i)$$
   N_k^{-i} is the number of other observations assigned to cluster k (see eq. (2.162)).
   (c) Update cached sufficient statistics to reflect the assignment of x_i to cluster z_i.
3. Set $z^{(t)} = z$. Optionally, mixture parameters may be sampled via steps 2–3 of Alg. 2.1.

Algorithm 2.2. Rao–Blackwellized Gibbs sampler for a K component exponential family mixture model, as defined in Fig. 2.9. Each iteration sequentially resamples the cluster assignments for all N observations $x = \{x_i\}_{i=1}^N$ in a different random order. Mixture parameters are integrated out of the sampling recursion using cached sufficient statistics of the parameters assigned to each cluster.

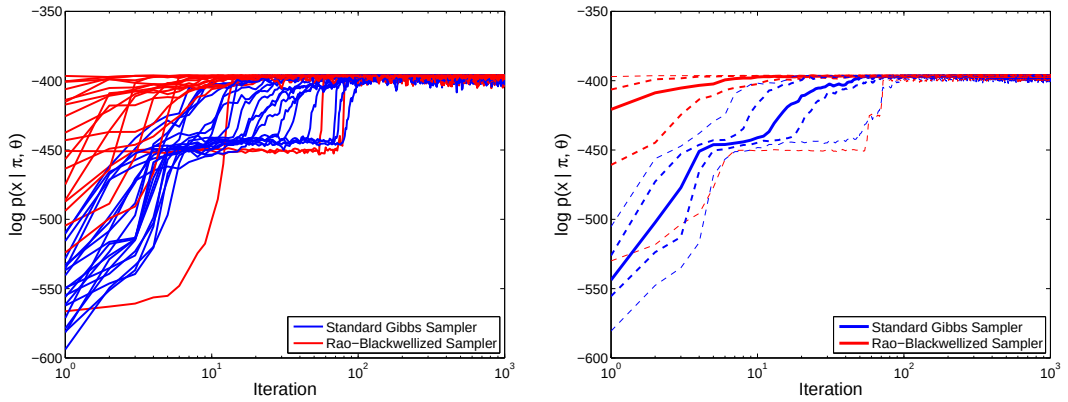


Figure 2.20. Comparison of standard (Alg. 2.1, dark blue) and Rao–Blackwellized (Alg. 2.2, light red) Gibbs samplers for a mixture of $K = 4$ two–dimensional Gaussians. We compare data log–likelihoods at each of 1000 iterations for the single $N = 300$ point dataset of Figs. 2.18 and 2.19. *Left:* Log–likelihood sequences for 20 different random initializations of each algorithm. *Right:* From 100 different random initializations, we show the median (solid), 0.25 and 0.75 quantiles (thick dashed), and 0.05 and 0.95 quantiles (thin dashed) of the resulting log–likelihood sequences. The Rao–Blackwellized sampler has superior typical performance, but occasionally remains trapped in local optima for many iterations.

optima for many iterations (see right columns of Figs. 2.18 and 2.19). These results suggest that while Rao–Blackwellization can usefully accelerate mixing, convergence diagnostics are still important.

2.5 Dirichlet Processes

It is often difficult to find simple parametric models which adequately describe complex, realistic datasets. *Nonparametric* statistical methods avoid assuming restricted functional forms, and thus allow the complexity and accuracy of the inferred model to grow as more data is observed. Strictly speaking, nonparametric models are rarely free of parameters, since they must have a concrete, computationally tractable representation. In Bayesian statistics, nonparametric methods typically learn distributions on function spaces, and thus effectively involve infinitely many parameters [21, 109, 113, 160, 216, 238]. Complexity is controlled via appropriate prior distributions, so that small datasets produce simple predictions, while additional observations induce richer posteriors.

To motivate nonparametric statistical methods, consider De Finetti's representation (see Thm. 2.2.2) of N infinitely exchangeable random variables:

$$p(x_1, x_2, \ldots, x_N) = \int_\Theta p(\theta) \prod_{i=1}^{N} p(x_i \mid \theta)\, d\theta \tag{2.164}$$

In general, this decomposition is only guaranteed when Θ is an infinite–dimensional space of probability measures. Many Bayesian nonparametric methods thus involve families of computationally tractable distributions on probability measures [84]. In particular, the *Dirichlet process* [28, 83, 254] provides a distribution on distributions with many attractive properties, and is widely used in practice [60, 76, 105, 160, 289].

The following sections establish several representations of the Dirichlet process, which characterize its behavior and lead to computationally tractable learning and inference algorithms. We then show that Dirichlet processes provide an elegant alternative to parametric model selection, and discuss extensions to structured, hierarchical models. For other introductions to Dirichlet processes, see [84, 109, 113, 160, 216, 289, 313].

2.5.1 Stochastic Processes on Probability Measures

Because nonparametric methods use stochastic processes to model infinite–dimensional spaces, they are often *implicitly* characterized by the distributions they induce on certain finite statistics. For example, *Gaussian processes* provide a distribution over real–valued functions which is widely used for non–linear regression and classification [1, 109, 229, 253]. By definition, a function $f : \mathcal{X} \to \mathbb{R}$ is distributed according to a Gaussian process if and only if $p(f(x_1), \ldots, f(x_N))$, the density of that function's values at any N points $x_i \in \mathcal{X}$, is jointly Gaussian. This allows Gaussian processes to be tractably parameterized by a mean function and a *covariance kernel* specifying the correlations within any finite point set.

While Gaussian processes define distributions on random functions, a *Dirichlet process* defines a distribution on random probability measures, or equivalently non–negative functions which integrate to one. Let Θ denote a measurable space, as in the parameter space underlying De Finetti's mixture representation (eq. (2.164)). A

Dirichlet process is then parameterized by a *base measure* H on Θ, and a positive scalar *concentration parameter* α. Analogously to the Gaussian case, Dirichlet processes are characterized by the distributions they induce on finite measurable *partitions* (see Fig. 2.21) of the parameter space.

Theorem 2.5.1. *Let H be a probability distribution on a measurable space Θ, and α a positive scalar. Consider a finite partition $(T_1, \ldots, T_K)$ of Θ:*

$$\bigcup_{k=1}^{K} T_k = \Theta \qquad T_k \cap T_\ell = \emptyset \qquad k \neq \ell \tag{2.165}$$

A random probability distribution G on Θ is drawn from a Dirichlet process if its measure on every finite partition follows a Dirichlet distribution:

$$(G(T_1), \ldots, G(T_K)) \sim \mathrm{Dir}(\alpha H(T_1), \ldots, \alpha H(T_K)) \tag{2.166}$$

For any base measure H and concentration parameter α, there exists a unique stochastic process satisfying these conditions, which we denote by $\mathrm{DP}(\alpha, H)$.

Proof. For a characterization as in eq. (2.166) to be valid, probabilities must appropriately add when a partition's cells are combined. The aggregation property of the finite Dirichlet distribution (see eq. (2.43)) is one way to guarantee this. Ferguson originally established the existence of the Dirichlet process via Kolmogorov's consistency conditions [83]. Later, Sethuraman provided a simpler, constructive definition [254] which we describe in Sec. 2.5.2. □

Fig. 2.21 illustrates the consistency requirements relating different partitions of the parameter space Θ. Combining eqs. (2.40) and (2.166), for any region $T \subset \Theta$ the expected measure of a random sample from a Dirichlet process equals

$$\mathbb{E}[G(T)] = H(T) \qquad G \sim \mathrm{DP}(\alpha, H) \tag{2.167}$$

The base measure H thus specifies the mean of $\mathrm{DP}(\alpha, H)$. As we show in Sec. 2.5.3, the concentration parameter α is similar to the precision of a finite Dirichlet distribution, and determines the average deviation of samples from the base measure.

Posterior Measures and Conjugacy

Let $G \sim \mathrm{DP}(\alpha, H)$ be sampled from a Dirichlet process, and $\bar{\theta} \sim G$ be a sample from that distribution. Consider the finite Dirichlet distribution induced by a fixed partition, as in eq. (2.166). Via the conjugacy of the Dirichlet distribution (see eq. (2.45)), the posterior distribution is also Dirichlet:

$$p\big((G(T_1), \ldots, G(T_K)) \mid \bar{\theta} \in T_k\big) = \mathrm{Dir}(\alpha H(T_1), \ldots, \alpha H(T_k) + 1, \ldots, \alpha H(T_K)) \tag{2.168}$$

Note that the observation $\bar{\theta}$ only affects the Dirichlet parameter of the unique, arbitrarily small cell T_k containing it [160]. Formalizing this analysis, it can be shown that the posterior distribution has a Dirac point mass $\delta_{\bar{\theta}}$ centered on each observation.

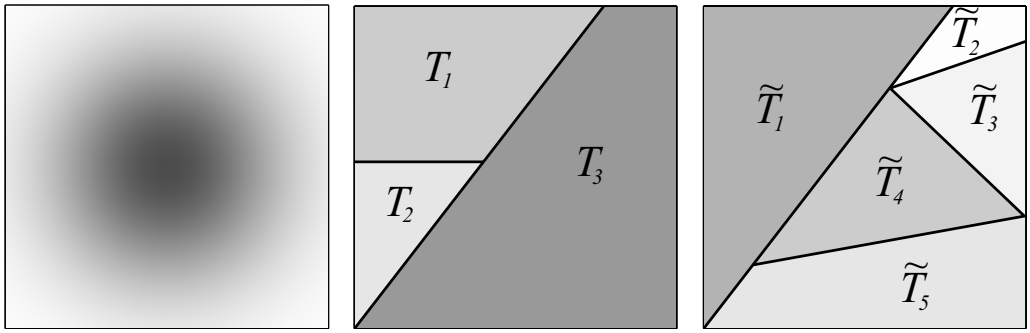


Figure 2.21. Dirichlet processes induce Dirichlet distributions on every finite, measurable partition. *Left:* An example base measure H on a bounded, two–dimensional space Θ (darker regions have higher probability). *Center:* A partition with $K = 3$ cells. The weight that a random measure $G \sim \mathrm{DP}(\alpha, H)$ assigns to these cells follows a Dirichlet distribution (see eq. (2.166)). We shade each cell T_k according to its mean $\mathbb{E}[G(T_k)] = H(T_k)$. *Right:* Another partition with $K = 5$ cells. The consistency of G implies, for example, that $(G(T_1) + G(T_2))$ and $G(\widetilde{T}_1)$ follow identical beta distributions.

Proposition 2.5.1. *Let $G \sim \mathrm{DP}(\alpha, H)$ be a random measure distributed according to a Dirichlet process. Given N independent observations $\bar{\theta}_i \sim G$, the posterior measure also follows a Dirichlet process:*

$$p\big(G \mid \bar{\theta}_1, \ldots, \bar{\theta}_N, \alpha, H\big) = \mathrm{DP}\left(\alpha + N, \frac{1}{\alpha + N}\left(\alpha H + \sum_{i=1}^{N} \delta_{\bar{\theta}_i}\right)\right) \tag{2.169}$$

Proof. As shown by Ferguson [83], this result follows directly from the conjugate form of finite Dirichlet posterior distributions (see eq. (2.45)). See Sethuraman [254] for an alternative proof. □

There are interesting similarities between eq. (2.169) and the general form of conjugate priors for exponential families (see Prop. 2.1.4). The Dirichlet process effectively defines a conjugate prior for distributions on arbitrary measurable spaces. In some contexts, the concentration parameter α can then be seen as expressing confidence in the base measure H via the size of a pseudo–dataset (see [113] for further discussion).

Neutral and Tailfree Processes

The conjugacy of Prop. 2.5.1, which leads to tractable computational methods discussed later, provides one practical motivation for the Dirichlet process. In this section, we show that Dirichlet processes are also characterized by certain conditional independencies. These properties reveal both strengths and weaknesses of the Dirichlet process, and have motivated several other families of stochastic processes.

Let G be a random probability measure on a parameter space Θ. The distribution

of G is *neutral* [69, 84] with respect to a finite partition $(T_1, \ldots, T_K)$ of Θ if and only if

$$G(T_k) \qquad \text{is independent of} \qquad \left\{ \frac{G(T_\ell)}{1 - G(T_k)} \;\middle|\; \ell \neq k \right\} \tag{2.170}$$

given that $G(T_k) < 1$. Thus, for a neutral process, the probability mass assigned to some cell T_k affects the weight of other cells only through the normalization constraint. The relative probabilities assigned to those cells are independent random variables. As shown by the following theorem, the Dirichlet process is characterized by its neutrality with respect to *every* measurable partition.

Theorem 2.5.2. *Consider a distribution $\mathcal{P}$ on probability measures G for some space Θ. Assume that $\mathcal{P}$ assigns positive probability to more than one measure G, and that with probability one samples $G \sim \mathcal{P}$ assign positive measure to at least three distinct points $\theta \in \Theta$. The following conditions are then equivalent:*

(i) $\mathcal{P} = \mathrm{DP}(\alpha, H)$ is a Dirichlet process for some base measure H on Θ.

(ii) $\mathcal{P}$ is neutral with respect to every finite, measurable partition of Θ.

(iii) For every measurable $T \subset \Theta$, and any N observations $\bar{\theta}_i \sim G$, the posterior distribution $p\big(G(T) \mid \bar{\theta}_1, \ldots, \bar{\theta}_N\big)$ depends only on the number of observations that fall within T (and not their particular locations).

Proof. This result was derived by Doksum and Fabius via related characterizations of the finite Dirichlet distribution. See [69, 84] for a more precise description of degenerate cases, and additional references. □

This theorem shows that Dirichlet processes effectively ignore the topology of the parameter space Θ. Observations provide information only about those cells which directly contain them. In addition, an observation near the boundary of a cell provides the same amount of information as an observation in its center. Thus, while neutrality simplifies the structure of posterior distributions, it also limits the expressiveness of the corresponding prior.

For problems in which $\Theta = \mathbb{R}$ is the real line, a less restrictive form of neutrality has been proposed. A random cumulative distribution $F(t) = \Pr[\theta \leq t]$ is *neutral to the right (NTR)* [69, 84] if, for any K times $t_1 < \cdots < t_K$, the normalized increments

$$\left\{ F(t_1), \frac{F(t_2) - F(t_1)}{1 - F(t_1)}, \ldots, \frac{F(t_K) - F(t_{K-1})}{1 - F(t_{K-1})} \right\} \tag{2.171}$$

are mutually independent. This condition is strictly weaker than that of eq. (2.170), and several NTR generalizations of the Dirichlet process have been suggested [69, 313]. Any NTR stochastic process can be expressed as $F(t) = 1 - \exp\{-Y(t)\}$ for some monotonically increasing, independent increments process $Y(t)$. For the Dirichlet process,

increments of $Y(t)$ are exponentially distributed [84, 150]. In addition, NTR processes are *tailfree*, so that the posterior distribution $p\big(F(t) \mid \bar{\theta}\big)$ is independent of observations at later times $\bar{\theta} > t$. Generalizing the conjugacy of Prop. 2.5.1, the posterior distribution of $F(t)$ given an observation $\bar{\theta} \leq t$ remains neutral to the right [69].

While NTR processes can more flexibly model temporal structure than the Dirichlet process, they are limited to the real line. A recently proposed class of *spatial neutral to the right* processes [152] provides one extension to general parameter spaces. Alternatively, tailfree processes can be generalized to define conditional independencies on arbitrary sequences of nested partitions [69, 84]. Analogously to Thm. 2.5.2, only Dirichlet processes are tailfree with respect to every hierarchical partition. However, a broader class of *Pólya tree* distributions [84, 179, 200] can be defined via particular, possibly inhomogeneous partition trees. While this tree structure can encode detailed prior knowledge [180], its use of a fixed discretization scales poorly to high–dimensional spaces, and can produce spurious discontinuities. *Dirichlet diffusion trees* [223] address these issues by using a branching process to sample hierarchical dependency structures.

■ 2.5.2 Stick–Breaking Processes

The preceding section provides several implicit characterizations of the Dirichlet process, including a desirable conjugacy property. However, these results do not directly provide a mechanism for sampling from Dirichlet processes, or predicting future observations. In this section, we describe an explicit *stick–breaking* construction [254] which shows that Dirichlet measures are *discrete* with probability one. This leads to a simple Pólya urn model for predictive distributions known as the *Chinese restaurant process* [28, 233]. These representations play a central role in computational methods for Dirichlet processes.

Consider Prop. 2.5.1, which provides an expression for the posterior distribution of a Dirichlet distributed random measure $G \sim \mathrm{DP}(\alpha, H)$ given N observations $\bar{\theta}_i \sim G$. From eq. (2.167), the expected measure of any set $T \subset \Theta$ then equals

$$\mathbb{E}\big[G(T) \mid \bar{\theta}_1, \ldots, \bar{\theta}_N, \alpha, H\big] = \frac{1}{\alpha + N} \left(\alpha H(T) + \sum_{i=1}^{N} \delta_{\bar{\theta}_i}(T) \right) \tag{2.172}$$

For any finite concentration parameter α, this implies that

$$\lim_{N \to \infty} \mathbb{E}\big[G(T) \mid \bar{\theta}_1, \ldots, \bar{\theta}_N, \alpha, H\big] = \sum_{k=1}^{\infty} \pi_k \delta_{\theta_k}(T) \tag{2.173}$$

where $\{\theta_k\}_{k=1}^{\infty}$ are the unique values of the observation sequence $\{\bar{\theta}_i\}_{i=1}^{\infty}$, and π_k is the limiting empirical frequency of θ_k. Assuming the posterior distribution concentrates about its mean, eq. (2.173) suggests that Dirichlet measures are discrete with probability one [160]. The following theorem verifies this hypothesis, and provides an explicit construction for the infinite set of mixture weights.

Theorem 2.5.3. *Let $\pi = \{\pi_k\}_{k=1}^{\infty}$ be an infinite sequence of mixture weights derived from the following stick–breaking process, with parameter $\alpha > 0$:*

$$\beta_k \sim \text{Beta}(1, \alpha) \qquad k = 1, 2, \ldots \tag{2.174}$$

$$\pi_k = \beta_k \prod_{\ell=1}^{k-1}(1 - \beta_\ell) = \beta_k\left(1 - \sum_{\ell=1}^{k-1}\pi_\ell\right) \tag{2.175}$$

Given a base measure H on Θ, consider the following discrete random measure:

$$G(\theta) = \sum_{k=1}^{\infty}\pi_k\delta(\theta, \theta_k) \qquad \theta_k \sim H \tag{2.176}$$

This construction guarantees that $G \sim \text{DP}(\alpha, H)$. Conversely, samples from a Dirichlet process are discrete with probability one, and have a representation as in eq. (2.176).

Proof. The consistency of eq. (2.175) follows from an induction argument. Manipulating this expression, it can be shown that

$$1 - \sum_{k=1}^{K}\pi_k = \prod_{k=1}^{K}(1 - \beta_k) \longrightarrow 0$$

with probability one as $K \to \infty$, so that eq. (2.176) defines a valid probability measure. Ferguson established the almost sure discreteness of G using a normalized gamma process representation [83, 168]. Sethuraman later derived the explicit stick–breaking construction for the mixture weights [254]. The beta distribution of eq. (2.174) arises from the form of marginal distributions of finite Dirichlet densities (see eq. (2.44)). □

The *stick–breaking* interpretation of this construction is illustrated in Fig. 2.22. Mixture weights π partition a unit–length "stick" of probability mass among an infinite set of random parameters. The k^{th} mass π_k is a random proportion β_k of the stick remaining after sampling the first $(k-1)$ mixture weights. As is standard in the statistics literature [150, 233, 289], we use $\pi \sim \text{GEM}(\alpha)$ to indicate a set of mixture weights sampled from this process, named after Griffiths, Engen, and McCloskey.

This representation of the Dirichlet process provides another interpretation of the concentration parameter α. Because the stick–breaking proportions $\beta_k \sim \text{Beta}(1, \alpha)$, standard moment formulas (see eq. (2.40)) show that

$$\mathbb{E}[\beta_k] = \frac{1}{1+\alpha} \tag{2.177}$$

For small α, it follows that the first few mixture components are typically assigned the majority of the probability mass. As $\alpha \to \infty$, samples $G \sim \text{DP}(\alpha, H)$ approach the base measure H by assigning small, roughly uniform weights to a densely sampled set of

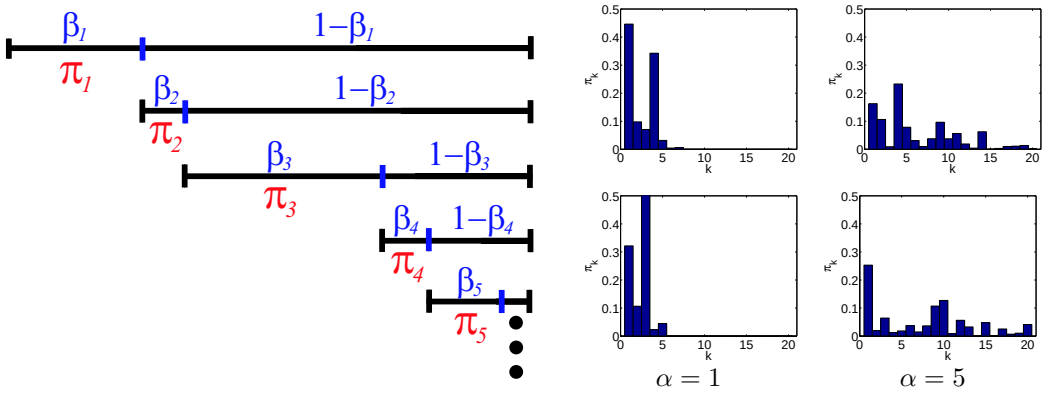


Figure 2.22. Sequential stick–breaking construction of the infinite set of mixture weights $\pi \sim \mathrm{GEM}(\alpha)$ corresponding to a measure $G \sim \mathrm{DP}(\alpha, H)$. *Left:* The first weight $\pi_1 \sim \mathrm{Beta}(1, \alpha)$. Each subsequent weight π_k (red) is some random proportion β_k (blue) of the remaining, unbroken "stick" of probability mass. *Right:* The first $K = 20$ weights generated by four random stick–breaking constructions (two with $\alpha = 1$, two with $\alpha = 5$). Note that the weights π_k do not monotonically decrease.

discrete parameters $\{\theta_k\}_{k=1}^{\infty}$. For a given α and dataset size N, there are strong bounds on the accuracy of particular finite truncations of this stick–breaking process [147], which are often used in approximate computational methods [29, 147, 148, 289].

Several other stick–breaking processes have been proposed which sample the proportions β_k from different distributions [147, 148, 233]. For example, the two–parameter Poisson–Dirichlet, or Pitman–Yor, process [234] can produce heavier–tailed weight distributions which better match power laws arising in natural language processing [117, 287]. As we show next, these stick–breaking processes sometimes lead to predictive distributions with simple Pólya urn representations.

Prediction via Pólya Urns

Because Dirichlet processes produce discrete random measures G, there is a strictly positive probability of multiple observations $\bar{\theta}_i \sim G$ taking identical values. Given N observations $\{\bar{\theta}_i\}_{i=1}^{N}$, suppose that they take $K \leq N$ distinct values $\{\theta_k\}_{k=1}^{K}$. The posterior expectation of any set $T \subset \Theta$ (see eq. (2.172)) can then be written as

$$\mathbb{E}\big[G(T) \mid \bar{\theta}_1, \ldots, \bar{\theta}_N, \alpha, H\big] = \frac{1}{\alpha + N}\left(\alpha H(T) + \sum_{k=1}^{K} N_k \delta_{\theta_k}(T)\right) \tag{2.178}$$

$$N_k \triangleq \sum_{i=1}^{N} \delta(\bar{\theta}_i, \theta_k) \qquad k = 1, \ldots, K \tag{2.179}$$

Note that N_k is defined to be the number of previous observations equaling θ_k, and that K is a random variable [10, 28, 233]. Analyzing this expression, the predictive distribution of the next observation $\bar{\theta}_{N+1} \sim G$ can be explicitly characterized.

Theorem 2.5.4. *Let $G \sim \mathrm{DP}(\alpha, H)$ be distributed according to a Dirichlet process, where the base measure H has corresponding density $h(\theta)$. Consider a set of N observations $\bar{\theta}_i \sim G$ taking K distinct values $\{\theta_k\}_{k=1}^K$. The predictive distribution of the next observation then equals*

$$p\big(\bar{\theta}_{N+1} = \theta \mid \bar{\theta}_1, \dots, \bar{\theta}_N, \alpha, H\big) = \frac{1}{\alpha + N}\left(\alpha h(\theta) + \sum_{k=1}^{K} N_k \delta(\theta, \theta_k)\right) \tag{2.180}$$

where N_k is the number of previous observations of θ_k, as in eq. (2.179).

Proof. Letting T_k be an arbitrarily small set containing θ_k, eq. (2.178) suggests that $\Pr\big[\bar{\theta}_{N+1} = \theta_k\big] \propto N_k$, while the base measure is assigned total posterior probability $\alpha/(\alpha + N)$. For a formal argument, see Blackwell and MacQueen [28]. □

Dirichlet processes thus lead to simple predictive distributions, which can be evaluated by caching the number of previous observations taking each distinct value.

The generative process defined by Thm. 2.5.4 can be interpreted via a generalized *Pólya urn* model [28]. Consider an urn containing one ball for each preceding observation, with a different color for each distinct θ_k. For each ball drawn from the urn, we replace that ball and add one more of the same color. There is also a special "weighted" ball which is drawn with probability proportional to α normal balls, and has some new, previously unseen color $\theta_{\bar{k}} \sim H$. This procedure can be used to sample observations from a Dirichlet process, without explicitly constructing the underlying mixture $G \sim \mathrm{DP}(\alpha, H)$.

Chinese Restaurant Processes

As the Dirichlet process assigns observations $\bar{\theta}_i$ to distinct values θ_k, it implicitly *partitions* the data. Let z_i indicate the subset, or cluster, associated with the i^{th} observation, so that $\bar{\theta}_i = \theta_{z_i}$. The predictive distribution of eq. (2.180) then shows that

$$p(z_{N+1} = z \mid z_1, \dots, z_N, \alpha) = \frac{1}{\alpha + N}\left(\sum_{k=1}^{K} N_k \delta(z, k) + \alpha \delta(z, \bar{k})\right) \tag{2.181}$$

where $\bar{k}$ denotes a new, previously empty cluster. Inspired by the seemingly infinite seating capacity of restaurants in San Francisco's Chinatown, Pitman and Dubins called this distribution over partitions the *Chinese restaurant process* [233]. The restaurant's infinite set of tables are analogous to clusters, and customers to observations (see Fig. 2.23). Customers are social, so that the i^{th} customer sits at table k with probability proportional to the number of already seated diners N_k. Sometimes, however, customers (observations) choose a new table (cluster). Note that there is no *a priori* distinction between the unoccupied tables. Dirichlet processes extend this construction by serving each table a different, independently chosen dish (parameter) θ_k.

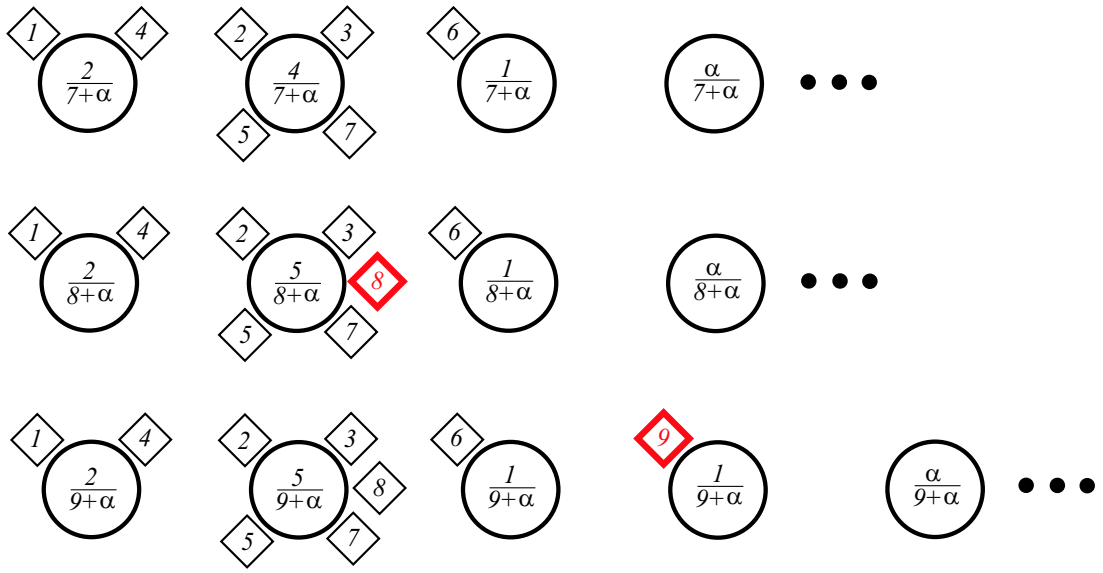


Figure 2.23. Chinese restaurant process interpretation of the partitions induced by the Dirichlet process $\mathrm{DP}(\alpha, H)$. Tables (circles) are analogous to clusters, and customers (diamonds) to a series of observations. *Top row:* A starting configuration, in which seven customers occupy three tables. Each table is labeled with the probability that the next customer sits there. *Middle row:* New customers sit at occupied table k with probability proportional to the number of previously seated diners N_k. In this example, the eighth customer joins the most popular, and hence likely, table. *Bottom row:* Customers may also sit at one of the infinitely many unoccupied tables. The ninth diner does this.

Importantly, the Chinese restaurant process induces an *exchangeable* distribution on partitions, so that the joint distribution is invariant to the order in which observations are assigned to clusters. Exchangeability follows from De Finetti's Theorem [28], given the connection to Dirichlet processes established by Thm. 2.5.4. Alternatively, it can be directly verified via an analysis of eq. (2.181). There are a variety of combinatorial characterizations of the partition structure produced by the Chinese restaurant process [10, 121, 232, 233]. In particular, the number of occupied tables K almost surely approaches $\alpha \log(N)$ as $N \to \infty$. This shows that the Dirichlet process is indeed a nonparametric prior, as it favors models whose complexity grows with the dataset size.

Generalizations of the Chinese restaurant process can be constructed for certain other stick–breaking processes, including the Pitman–Yor process [147, 233]. Importantly, the simple predictive distributions induced by these processes lead to efficient Monte Carlo algorithms for learning and inference [76, 222, 237]. In contrast, other alternatives such as neutral to the right processes may have posterior distributions which lack simple, explicit forms [152].

■ 2.5.3 Dirichlet Process Mixtures

Using nonparametric methods, we now revisit De Finetti's representation (Thm. 2.2.2) of exchangeable random variables $\{x_i\}_{i=1}^N$. To apply this theory when $x_i \in \mathcal{X}$ is continuous, we need a tractable framework for learning infinite–dimensional probability measures. As shown in previous sections, Dirichlet processes lead to posterior distributions with simple, explicit forms. However, because it assigns probability one to discrete measures (Thm. 2.5.3), a Dirichlet process prior expects multiple observations to take *identical* values. Furthermore, Thm. 2.5.2 shows that the posterior measure assigned to x_i would *never* be influenced by observations $x_j \neq x_i$, regardless of their proximity. In many applications, Dirichlet processes are thus too restrictive to directly model continuous observations [216, 232].

To address these issues, we consider a hierarchical model in which observations are sampled from some parameterized family $F(\theta)$. As in finite mixture models (see Fig. 2.9), each observation x_i is based on an independently sampled parameter $\bar{\theta}_i$:

$$\begin{aligned} \bar{\theta}_i &\sim G \\ x_i &\sim F\big(\bar{\theta}_i\big) \end{aligned} \tag{2.182}$$

For greater flexibility and robustness, we place a nonparametric, Dirichlet process prior on the latent parameter distribution $G \sim \mathrm{DP}(\alpha, H)$. The stick–breaking construction of Thm. 2.5.3 then implies that

$$G(\theta) = \sum_{k=1}^{\infty} \pi_k \delta(\theta, \theta_k) \qquad \begin{aligned} \pi &\sim \mathrm{GEM}(\alpha) \\ \theta_k &\sim H(\lambda) \qquad k = 1, 2, \ldots \end{aligned} \tag{2.183}$$

Fig. 2.24 shows a graphical representation of the resulting *Dirichlet process mixture* model [10, 76, 187]. Typically, $F(\theta)$ is some exponential family of densities, and $H(\lambda)$ a corresponding conjugate prior. Note that this construction allows differing observations to be associated with the same underlying cluster. The likelihood $F(\theta)$ effectively imposes a notion of distance on $\mathcal{X}$, and thus allows observations to be extrapolated to neighboring regions. By using a Dirichlet process, however, we avoid constraining these predictions with a global parametric form. Fig. 2.25 illustrates Dirichlet process mixtures in which $\theta_k = (\mu_k, \Lambda_k)$ parameterizes a two–dimensional Gaussian.

The Chinese restaurant process provides another useful representation of Dirichlet process mixtures [76, 237]. Letting z_i denote the unique cluster, or table, associated with x_i, the generative process of eq. (2.182) can be equivalently expressed as

$$\begin{aligned} z_i &\sim \pi \\ x_i &\sim F(\theta_{z_i}) \end{aligned} \tag{2.184}$$

As summarized in Fig. 2.24, marginalizing these indicator variables reveals an *infinite* mixture model with the following form:

$$p(x \mid \pi, \theta_1, \theta_2, \ldots) = \sum_{k=1}^{\infty} \pi_k f(x \mid \theta_k) \tag{2.185}$$

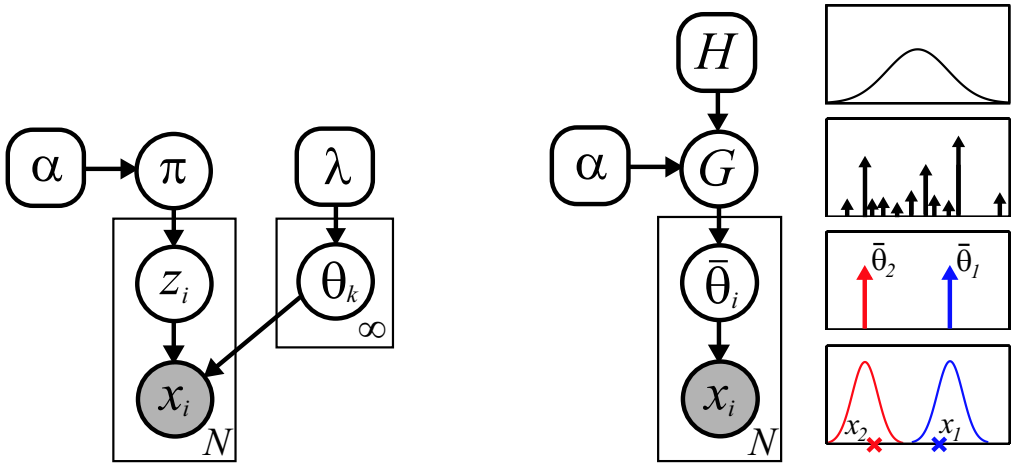


Figure 2.24. Directed graphical representations of an infinite, Dirichlet process mixture model. Mixture weights $\pi \sim \text{GEM}(\alpha)$ follow a stick–breaking process, while cluster parameters are assigned independent priors $\theta_k \sim H(\lambda)$. *Left:* Indicator variable representation, in which $z_i \sim \pi$ is the cluster that generates $x_i \sim F(\theta_{z_i})$. *Right:* Alternative distributional form, in which G is an infinite discrete distribution on Θ. $\bar{\theta}_i \sim G$ are the parameters of the cluster that generates $x_i \sim F(\bar{\theta}_i)$. We illustrate with an infinite Gaussian mixture, where cluster variances are known (bottom) and $H(\lambda)$ is a Gaussian prior on cluster means (top). Sampled cluster means $\bar{\theta}_1, \bar{\theta}_2$, and corresponding Gaussians, are shown for two observations x_1, x_2.

Rather than choose a finite model order K, Dirichlet process mixtures use the stick–breaking prior to control complexity (see Fig. 2.22). As we discuss later, this relaxation leads to algorithms which automatically infer the number of clusters exhibited by a particular dataset. Importantly, the predictive distribution implied by the Chinese restaurant process (eq. (2.181)) has a *clustering* bias, and favors simpler models in which observations (customers) share parameters (dishes). Additional clusters (tables) appear as more observations are generated (see Fig. 2.25).

Learning via Gibbs Sampling

Given N observations $x = \{x_i\}_{i=1}^N$ from a Dirichlet process mixture as in Fig. 2.24, we would like to infer the number of latent clusters underlying those observations, and their parameters θ_k. As with finite mixture models, the exact posterior distribution $p(\pi, \theta \mid x)$ contains terms corresponding to each possible partition z of the observations [10, 187]. While the Chinese restaurant process tractably specifies the prior probability of individual partitions (see eq. (2.181)), explicit enumeration of the exponentially large set of potential partitions is intractable. There is thus an extensive literature on approximate computational methods for Dirichlet process mixtures [29, 76, 121, 147, 148, 151, 222].

In this section, we generalize the Rao–Blackwellized Gibbs sampler of Alg. 2.2 from finite to infinite mixture models. As before, we sample the indicator variables $z = \{z_i\}_{i=1}^N$ assigning observations to latent clusters, marginalizing mixture weights π

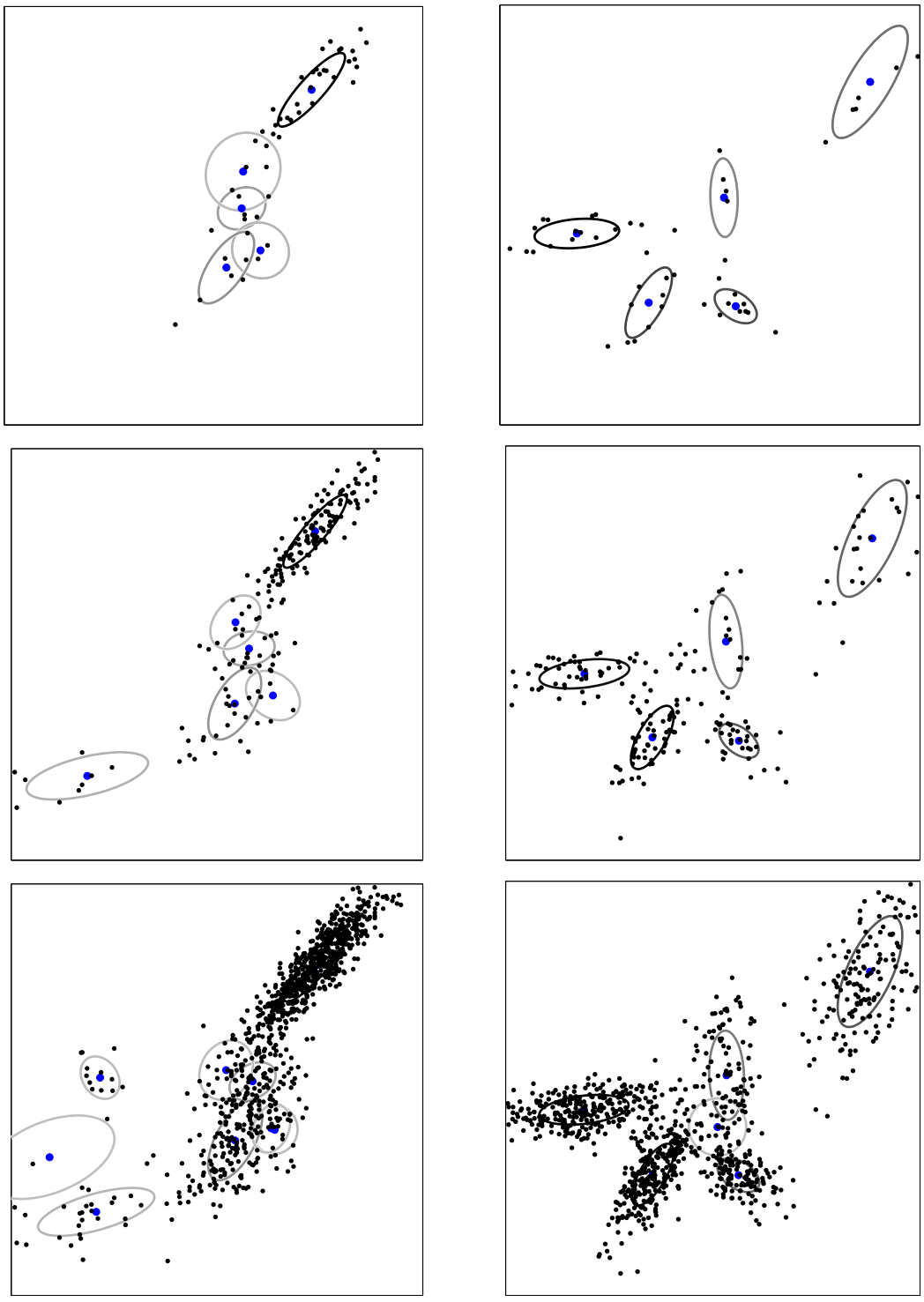


Figure 2.25. Each column shows an observation sequence from a Dirichlet process mixture of 2D Gaussians, with concentration $\alpha = 1$. We show the existing clusters (covariance ellipses, intensity proportional to probability) after $N = 50$ (top), $N = 200$ (middle), and $N = 1000$ (bottom) observations.

and parameters $\{\theta_k\}_{k=1}^{\infty}$. The resulting *collapsed* Gibbs sampler [222] is typically more efficient than alternatives which explicitly sample parameters [76, 237]. For simplicity, we assume that cluster priors $H(\lambda)$ are conjugate to the chosen likelihood $F(\theta)$. Non–conjugate priors can be handled via auxiliary variable methods [222].

Given fixed cluster assignments $z_{\backslash i}$ for other observations, Fig. 2.24 implies that the posterior distribution of z_i factors as follows:

$$p(z_i \mid z_{\backslash i}, x, \alpha, \lambda) \propto p(z_i \mid z_{\backslash i}, \alpha)\, p(x_i \mid z, x_{\backslash i}, \lambda) \tag{2.186}$$

The first term expresses the prior on partitions implied by the Chinese restaurant process. Recall that the Dirichlet process induces an exchangeable distribution on partitions, which is invariant to the order of observations. In evaluating eq. (2.186), we may thus equivalently think of z_i as the *last* in a sequence of N observations. If $z_{\backslash i}$ instantiates K clusters, and assigns N_k^{-i} observations to the k^{th} cluster, eq. (2.181) then implies that

$$p(z_i \mid z_{\backslash i}, \alpha) = \frac{1}{\alpha + N - 1} \left(\sum_{k=1}^{K} N_k^{-i} \delta(z_i, k) + \alpha \delta(z_i, \bar{k}) \right) \tag{2.187}$$

As before, $\bar{k}$ denotes one of the infinitely many unoccupied clusters.

For the K clusters to which $z_{\backslash i}$ assigns observations, the likelihood of eq. (2.186) follows the expression (eq. (2.163)) derived for the finite mixture Gibbs sampler:

$$p(x_i \mid z_i = k, z_{\backslash i}, x_{\backslash i}, \lambda) = p(x_i \mid \{x_j \mid z_j = k, j \neq i\}, \lambda) \tag{2.188}$$

This term is the predictive likelihood of x_i, as determined by Prop. 2.1.4, given the other observations which $z_{\backslash i}$ associates with that cluster. Similarly, new clusters $\bar{k}$ are based upon the predictive likelihood implied by the prior hyperparameters λ:

$$p\big(x_i \mid z_i = \bar{k}, z_{\backslash i}, x_{\backslash i}, \lambda\big) = p(x_i \mid \lambda) = \int_{\Theta} f(x_i \mid \theta)\, h(\theta \mid \lambda)\, d\theta \tag{2.189}$$

Assuming $H(\lambda)$ specifies a proper, conjugate prior, eq. (2.189) has a closed form similar to that of eq. (2.188).

Combining these expressions, we arrive at the Gibbs sampler of Alg. 2.3. As in Alg. 2.2, we cache and recursively update statistics of each cluster's associated observations (see Thm. 2.1.2). Because the infinite set of potential clusters have identical priors, we only explicitly store a randomly sized list of those clusters to which at least one observation is assigned. Standard data structures then allow clusters to be efficiently created when needed (Alg. 2.3, step 2(c)), and deleted if all associated observations are reassigned (Alg. 2.3, step 4). Comparing Algs. 2.2 and 2.3, we see that even though Dirichlet process mixtures have infinitely many parameters, learning is possible via a simple extension of algorithms developed for finite mixture models.

Cluster assignments $z^{(t)}$ produced by the Gibbs sampler of Alg. 2.3 provide estimates $K^{(t)}$ of the *number* of clusters underlying the observations x, as well as their

Given the previous concentration parameter $\alpha^{(t-1)}$, cluster assignments $z^{(t-1)}$, and cached statistics for the K current clusters, sequentially sample new assignments as follows:

1. Sample a random permutation $\tau(\cdot)$ of the integers $\{1, \dots, N\}$.
2. Set $\alpha = \alpha^{(t-1)}$ and $z = z^{(t-1)}$. For each $i \in \{\tau(1), \dots, \tau(N)\}$, resample z_i as follows:
   (a) For each of the K existing clusters, determine the predictive likelihood
   $$f_k(x_i) = p(x_i \mid \{x_j \mid z_j = k, j \neq i\}, \lambda)$$
   This likelihood can be computed from cached sufficient statistics via Prop. 2.1.4. Also determine the likelihood $f_{\bar{k}}(x_i)$ of a potential new cluster $\bar{k}$ via eq. (2.189).
   (b) Sample a new cluster assignment z_i from the following $(K+1)$–dim. multinomial:
   $$z_i \sim \frac{1}{Z_i}\left(\alpha f_{\bar{k}}(x_i)\delta(z_i, \bar{k}) + \sum_{k=1}^{K} N_k^{-i} f_k(x_i)\delta(z_i, k)\right) \qquad Z_i = \alpha f_{\bar{k}}(x_i) + \sum_{k=1}^{K} N_k^{-i} f_k(x_i)$$
   N_k^{-i} is the number of other observations currently assigned to cluster k.
   (c) Update cached sufficient statistics to reflect the assignment of x_i to cluster z_i. If $z_i = \bar{k}$, create a new cluster and increment K.
3. Set $z^{(t)} = z$. Optionally, mixture parameters for the K currently instantiated clusters may be sampled as in step 3 of Alg. 2.1.
4. If any current clusters are empty ($N_k = 0$), remove them and decrement K accordingly.
5. If $\alpha \sim \text{Gamma}(a, b)$, sample $\alpha^{(t)} \sim p(\alpha \mid K, N, a, b)$ via auxiliary variable methods [76].

Algorithm 2.3. Rao–Blackwellized Gibbs sampler for an infinite, Dirichlet process mixture model, as defined in Fig. 2.24. Each iteration sequentially resamples the cluster assignments for all N observations $x = \{x_i\}_{i=1}^N$ in a different random order. Mixture parameters are integrated out of the sampling recursion using cached sufficient statistics. These statistics are stored in a dynamically resized list of those clusters to which observations are currently assigned.

associated parameters. Dirichlet processes thus effectively allow integrated exploration of models with different complexity. Predictions based on these samples average over mixtures of varying size, avoiding the difficulties inherent in selecting a single model. The computational cost of each sampling update is proportional to the number of currently instantiated clusters $K^{(t)}$, and thus varies randomly from iteration to iteration. Asymptotically, $K \to \alpha \log(N)$ as $N \to \infty$ (see [10, 233]), so each iteration of Alg. 2.3 requires approximately $\mathcal{O}(\alpha N \log(N))$ operations to resample all assignments. For practical datasets, however, the number of instantiated clusters depends substantially on the structure and alignment of the given observations.

While predictions derived from Dirichlet process mixtures are typically robust to the concentration parameter α, the number K of clusters with significant posterior probability shows greater sensitivity [76]. In many applications, it is therefore useful to choose a weakly informative prior for α, and sample from its posterior while learning cluster parameters. If $\alpha \sim \text{Gamma}(a, b)$ is assigned a gamma prior [107], its posterior is a simple function of K, and samples are easily drawn via an auxiliary variable method [76]. Incorporating this technique in our Gibbs sampler (Alg. 2.3, step 5), we empirically

find that it converges more reliably, and matches the performance of procedures which tune α via computationally demanding cross–validation.

In Fig. 2.26, we use the Gibbs sampler of Alg. 2.3 to fit a Dirichlet process mixture of Gaussians to $N = 300$ two–dimensional observations. Placing a vague gamma prior $\alpha \sim \text{Gamma}(0.2, 0.1)$ on the concentration parameter, initial iterations frequently create and delete mixture components. However, the sampler quickly stabilizes (see Fig. 2.27), and discovers that with high probability the data was generated by $K = 4$ Gaussians. Fig. 2.27 also compares this Dirichlet process model to a 4–component mixture estimated via the Rao–Blackwellized sampler of Alg. 2.2. Despite having to search over mixtures of varying order, the Dirichlet process sampler typically converges faster. In particular, by creating redundant clusters in early iterations, it avoids local optima which trap the 4–component Gibbs sampler. This behavior is reminiscent of methods which iteratively prune clusters from finite mixtures [87], but arises directly from the Dirichlet process prior rather than complexity–based model selection criteria.

An Infinite Limit of Finite Mixtures

The graphical representation of the Dirichlet process mixture model (see Fig. 2.24) exhibits striking similarities to the finite, K–component mixture model of Fig. 2.9. In this section, we show that the Dirichlet process is indeed the limit as $K \to \infty$ of a particular sequence of finite Dirichlet distributions. This result provides intuition about the assumptions and biases inherent in Dirichlet processes, and leads to alternative computational methods for Dirichlet process mixtures.

As in Sec. 2.2.4, we begin by placing a symmetric Dirichlet prior, with precision α, on the weights π assigned to the K components of a finite mixture model:

$$(\pi_1, \ldots, \pi_K) \sim \text{Dir}\left(\frac{\alpha}{K}, \ldots, \frac{\alpha}{K}\right) \tag{2.190}$$

Consider the Rao–Blackwellized Gibbs sampler for this finite mixture, as summarized in Alg. 2.2. Given cluster assignments $z_{\setminus i}$ for all observations except x_i, the Dirichlet prior implies the following predictive distribution (see eq. (2.162)):

$$p(z_i = k \mid z_{\setminus i}, \alpha) = \frac{N_k^{-i} + \alpha/K}{\alpha + N - 1} \qquad N_k^{-i} = \sum_{j \neq i} \delta(z_j, k) \tag{2.191}$$

In the limit as $K \to \infty$ with fixed precision α, the predictive probability of clusters k to which observations are assigned ($N_k^{-i} > 0$) approaches

$$\lim_{K \to \infty} p(z_i = k \mid z_{\setminus i}, \alpha) = \frac{N_k^{-i}}{\alpha + N - 1} \tag{2.192}$$

Similarly, the probability of any particular unoccupied cluster approaches zero as K becomes large. However, the total probability assigned to *all* unoccupied clusters is

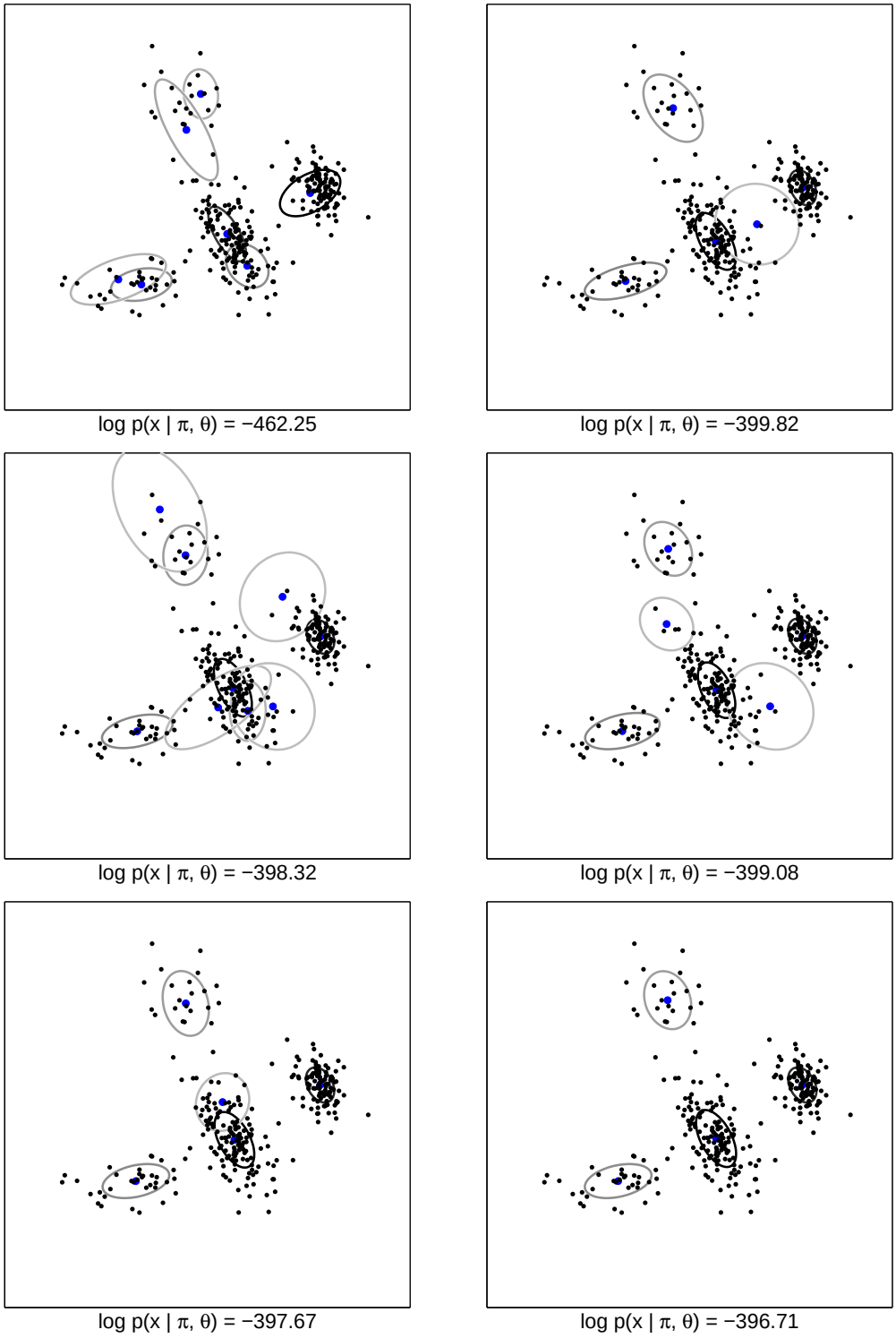


Figure 2.26. Learning a mixture of Gaussians using the Dirichlet process Gibbs sampler of Alg. 2.3. Columns show the parameters of clusters currently assigned to observations, and corresponding data log–likelihoods, after T=2 (top), T=10 (middle), and T=50 (bottom) iterations from two initializations.

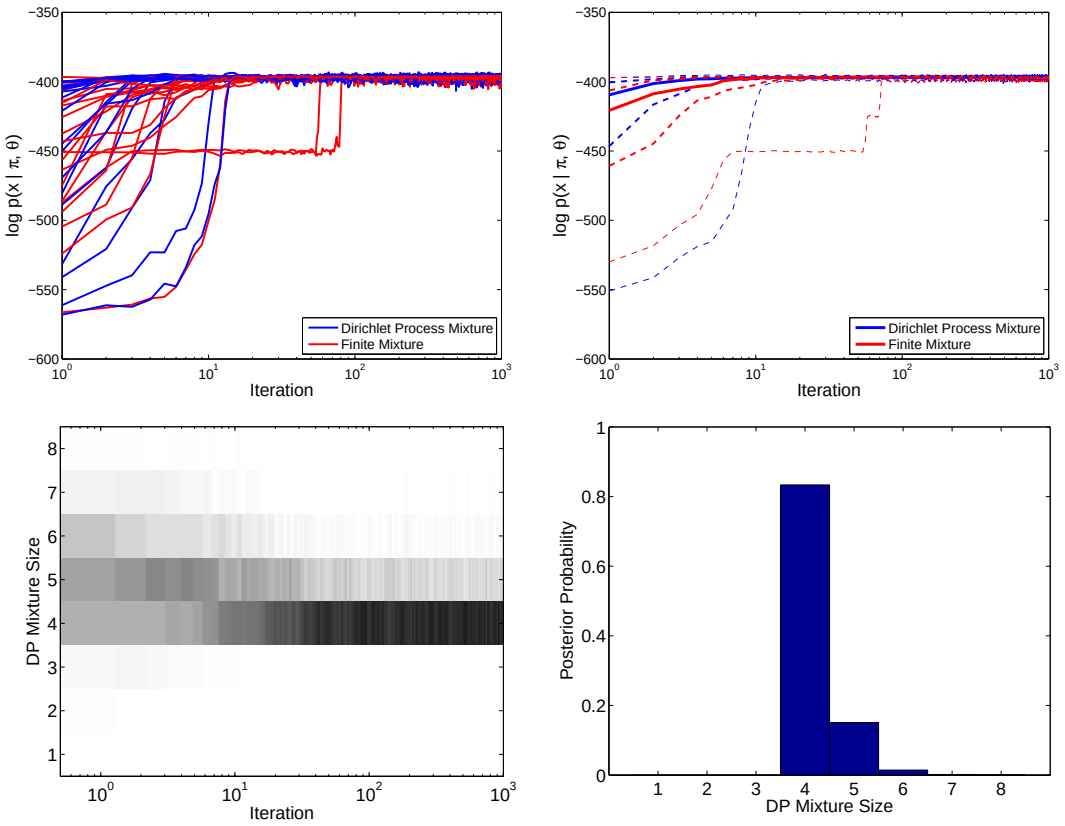


Figure 2.27. Comparison of Rao–Blackwellized Gibbs samplers for a Dirichlet process mixture (Alg. 2.3, dark blue) and a finite, 4–component mixture (Alg. 2.2, light red). We compare data log–likelihoods at each of 1000 iterations for the single $N = 300$ point dataset of Fig. 2.26. *Top left:* Log–likelihood sequences for 20 different random initializations of each algorithm. *Top Right:* From 100 different random initializations, we show the median (solid), 0.25 and 0.75 quantiles (thick dashed), and 0.05 and 0.95 quantiles (thin dashed) of the resulting log–likelihood sequences. *Bottom:* Number of mixture components with at least 2% of the probability mass at each iteration (left, intensity proportional to posterior probability), and averaging across the final 900 iterations (right).

positive, and determined by the complement of existing cluster weights as follows:

$$p(z_i \neq z_j \text{ for all } j \neq i \mid z_{\setminus i}, \alpha) = 1 - \sum_{k | N_k^{-i} > 0} p(z_i = k \mid z_{\setminus i}, \alpha) \tag{2.193}$$

$$\lim_{K \to \infty} p(z_i \neq z_j \text{ for all } j \neq i \mid z_{\setminus i}, \alpha) = 1 - \sum_{k} \frac{N_k^{-i}}{\alpha + N - 1} = \frac{\alpha}{\alpha + N - 1} \tag{2.194}$$

Note that if z_i is not assigned to an occupied cluster, it must be associated with a new cluster $\bar{k}$. Comparing to eq. (2.187), we then see that the limits of eqs. (2.192, 2.194) are equivalent to the predictive distributions implied by the Chinese restaurant process. The Dirichlet process Gibbs sampler of Alg. 2.3 can thus be directly derived as an

infinite limit of Alg. 2.2, without explicitly invoking the theory underlying Dirichlet processes [221, 237].

The relationships suggested by the preceding arguments can be made more precise. In particular, a combinatorial analysis [121, 150] shows that the finite Dirichlet prior of eq. (2.190) induces a joint distribution on partitions z which approaches the Chinese restaurant process as $K \to \infty$. In this limit, predictions based on the finite mixture model also approach those of the corresponding Dirichlet process.

Theorem 2.5.5. *Let H denote a probability measure on Θ, and $f : \Theta \to \mathbb{R}$ a measurable function which is integrable with respect to H. Consider the K–component discrete distribution G^K, as in eq. (2.83), corresponding to a mixture model with weights following the finite Dirichlet prior $\mathrm{Dir}(\alpha)$ of eq. (2.190). As $K \to \infty$, expectations with respect to G^K then converge in distribution to a corresponding Dirichlet process:*

$$\int_\Theta f(\theta)\, dG^K(\theta) \xrightarrow{\mathcal{D}} \int_\Theta f(\theta)\, dG(\theta) \qquad G \sim \mathrm{DP}(\alpha, H) \tag{2.195}$$

Proof. This result was derived via a stick–breaking representation of the Dirichlet process by Ishwaran and Zarepour (see Thm. 2 of [150]). □

Given the correspondence implied by Thm. 2.5.5, the mixture weights $(\pi_1, \ldots, \pi_K)$ of eq. (2.190) should, in some sense, converge to $\pi \sim \mathrm{GEM}(\alpha)$ as $K \to \infty$. As discussed in Sec. 2.1.3, finite Dirichlet distributions with small precisions are biased towards sparse multinomial distributions (see Fig. 2.1). It can be shown that the stick–breaking construction of Thm. 2.5.3 induces a random, *size–biased permutation* [233] in which the largest weights are typically assigned to earlier clusters (for examples, see Fig. 2.22). By rank ordering $\pi \sim \mathrm{GEM}(\alpha)$, we recover the *Poisson–Dirichlet* distribution [233, 234], which is also the limiting distribution of reordered, finite Dirichlet samples [168].

Given the limiting behavior of finite mixture models with Dirichlet priors as in eq. (2.190), they provide a natural mechanism for approximating Dirichlet processes. Indeed, a Gibbs sampler similar to those of Algs. 2.1 and 2.2 has been suggested for approximate learning of Dirichlet process mixtures [148]. In general, however, this finite mixture approximation converges slowly with K, and a large number of potential clusters may be required [148, 150]. More accurate approximations, whose error decreases exponentially with K, are obtained by truncating the stick–breaking representation of Thm. 2.5.3. This approach has been used to develop alternative Gibbs samplers [147, 148], as well as a deterministic, variational approximation [29] which adapts the mean field method described in Sec. 2.3.1.

Model Selection and Consistency

Dirichlet process mixture models provide a popular Bayesian alternative to the kernel density estimators described in Sec. 2.4.2. In such applications, clusters are usually associated with Gaussian kernels [76, 187]. The base measure $H(\lambda)$ may then be used to

encode domain–specific knowledge about the observations' expected location, scale, and variability. For target distributions with sufficiently small tail probabilities, Dirichlet process mixtures of Gaussians provide strongly consistent density estimates [112, 113]. In addition, by allowing posterior covariances to vary across clusters, Dirichlet processes often provide more robust predictions than classic, asymptotically motivated bandwidth selection schemes [263]. Importantly, the Gibbs sampler of Alg. 2.3 also characterizes the posterior uncertainty in the estimated density.

Many other applications of Dirichlet process mixtures involve data generated from some finite, but unknown, number of latent factors [76, 121, 149, 289]. In such cases, the parameters corresponding to different clusters are typically of interest. Several different complexity criteria [87, 203, 314], including Bayesian formulations which optimize predictive likelihoods [46], have been proposed in this context. For applications involving high–dimensional data, however, there may be inherent ambiguities which prevent reliable selection of a single "best" model. Dirichlet process mixtures avoid this issue via an infinite model encompassing finite mixtures of varying order. Mild conditions then guarantee that the Dirichlet process posterior, as characterized by Prop. 2.5.1, asymptotically concentrates on the true set of finite mixture parameters [149].

Other models for finite mixtures place an explicit prior on the number of clusters K, and then separately parameterize mixtures of each order [121, 208, 243]. When mixture weights follow finite Dirichlet distributions, this approach produces the *Dirichlet/multinomial allocation (DMA)* model [121]. In some applications, complex priors $p(K)$ can then be used to encode detailed prior knowledge. However, when less is known about the underlying generative process, these priors involve nuisance parameters which are difficult to specify uninformatively [243, 272]. Indeed, in some applications where the Dirichlet process has favorable asymptotic properties, apparently uninformative finite Dirichlet priors lead to inconsistent parameter estimates [149].

Computational considerations also practically motivate Dirichlet process priors. DMA models are typically learned via Monte Carlo methods which use Metropolis–Hastings moves to step between models of varying order [243, 272]. Such algorithms, including variants of *reversible jump MCMC* [9, 243], require proposal distributions which split, merge, and otherwise transform cluster parameters. Effective proposals must usually be tuned to particular applications, and can be difficult to formulate for hierarchical models of complex, high–dimensional data. While split–merge MCMC methods are readily generalized to Dirichlet process mixtures [55, 121, 151], the simple but effective collapsed Gibbs sampler (Alg. 2.3) has no direct analog for DMA models. For realistic datasets, differences between Dirichlet process and DMA models are often small, with Dirichlet processes exhibiting a slight posterior bias towards mixtures with a few additional, low–weight components [121].

Finally, we note that while Bayesian estimators derived from finite–dimensional models are usually consistent, the asymptotic behavior of nonparametric methods is more subtle [68, 113, 160, 317]. For example, Diaconis and Freedman [68] considered a semiparametric model in which a latent location parameter $\theta \sim \mathcal{N}(0, \Lambda)$, and the un-

known *measurement* distribution underlying independent observations has a Dirichlet process prior $\mathrm{DP}(\alpha, H)$. They demonstrated that a heavy–tailed, Student–t base measure H may then lead to inconsistent estimates of θ. As predicted by more recent theoretical results [113], consistency is regained for log–concave base measures. This and other examples [68, 317] demonstrate the need for careful empirical and, where possible, theoretical validation of nonparametric methods.

■ 2.5.4 Dependent Dirichlet Processes

Many applications involve complex, structured datasets, and cannot be directly posed as standard density estimation problems. In this section, we describe a framework for *dependent Dirichlet processes (DDPs)* [191] which extends nonparametric Bayesian methods to a rich family of hierarchical models.

Consider a continuous or discrete *covariate* space Ω capturing the temporal, spatial, or categorical structure associated with a given dataset. As in many hierarchical models, we associate each $\omega \in \Omega$ with a latent parameter $\theta(\omega)$, whose marginal distribution equals G_ω. Let $\theta = \{\theta(\omega) \mid \omega \in \Omega\}$, $\theta \in \Theta$, denote a global configuration of the parameters. We would like to design a flexible, nonparametric prior for the joint distribution $G(\theta)$. Generalizing the stick–breaking representation of Thm. 2.5.3, a DDP prior takes the following form:

$$G(\theta(\omega)) = \sum_{k=1}^{\infty} \pi_k(\omega)\delta(\theta(\omega), \theta_k(\omega)) \qquad \theta_k \sim H \tag{2.196}$$

In this construction, the base measure H is a stochastic process on Θ. For example, if parameters $\theta_k(\omega)$ are assigned Gaussian marginals H_ω, a Gaussian process provides a natural joint measure [105]. The infinite set of mixture weights then follow a generalized stick–breaking process:

$$\pi_k(\omega) = \beta_k(\omega) \prod_{\ell=1}^{k-1} (1 - \beta_\ell(\omega)) \qquad \beta_k \sim B \tag{2.197}$$

If the stochastic process B is chosen so that its marginals $\beta_k(\omega) \sim \mathrm{Beta}(1, \alpha)$, Thm. 2.5.3 shows that $G_\omega \sim \mathrm{DP}(\alpha, H_\omega)$. However, for appropriately chosen H and B, there will be interesting dependencies in the joint distribution G, implicitly coupling the measures for parameters $\theta(\omega)$ associated with different covariates. See MacEachern [191] for a discussion of conditions ensuring the existence of DDP models.

In the simplest case, the stick–breaking weights of eq. (2.197) are set to the same, constant value $\beta_k(\omega) = \bar{\beta}_k \sim \mathrm{Beta}(1, \alpha)$ for all covariates $\omega \in \Omega$. The resulting DDP models capture dependency by sampling joint parameters θ_k from an appropriately chosen stochastic process [60, 105, 191]. More generally, B may be designed to encourage mixture weights which vary to capture local features of the covariate space [122, 342]. In the following section, we describe a model which uses hierarchically dependent Dirichlet processes to choose weights distinguishing several groups of observations.

Hierarchical Dirichlet Processes

As in Sec. 2.2.4, consider a dataset with J related groups $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_J)$, where $\mathbf{x}_j = (x_{j1}, \ldots, x_{jN_j})$ contains N_j observations. Just as the LDA model [31] shares a finite set of clusters among such groups (see Fig. 2.11), the *hierarchical Dirichlet process (HDP)* [288, 289] provides a nonparametric approach to sharing infinite mixtures.

To construct an HDP, a *global* probability measure $G_0 \sim \text{DP}(\gamma, H)$ is first used to define a set of shared clusters:

$$G_0(\theta) = \sum_{k=1}^{\infty} \beta_k \delta(\theta, \theta_k) \qquad \begin{aligned} \boldsymbol{\beta} &\sim \text{GEM}(\gamma) \\ \theta_k &\sim H(\lambda) \qquad k = 1, 2, \ldots \end{aligned} \tag{2.198}$$

Group–specific mixture distributions $G_j \sim \text{DP}(\alpha, G_0)$ are then independently sampled from a Dirichlet process with discrete base measure G_0, so that

$$G_j(\theta) = \sum_{t=1}^{\infty} \widetilde{\pi}_{jt} \delta(\theta, \widetilde{\theta}_{jt}) \qquad \begin{aligned} \widetilde{\boldsymbol{\pi}}_j &\sim \text{GEM}(\alpha) \\ \widetilde{\theta}_{jt} &\sim G_0 \qquad t = 1, 2, \ldots \end{aligned} \tag{2.199}$$

Each *local* cluster in group j has parameters $\widetilde{\theta}_{jt}$ copied from some global cluster $\theta_{k_{jt}}$, which we indicate by $k_{jt} \sim \boldsymbol{\beta}$. As summarized in the graph of Fig. 2.28, data points in group j are then independently sampled according to this parameter distribution:

$$\begin{aligned} \bar{\theta}_{ji} &\sim G_j \\ x_{ji} &\sim F(\bar{\theta}_{ji}) \end{aligned} \tag{2.200}$$

For computational convenience, we typically define $F(\theta)$ to be an appropriate exponential family, and $H(\lambda)$ a corresponding conjugate prior. As with standard mixtures, eq. (2.200) can be equivalently expressed via a discrete variable t_{ji} indicating the cluster associated with the i^{th} observation:

$$\begin{aligned} t_{ji} &\sim \widetilde{\boldsymbol{\pi}}_j \\ x_{ji} &\sim F(\widetilde{\theta}_{jt_{ji}}) \end{aligned} \tag{2.201}$$

Fig. 2.29 shows an alternative graphical representation of the HDP, based on these explicit assignments of observations to local clusters, and local clusters to global clusters.

Because G_0 is discrete, each group j may create several different copies $\widetilde{\theta}_{jt}$ of the same global cluster θ_k. Aggregating the probabilities assigned to these copies, we can directly express G_j in terms of the distinct global cluster parameters:

$$G_j(\theta) = \sum_{k=1}^{\infty} \pi_{jk} \delta(\theta, \theta_k) \qquad \pi_{jk} = \sum_{t | k_{jt} = k} \widetilde{\pi}_{jt} \tag{2.202}$$

Groups then reuse a common set of global clusters in different proportions. Using Thm. 2.5.1, it can be shown that $\boldsymbol{\pi}_j \sim \text{DP}(\alpha, \boldsymbol{\beta})$, where $\boldsymbol{\beta}$ and $\boldsymbol{\pi}_j$ are interpreted as

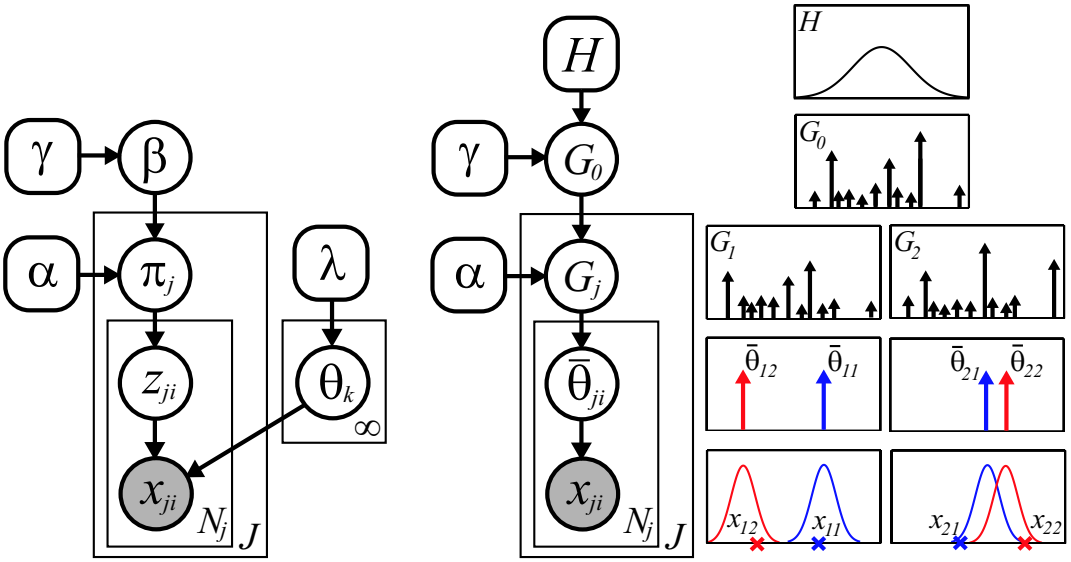


Figure 2.28. Directed graphical representations of a hierarchical Dirichlet process (HDP) mixture model. Global cluster weights $\boldsymbol{\beta} \sim \text{GEM}(\gamma)$ follow a stick–breaking process, while cluster parameters are assigned independent priors $\theta_k \sim H(\lambda)$. *Left:* Explicit stick–breaking representation, in which each group reuses the global clusters with weights $\boldsymbol{\pi}_j \sim \text{DP}(\alpha, \boldsymbol{\beta})$. $z_{ji} \sim \boldsymbol{\pi}_j$ indicates the cluster that generates $x_{ji} \sim F(\theta_{z_{ji}})$. *Right:* Alternative distributional form, in which $G_0 \sim \text{DP}(\gamma, H)$ is an infinite discrete distribution on Θ, and $G_j \sim \text{DP}(\alpha, G_0)$ a reweighted, group–specific distribution. $\bar{\theta}_{ji} \sim G_j$ are then the parameters of the cluster that generates $x_{ji} \sim F(\bar{\theta}_{ji})$. We illustrate with a shared, infinite Gaussian mixture, where cluster variances are known (bottom) and $H(\lambda)$ is a Gaussian prior on cluster means (top). Sampled cluster means $\bar{\theta}_{j1}, \bar{\theta}_{j2}$, and corresponding Gaussians, are shown for two observations x_{j1}, x_{j2} in each of two groups G_1, G_2.

measures on the positive integers [289]. Thus, $\boldsymbol{\beta}$ determines the average weight of local clusters ($\mathbb{E}[\pi_{jk}] = \beta_k$), while α controls the variability of cluster weights across groups. Note that eq. (2.202) suggests the alternative graphical model of Fig. 2.28, in which $z_{ji} \sim \boldsymbol{\pi}_j$ directly indicates the *global* cluster associated with x_{ji}. In contrast, Fig. 2.29 indirectly determines global cluster assignments via local clusters, taking $z_{ji} = k_{jt_{ji}}$.

Comparing these representations to Fig. 2.11, we see that HDPs share clusters as in the LDA model, but remove the need for model order selection. In terms of the DDP framework, the global measure G_0 provides a particular, convenient mechanism for inducing dependencies among the mixture weights in different groups. Note that the discreteness of G_0 plays a critical role in this construction. If, for example, we had instead taken $G_j \sim \text{DP}(\alpha, H)$ with H continuous, the stick–breaking construction of Thm. 2.5.3 shows that groups would learn independent sets of *disjoint* clusters.

Extending the analogy of Fig. 2.23, we may alternatively formulate the HDP representation of Fig. 2.29 in terms of a *Chinese restaurant franchise* [289]. In this interpretation, each group defines a separate restaurant in which customers (observations) x_{ji} sit at tables (clusters) t_{ji}. Each table shares a single dish (parameter) $\widetilde{\theta}_t$, which is

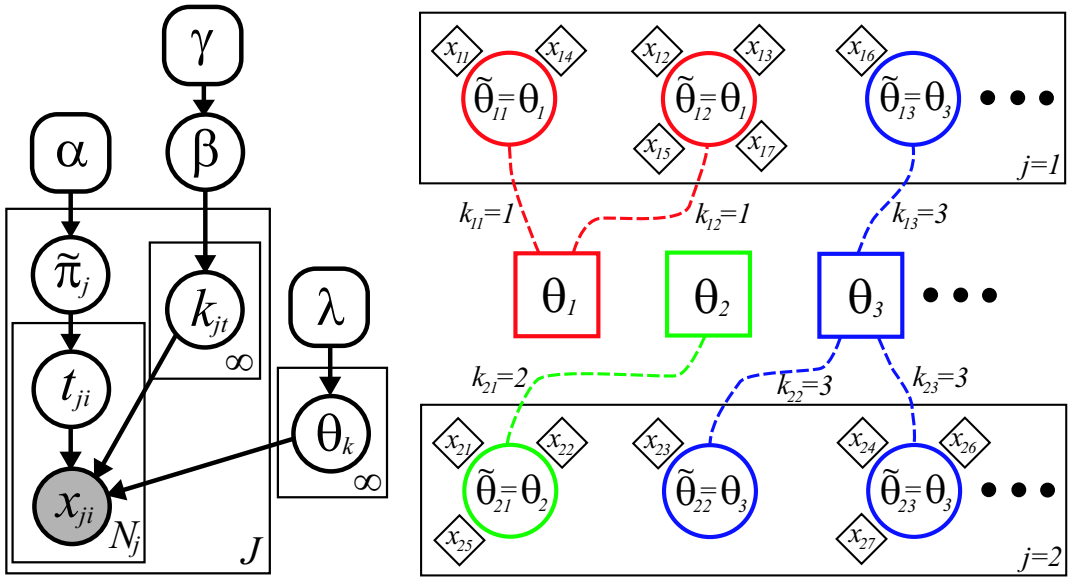


Figure 2.29. Chinese restaurant franchise representation of the HDP model of Fig. 2.28. *Left:* Global cluster parameters are assigned independent priors $\theta_k \sim H(\lambda)$, and reused by groups with frequencies $\boldsymbol{\beta} \sim \text{GEM}(\gamma)$. Each group j has infinitely many local clusters (tables) t, which are associated with a single global cluster $k_{jt} \sim \boldsymbol{\beta}$. Observations (customers) x_{ji} are independently assigned to some table $t_{ji} \sim \tilde{\boldsymbol{\pi}}_j$, and thus indirectly associated with the global cluster (dish) $\theta_{z_{ji}}$, where $z_{ji} = k_{jt_{ji}}$. *Right:* Example in which a franchise menu with dishes θ_k (squares, center) is shared among tables (circles, top and bottom) in two different restaurants (groups). All customers (diamonds) seated at a given table share the same dish (global cluster parameter).

ordered from a menu G_0 shared among restaurants (groups). As before, let k_{jt} indicate the global parameter $\theta_{k_{jt}}$ assigned to table t in group j, and $\mathbf{k}_j$ the parameters for all of that group's tables. We may then integrate over G_0 and G_j (as in eq. (2.181)) to find the conditional distributions of these indicator variables:

$$p(t_{ji} \mid t_{j1}, \ldots, t_{ji-1}, \alpha) \propto \sum_t N_{jt}\delta(t_{ji}, t) + \alpha\delta(t_{ji}, \bar{t}) \tag{2.203}$$

$$p(k_{jt} \mid \mathbf{k}_1, \ldots, \mathbf{k}_{j-1}, k_{j1}, \ldots, k_{jt-1}, \gamma) \propto \sum_k M_k\delta(k_{jt}, k) + \gamma\delta(k_{jt}, \bar{k}) \tag{2.204}$$

Here, M_k is the number of tables previously assigned to θ_k, and N_{jt} the number of customers already seated at the t^{th} table in group j. As before, customers prefer tables t at which many customers are already seated (eq. (2.203)), but sometimes choose a new table $\bar{t}$. Each new table is assigned a dish $k_{j\bar{t}}$ according to eq. (2.204). Popular dishes are more likely to be ordered, but a new dish $\theta_{\bar{k}} \sim H$ may also be selected.

The stick–breaking (Fig. 2.28) and Chinese restaurant franchise (Fig. 2.29) representations provide complementary perspectives on the HDP. In particular, they have each been used to design Monte Carlo methods which infer shared clusters from training

data [289]. In Chap. 5, we describe and extend a Gibbs sampler based on the Chinese restaurant franchise, generalizing the Dirichlet process sampler of Alg. 2.3.

Temporal and Spatial Processes

Models derived from, or related to, the DDP framework have been applied to several application domains. For example, an *analysis of densities* [296, 297] approach has been used to determine multiple related density estimates. This model is similar to the HDP of Fig. 2.28, except that the base measure G_0 is convolved with a Gaussian kernel to construct a continuous, global density estimate. Alternatively, for applications involving observed covariates, the DDP construction of eq. (2.196) has been used to design nonparametric models in which each cluster parameterizes a standard, Gaussian ANOVA model [60]. This method more robustly describes datasets which mix several different global correlation structures.

Related methods have been used to model temporal processes. In particular, *time–sensitive* Dirichlet process mixtures [342] consider applications where each observation has an associated time stamp. A generalization of the Chinese restaurant process then encourages observations at similar times to be associated with the same cluster. Dirichlet processes have also been used to develop an *infinite hidden Markov model* [16], avoiding explicit selection of a finite set of discrete states. The infinite HMM can be seen as a special case of the HDP, in which the global measure G_0 is used to couple the transition distributions associated with different latent states [289].

Gaussian processes provide a standard, widely used framework for modeling spatial data [285, 330]. Generalizing this approach, dependent Dirichlet processes have been used to construct infinite mixtures of Gaussian processes [105]. The marginal distribution at each spatial location is then a Dirichlet process mixture of Gaussians, but the Gaussian parameters associated with nearby sites are correlated. Like DDP models based on ANOVA clusters [60], these mixtures of Gaussian processes implicitly assume absolute spatial locations are statistically meaningful, and require replicated observations at *identical* sites. In Chap. 6, we develop a *transformed Dirichlet process* [282] adapted to datasets with different forms of spatial structure.

Chapter 3

Nonparametric Belief Propagation

IN FIELDS such as computer vision, graphical models are often used to describe complex, multimodal relationships among high–dimensional, continuous variables. For example, the articulated models used in visual tracking applications typically have dozens of degrees of freedom. Realistic graphical models for such problems must represent outliers, bimodalities, and other non–Gaussian statistical features. The optimal inference procedures associated with these models typically involve integral equations which lack closed, analytic forms. It is thus necessary to develop families of approximate representations, and corresponding computational methods.

We begin in Sec. 3.1 by reviewing particle filters, which use sample–based density estimates to track complex temporal processes. Sec. 3.2 then generalizes these sequential Monte Carlo methods to inference problems defined on arbitrary graphs. The resulting *nonparametric belief propagation* (NBP) algorithm uses importance sampling techniques to recursively approximate continuous, non–Gaussian distributions. As described in Sec. 3.3, NBP makes few assumptions about the potentials defining the underlying model, and can thus be flexibly applied in many different application domains.

At each iteration of the NBP algorithm, we sample from a product of Gaussian mixtures, and thereby fuse information from different parts of the graph. Sec. 3.4 develops several algorithms for this sampling step, including a pair of efficient samplers derived from multiscale, KD–tree density representations. In Sec. 3.5, we then validate NBP using Gaussian graphical models, and describe a simple part–based model which allows reconstruction of occluded facial features.

This chapter describes results developed in collaboration with Dr. Alexander Ihler. Portions of this work were presented at the 2003 IEEE Conference on Computer Vision and Pattern Recognition [277], and the 2003 Conference on Neural Information Processing Systems [144].

■ 3.1 Particle Filters

Consider a temporal stochastic process described by a first–order hidden Markov model (HMM), as introduced in Sec. 2.2.3. Let $x = \{x_t\}_{t=0}^{T-1}$ denote the hidden states at each of T time points, and $y = \{y_t\}_{t=0}^{T-1}$ a corresponding observation sequence. As expressed

by the graph of Fig. 2.7, the joint distribution factors as

$$p(x, y) = p(x_0)\, p(y_0 \mid x_0) \prod_{t=1}^{T-1} p(x_t \mid x_{t-1})\, p(y_t \mid x_t) \tag{3.1}$$

In this section, we suppose that the hidden states take values in some continuous space $x_t \in \mathcal{X}_t$, and develop computational methods which tractably approximate their posterior distribution.

In Markov chains and other tree–structured graphs, the belief propagation (BP) algorithm [178, 231, 255, 339] can be used to efficiently infer the posterior distributions $p(x_t \mid y)$ of the state variables. As described in Sec. 2.3.2, BP is based on a series of messages passed between neighboring nodes. For an HMM factorized as in eq. (3.1), the "forward" BP message passed to subsequent time steps is computed via the following recursion:

$$m_{t,t+1}(x_{t+1}) \propto \int_{\mathcal{X}_t} p(x_{t+1} \mid x_t)\, p(y_t \mid x_t)\, m_{t-1,t}(x_t)\; dx_t \tag{3.2}$$

For such HMMs, these BP messages have an interesting probabilistic interpretation. In particular, the outgoing message from the starting timepoint equals

$$m_{0,1}(x_1) \propto \int_{\mathcal{X}_0} p(x_1 \mid x_0)\, p(x_0)\, p(y_0 \mid x_0)\; dx_0 \propto p(x_1 \mid y_0) \tag{3.3}$$

Letting $y_{\overline{t}} = \{y_0, y_1, \ldots, y_t\}$ denote those observations seen up to time t, a simple induction argument then shows that

$$m_{t-1,t}(x_t) \propto p\big(x_t \mid y_{\overline{t-1}}\big) \tag{3.4}$$

$$m_{t-1,t}(x_t)\, p(y_t \mid x_t) \propto p(x_t \mid y_{\overline{t}}) \tag{3.5}$$

Forward messages thus equal the predictive distribution of the next hidden state, given all preceding observations. Rescaling these messages by the current observation's likelihood $p(y_t \mid x_t)$, as in eq. (3.5), we then recover *filtered* estimates of the state variables. This approach is widely used in online tracking applications, where causal processing of an observation sequence is required.

As discussed in Sec. 2.3.2, analytic evaluation of BP's message update integral is typically intractable for non–linear or non–Gaussian dynamical systems. For high–dimensional state spaces, like those arising in visual tracking problems, fixed discretizations of $\mathcal{X}_t$ are also computationally infeasible. In these applications, *particle filters* [11, 70, 72, 183] provide a popular method of approximate inference. In their simplest form, particle filters approximate the forward BP messages via a collection of L weighted samples, or particles:

$$m_{t-1,t}(x_t) \approx \sum_{\ell=1}^{L} w_{t-1,t}^{(\ell)} \delta(x_t, x_t^{(\ell)}) \qquad \sum_{\ell=1}^{L} w_{t-1,t}^{(\ell)} = 1 \tag{3.6}$$

Importance sampling methods (see Sec. 2.4.1) are then used to update the particles $x_t^{(\ell)}$, and corresponding weights $w_{t-1,t}^{(\ell)}$, as the dynamical system evolves. Given these sample–based density estimates, any statistic $f_t(x_t)$ of the filtering distribution may be approximated as in eq. (2.141).

Particle filters were independently proposed in several different research communities, and variants are alternatively known as *bootstrap filters* [119], the *condensation* algorithm [146], and *survival of the fittest* [165]. However, all of these *sequential Monte Carlo* methods [70, 72] share the use of importance sampling, possibly coupled with additional MCMC iterations [114, 183], to recursively update nonparametric density estimates. In the following sections, we describe the basic structure of particle filters, and discuss various extensions.

■ 3.1.1 Sequential Importance Sampling

Examining the message update integral of eq. (3.2), we see that it can be conceptually decomposed into two stages. First, the measurement update $p(y_t \mid x_t)\, m_{t-1,t}(x_t)$ combines information from preceding observations with the evidence provided by the new observation y_t. The resulting posterior distribution (see eq. (3.5)) is then convolved with the state transition density $p(x_{t+1} \mid x_t)$, optimally propagating information to subsequent times. Particle filters stochastically approximate these two stages, and thereby compute consistent nonparametric estimates of the exact filtering distributions.

Throughout this section, we use $m_{t-1,t}(x_t)$ to denote a sample–based approximation, as in eq. (3.6), of the exact BP message function. We then let $q_{\overline{t}}(x_t)$ indicate a corresponding nonparametric estimate of the filtering distribution $p(x_t \mid y_{\overline{t}})$ (see eq. (3.5)).

Measurement Update

Suppose that $m_{t-1,t}(x_t)$, the BP message from the preceding point in time, is represented by L weighted samples as in eq. (3.6). At time $t = 0$, the algorithm may be initialized by drawing L independent samples $x_0^{(\ell)} \sim p(x_0)$ from the prior. From eq. (3.5), the posterior distribution of x_t then equals

$$q_{\overline{t}}(x_t) \propto m_{t-1,t}(x_t)\, p(y_t \mid x_t) \propto \sum_{\ell=1}^{L} w_{t-1,t}^{(\ell)}\, p(y_t \mid x_t^{(\ell)})\, \delta(x_t, x_t^{(\ell)}) \tag{3.7}$$

Normalizing these importance weights, which are determined by evaluating the likelihood of y_t with respect to each particle, we then have

$$q_{\overline{t}}(x_t) = \sum_{\ell=1}^{L} w_t^{(\ell)} \delta(x_t, x_t^{(\ell)}) \qquad\qquad w_t^{(\ell)} \triangleq \frac{w_{t-1,t}^{(\ell)}\, p(y_t \mid x_t^{(\ell)})}{\sum_{m=1}^{L} w_{t-1,t}^{(m)}\, p(y_t \mid x_t^{(m)})} \tag{3.8}$$

This update equation is motivated by the general importance sampling framework described in Sec. 2.4.1. In particular, if $m_{t-1,t}(x_t)$ defines an unbiased estimate of $p\big(x_t \mid y_{\overline{t-1}}\big)$, it is easily shown [72, 183] that $q_{\overline{t}}(x_t)$ also leads to unbiased importance estimates for statistics of $p(x_t \mid y_{\overline{t}})$.

Sample Propagation

Given a particle–based filtering distribution $q_{\overline{t}}(x_t)$, we may predict likely values of subsequent states by simulating the dynamical system underlying the HMM. In the simplest case, particle filters use this approach to compute outgoing messages:

$$m_{t,t+1}(x_{t+1}) = \sum_{\ell=1}^{L} w_{t,t+1}^{(\ell)} \delta(x_{t+1}, x_{t+1}^{(\ell)}) \qquad \begin{aligned} x_{t+1}^{(\ell)} &\sim p(x_{t+1} \mid x_t^{(\ell)}) \\ w_{t,t+1}^{(\ell)} &= w_t^{(\ell)} \end{aligned} \tag{3.9}$$

Given that $x_{t+1}^{(\ell)}$ is sampled from the prior, these weights $w_{t,t+1}^{(\ell)}$ can be justified as an importance estimate of the joint distribution $p(x_t, x_{t+1} \mid y_{\overline{t}})$. Discarding $x_t^{(\ell)}$, which by the model's Markov structure is not needed for subsequent simulation steps, we then arrive at eq. (3.9).

Depletion and Resampling

By iterating the predictions and measurement updates of eqs. (3.8, 3.9), we recursively compute unbiased estimates $q_{\overline{t}}(x_t)$ of the filtering densities $p(x_t \mid y_{\overline{t}})$. The practical reliability of this sequential Monte Carlo method depends critically on the variability of these estimates, which is characterized by the variance of the importance weights [9]. Although exact performance evaluation is typically intractable, the *effective sample size* [11, 72] can be estimated as follows:

$$L_{\text{eff}} = \left(\sum_{\ell=1}^{L} \left(w^{(\ell)} \right)^2 \right)^{-1} \tag{3.10}$$

For uniform weights $w^{(\ell)} = 1/L$, the effective sample size $L_{\text{eff}} = L$, the number of particles. In contrast, when one particle is allocated most of the posterior probability mass, $L_{\text{eff}} \to 1$.

Due to accumulation of the approximations underlying subsequent message updates, the expected variance of the importance weights increases over time [72]. For any finite sample size L, the number of effective particles L_{eff} thus approaches one after sufficiently many iterations. In practice, the resulting sample depletion is avoided via a *resampling* operation. Given the discrete filtering distribution $q_{\overline{t}}(x_t)$ of eq. (3.8), L new particles $\tilde{x}_t^{(\ell)}$ are resampled, and then propagated to subsequent timesteps:

$$\begin{aligned} \tilde{x}_t^{(\ell)} &\sim q_{\overline{t}}(x_t) \\ x_{t+1}^{(\ell)} &\sim p(x_{t+1} \mid \tilde{x}_t^{(\ell)}) \end{aligned} \qquad \ell = 1, \ldots, L \tag{3.11}$$

After such resampling, outgoing message particles are equally weighted as $w_{t,t+1}^{(\ell)} = 1/L$. Because new samples $\tilde{x}_t^{(\ell)}$ are drawn with replacement, they typically repeat those $x_t^{(m)}$ with large weights $w_t^{(m)}$ multiple times, and ignore some low weight samples entirely.

Resampling thus focuses the particle filter's computational resources on the most probable regions of the state space. For HMMs with non–degenerate dynamics, the outgoing message propagation shifts these particles to distinct values, restoring sample diversity. Mild conditions then guarantee that this iteration almost surely converges to the true filtering distribution as the number of particles L approaches infinity [51].

In many applications of particle filters, the resampling operation of eq. (3.11) is applied prior to each message update. Alternatively, the effective sample size L_{eff} can be monitored, and resampling performed whenever it drops below some pre–specified threshold [11, 72, 183]. This approach reduces the algorithm's Monte Carlo variability, and may lead to more robust state estimates [183].

■ 3.1.2 Alternative Proposal Distributions

The basic particle filter described in Sec. 3.1.1 is widely used because it is typically simple to implement. It assumes only that it is possible to simulate the underlying dynamical model, and evaluate the observations' likelihood up to some normalization constant. In some applications, however, extremely large sample sizes L are needed for adequate performance. This is particularly true in models where the prior dynamics are highly variable, as many samples are needed to robustly "guess" those states with high likelihood.

As discussed in Sec. 2.4.1, the efficiency of importance sampling algorithms can be improved via a carefully chosen proposal distribution. Applying this idea, more effective sequential Monte Carlo methods arise from integrations of the sample propagation and measurement update stages [11, 70, 72, 183]. In the general case, a filtered density estimate $q_{\overline{t}}(x_t)$ as in eq. (3.8) is propagated as follows:

$$\begin{aligned} \tilde{x}_t^{(\ell)} &\sim q_{\overline{t}}(x_t) \\ x_{t+1}^{(\ell)} &\sim r(x_{t+1} \mid \tilde{x}_t^{(\ell)}, y_{t+1}) \end{aligned} \qquad \ell = 1, \dots, L \tag{3.12}$$

Here, the proposal distribution $r(\cdot \mid \cdot)$ may also incorporate the subsequent observation y_{t+1}. To retain an unbiased importance estimate, the weights are then set as

$$w_{t+1}^{(\ell)} \propto \frac{p(y_{t+1} \mid x_{t+1}^{(\ell)})\, p(x_{t+1}^{(\ell)} \mid \tilde{x}_t^{(\ell)})}{r(x_{t+1}^{(\ell)} \mid \tilde{x}_t^{(\ell)}, y_{t+1})} \qquad \sum_{\ell=1}^{L} w_{t+1}^{(\ell)} = 1 \tag{3.13}$$

This expression specializes to the measurement update of eq. (3.8) when the proposal distribution equals the prior dynamics $p(x_{t+1} \mid x_t)$, and particles are resampled at each iteration as in eq. (3.11).

Through simple manipulations of eq. (3.13), it can be shown that $p(x_{t+1} \mid \tilde{x}_t^{(\ell)}, y_{t+1})$ provides an optimal proposal distribution, which minimizes the conditional variance of the importance weights [11, 72]. In certain special cases, such as when observations are Gaussian noise–corrupted linear functions of the state, this optimal proposal is tractable. More generally, deterministic approximations such as the extended or

unscented Kalman filter [8, 153, 162] lead to proposals which partially incorporate observations when sampling new particles [72].

In applications where the state space $\mathcal{X}_t$ has internal structure, we can often tractably marginalize some variables conditioned on another set of "difficult" variables. For example, switching linear dynamical systems [73] have state variables which are conditionally Gaussian given a discrete model–selection state. One can then design sequential Monte Carlo algorithms which only sample these difficult variables, and treat the remaining states analytically. Adapting results in Sec. 2.4.4, such *Rao–Blackwellized particle filters* [71, 72, 73, 183] are guaranteed to provide state estimates with smaller variance.

■ 3.1.3 Regularized Particle Filters

Standard particle filters rely on the mixing provided by temporal dynamics to ensure diversity among the sampled particles. In some high–dimensional tracking applications, however, these raw particles may not adequately capture the true posterior uncertainty. This is particularly true when the dynamics contain a deterministic component, or the state is augmented to also include static parameters. In some cases, MCMC moves which are invariant to the target posterior may be used to perturb particles [114, 183]. In applications like those considered in this thesis, however, *regularized particle filters* [11, 220, 325] provide a simpler, but still effective, alternative.

In a regularized particle filter, the discrete filtering distribution of eq. (3.8) is convolved with a continuous kernel function, thus constructing an explicit nonparametric density estimate [263]. For a Gaussian kernel, we then have

$$q_{\overline{t}}(x_t) = \sum_{\ell=1}^{L} w_t^{(\ell)} \mathcal{N}(x_t; x_t^{(\ell)}, \Lambda_t) \tag{3.14}$$

The covariance Λ_t is typically chosen via a standard bandwidth selection method (see Sec. 2.4.2). Given this kernel density estimate, particles are propagated with resampling as in eq. (3.11). By using a continuous kernel function, we ensure that the resampled particles are distinct, regardless of the underlying temporal dynamics. At the subsequent time step, particle weights are updated as before to incorporate the latest observation, and a bandwidth for the smoothing kernels is then selected.

Although regularized particle filters no longer provide unbiased state estimates, they are often more robust in practice. In addition, the kernel density estimate of eq. (3.14) better characterizes the state's true posterior uncertainty. We note that *Gaussian sum filters* [5, 8, 267] are based on a similar density representation, but use deterministic approximations to update and prune mixture components over time.

The algorithm just described is known as a *post–regularized* particle filter, because the particles are smoothed after the observation likelihood is incorporated. Each Gaussian kernel then receives a single, constant weight $w_t^{(\ell)}$. One can also consider *pre–regularized* particle filters [220], which use rejection sampling to implicitly multiply each kernel by the spatially varying likelihood function. However, this approach is more computationally intensive, and requires likelihoods with convenient analytic forms.

■ 3.2 Belief Propagation using Gaussian Mixtures

Although particle filters can be adapted to an extremely wide range of dynamical models and observation types, they are specialized to the structure of temporal filtering problems. Conversely, loopy belief propagation can in principle be applied to graphs of any structure, but is only analytically tractable when all hidden variables are discrete or jointly Gaussian. In this chapter, we propose a *nonparametric belief propagation* (NBP) algorithm [277] which generalizes sequential Monte Carlo methods to arbitrary graphs. As in regularized particle filters, we approximate the true BP messages and beliefs by nonparametric density estimates. Importance sampling methods, combined with efficient local MCMC iterations, then update these sample–based messages, propagating information from local observations throughout the graph.

To simplify our exposition, we focus on pairwise Markov random fields specified by undirected graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. As described in Sec. 2.2.2, given observations y of hidden variables $x = \{x_i \mid i \in \mathcal{V}\}$, the joint distribution takes the following form:

$$p(x \mid y) \propto p(x, y) \propto \prod_{(i,j)\in\mathcal{E}} \psi_{ij}(x_i, x_j) \prod_{i\in\mathcal{V}} \psi_i(x_i, y) \tag{3.15}$$

The general form of the BP algorithm for such models is summarized in Fig. 2.16. As we discuss in Sec. 3.6, however, the computational methods underlying NBP could also be adapted to message–passing in directed or factor graphs [175, 178, 231, 324].

In this section, we assume that the potential functions defining the pairwise MRF of eq. (3.15) are finite Gaussian mixtures (see Sec. 2.2.4). Such representations arise naturally from learning–based approaches to model identification [95]. Sec. 3.3 then generalizes NBP to accomodate potentials specified by more general analytic functions.

■ 3.2.1 Representation of Messages and Beliefs

As derived in Sec. 2.3.2, BP approximates the posterior marginal distribution $p(x_i \mid y)$ via the following belief update equation:

$$q_i(x_i) \propto \psi_i(x_i, y) \prod_{j\in\Gamma(i)} m_{ji}(x_i) \tag{3.16}$$

Directly generalizing the particle filter of Sec. 3.1.1, suppose that incoming messages $m_{ji}(x_i)$ are represented by a set of discrete samples, as in eq. (3.9). If $x_i \in \mathcal{X}_i$ takes values in a continuous sample space, and these messages are constructed from independent, absolutely continuous proposal distributions, their particles will be distinct with probability one. The product of eq. (3.16) is then *guaranteed* to be zero everywhere, and thus provide an invalid belief estimate.

In practice, importance sampling methods like particle filters implicitly assume that the target distribution is well approximated by its values on a finite sample set. Making the regularities inherent in this assumption explicit, we construct smooth, strictly

positive messages by convolving raw particle sets with a Gaussian kernel:

$$m_{ji}(x_i) = \sum_{\ell=1}^{L} w_{ji}^{(\ell)} \mathcal{N}(x_i; x_{ji}^{(\ell)}, \Lambda_{ji}) \tag{3.17}$$

As we discuss later, we use methods from the nonparametric density estimation literature to construct these estimates [263]. As in regularized particle filters [11, 220, 325], belief estimates are similarly approximated as

$$q_i(x_i) = \sum_{\ell=1}^{L} w_i^{(\ell)} \mathcal{N}(x_i; x_i^{(\ell)}, \Lambda_i) \tag{3.18}$$

Gaussian kernels are strictly positive, and thus ensure that message products are always non–degenerate. In addition, the product of two Gaussians is itself a scaled Gaussian distribution, a fact which facilitates our later computational methods. For these reasons, we focus exclusively on density estimates derived from mixtures of Gaussians.

The NBP algorithm is based on Monte Carlo methods which sequentially update nonparametric message approximations as in eq. (3.17). For notational simplicity, we let $m_{ji}(x_i)$ and $q_i(x_i)$ denote sample–based estimates as in eqs. (3.17, 3.18), which we do not explicitly distinguish from the "ideal" outputs of integral BP message updates.

■ 3.2.2 Message Fusion

We begin by considering the BP belief update of eq. (3.16), where incoming messages are specified nonparametrically as in eq. (3.17). Combined with the observation potential $\psi_i(x_i, y)$, the belief $q_i(x_i)$ is then defined by a product of $d = (|\Gamma(i)| + 1)$ Gaussian mixtures. As detailed in Sec. 3.4.1, the product of d Gaussian mixtures, each containing L components, is itself a mixture of L^d Gaussians. In principle, this belief update could thus be performed exactly. In practice, however, exponential growth in the number of mixture components necessitates approximations.

As suggested by the example of Fig. 3.1, products of Gaussian mixtures often take a simple form, which can be well approximated by far fewer components. If the product mixture were explicitly available, a wide range of methods [116, 141] could be used to construct reduced order models. For practical sample sizes, however, enumeration of all L^d components is intractable. One alternative method begins by simplifying the product of a single pair of Gaussian mixtures, and then successively approximates that density's product with other mixtures [94]. However, this approach is sensitive to the order in which mixtures are incorporated, and can sometimes lead to significant errors.

The NBP algorithm takes a simpler approach, approximating the product mixture via a collection of L independent samples:

$$x_i^{(\ell)} \sim \frac{1}{Z_i} \psi_i(x_i, y) \prod_{j \in \Gamma(i)} m_{ji}(x_i) \qquad \ell = 1, \ldots, L \tag{3.19}$$

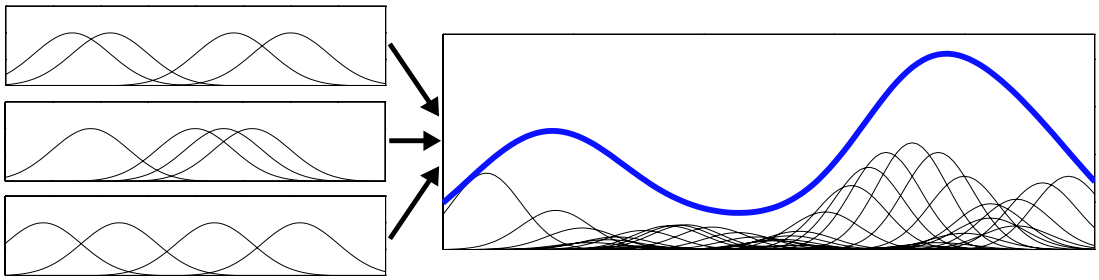


Figure 3.1. A product of three mixtures of $L = 4$ one–dimensional Gaussian distributions. Although the $4^3 = 64$ components in the product density (thin lines) vary widely in their position and weight (rescaled for visibility), their normalized sum (thick line) has a simple form.

Given these samples, the nonparametric belief estimate of eq. (3.18) is constructed via one of the automatic bandwidth selection methods described in Sec. 2.4.2. For sufficiently large L, standard density estimation asymptotics [263] guarantee that these samples accurately characterize the true product distribution. Sec. 3.4 develops efficient Monte Carlo methods which draw such samples without explicitly constructing the corresponding product density.

■ 3.2.3 Message Propagation

For pairwise Markov random fields, BP messages are updated according to the following integral equation:

$$m_{ji}(x_i) \propto \int_{\mathcal{X}_j} \psi_{ij}(x_i, x_j)\, \psi_j(x_j, y) \prod_{k \in \Gamma(j) \setminus i} m_{kj}(x_j)\, dx_j \tag{3.20}$$

Conceptually, this message update naturally decomposes into two stages. First, the message product operation combines information from neighboring nodes with the local observation potential:

$$q_{j \setminus i}(x_j) \propto \psi_j(x_j, y) \prod_{k \in \Gamma(j) \setminus i} m_{kj}(x_j) \tag{3.21}$$

This *partial belief estimate* summarizes all available evidence about x_j, excluding the incoming message from node i. Convolving these beliefs with the pairwise potential relating x_j to x_i, we then have

$$m_{ji}(x_i) \propto \int_{\mathcal{X}_j} \psi_{ij}(x_i, x_j)\, q_{j \setminus i}(x_j)\, dx_j \tag{3.22}$$

The NBP algorithm stochastically approximates these two stages, and thus provides a consistent nonparametric estimate (as in eq. (3.17)) of the outgoing message.

Algorithm 3.1 summarizes the general form of NBP message updates, including extensions which handle analytic potential functions (see Sec. 3.3). To derive this

Given input messages $m_{kj}(x_j)$ for each $k \in \Gamma(j)\backslash i$, which may be either analytic functions or kernel densities $m_{kj}(x_j) = \{x_{kj}^{(\ell)}, w_{kj}^{(\ell)}, \Lambda_{kj}^{(\ell)}\}_{\ell=1}^{L}$, construct an output message $m_{ji}(x_i)$ as follows:

1. Determine the marginal influence function $\varphi_{ij}(x_j)$ defined by eq. (3.26):
   (a) If $\psi_{ij}(x_i, x_j)$ is a Gaussian mixture, $\varphi_{ij}(x_j)$ is the marginal distribution of x_j.
   (b) For analytic $\psi_{ij}(x_i, x_j)$, determine $\varphi_{ij}(x_j)$ by symbolic or numeric integration.
2. Draw L independent, importance weighted samples from the product distribution
$$(\tilde{x}_j^{(\ell)}, \tilde{w}_j^{(\ell)}) \sim \frac{1}{Z_j}\varphi_{ij}(x_j)\,\psi_j(x_j, y) \prod_{k\in\Gamma(j)\backslash i} m_{kj}(x_j) \qquad \ell = 1, \ldots, L$$
   As in Sec. 3.4, sample particles $\tilde{x}_j^{(\ell)}$ from the product of those terms which are Gaussian mixtures, and evaluate the remaining terms to determine importance weights $\tilde{w}_j^{(\ell)}$.
3. Evaluate the effective sample size L_{eff} of these auxiliary particles via eq. (3.10). If L_{eff} is small, resample by drawing L particles with replacement according to $\Pr[\tilde{x}_j^{(\ell)}] \propto \tilde{w}_j^{(\ell)}$.
4. If $\psi_{ij}(x_i, x_j)$ satisfies the normalization condition of eq. (3.31), construct a kernel–based output message $m_{ji}(x_i) = \{x_{ji}^{(\ell)}, w_{ji}^{(\ell)}, \Lambda_{ji}^{(\ell)}\}_{\ell=1}^{L}$ as follows:
   (a) For each auxiliary particle $\tilde{x}_j^{(\ell)}$, sample an outgoing particle
$$x_{ji}^{(\ell)} \sim \psi_{ij}(x_i, x_j = \tilde{x}_j^{(\ell)}) \qquad \ell = 1, \ldots, L$$
   For Gaussian mixture potentials, we sample from the conditional distribution of x_i given $\tilde{x}_j^{(\ell)}$. More generally, importance sampling or MCMC methods may be used.
   (b) Set $w_{ji}^{(\ell)}$ to account for any importance weights generated by steps 2, 3, & 4(a).
   (c) Choose $\{\Lambda_{ji}^{(\ell)}\}_{\ell=1}^{L}$ using any appropriate bandwidth selection method (see [263]).
5. Otherwise, treat $m_{ji}(x_i)$ as an analytic function in subsequent message updates:
$$m_{ji}(x_i) \propto \sum_{\ell=1}^{L} \tilde{w}_j^{(\ell)} \psi_{ij}(x_i, \tilde{x}_j^{(\ell)})$$
   In this case, $m_{ji}(x_i)$ is parameterized by the auxiliary particles $\{\tilde{x}_j^{(\ell)}, \tilde{w}_j^{(\ell)}\}_{\ell=1}^{L}$.

Algorithm 3.1. Nonparametric BP update for the message $m_{ji}(x_i)$ sent from node j to node i as in eq. (3.20). Input and output messages may be either kernel density estimates or analytic functions, depending on the form of the corresponding pairwise potentials.

algorithm, we first decompose the pairwise potential function $\psi_{ij}(x_i, x_j)$, separating its *marginal* influence on x_j from the *conditional* relationship it defines between x_i and x_j. We then propose Monte Carlo approximations to the belief fusion and propagation updates of eqs. (3.21, 3.22).

Pairwise Potentials and Marginal Influence

In hidden Markov models as in eq. (3.1), the joint distribution of the hidden states x is factored via a series of one–step transition distributions $p(x_{t+1} \mid x_t)$. The particle filter of Sec. 3.1.1 then propagates belief estimates to subsequent time steps by sam-

pling $x_{t+1}^{(\ell)} \sim p(x_{t+1} \mid x_t^{(\ell)})$. The consistency of this procedure depends critically on the HMM's factorization into properly normalized conditional distributions, so that

$$\int_{\mathcal{X}_{t+1}} p(x_{t+1} \mid x_t)\, dx_t = 1 \qquad \text{for all } x_t \in \mathcal{X}_t \tag{3.23}$$

By definition, this conditional distribution places no biases on x_t. In contrast, for pairwise MRFs the clique potential $\psi_{ij}(x_i, x_j)$ is an arbitrary non–negative function. It may thus strongly influence the values assumed by either associated hidden variable.

To develop a consistent propagation scheme for arbitrary pairwise potentials, we consider the derivation of the BP message update equation presented in Sec. 2.3.2. As illustrated in Fig. 2.16, the BP message $m_{ji}(x_i)$ arises as a marginal of the following joint distribution:

$$\tilde{q}_{ij}(x_i, x_j) \propto \psi_{ij}(x_i, x_j)\, q_{j\setminus i}(x_j) \tag{3.24}$$

We can thus approximate the outgoing message by marginalizing joint samples $(x_{ji}^{(\ell)}, \tilde{x}_j^{(\ell)})$ drawn according to eq. (3.24). Here, we denote $x_{ji}^{(\ell)} \in \mathcal{X}_i$ by double subscripts to indicate that it is distributed as $x_{ji}^{(\ell)} \sim m_{ji}(x_i)$, rather than according to the marginal belief at node i. To draw these samples, we consider the marginal distribution of x_j:

$$\tilde{q}_j(x_j) \propto \int_{\mathcal{X}_i} \psi_{ij}(x_i, x_j)\, q_{j\setminus i}(x_j)\, dx_i \propto \left[\int_{\mathcal{X}_i} \psi_{ij}(x_i, x_j)\, dx_i\right] q_{j\setminus i}(x_j) \tag{3.25}$$

The *marginal influence* of $\psi_{ij}(x_i, x_j)$ on x_j is then quantified by the following function:

$$\varphi_{ij}(x_j) = \int_{\mathcal{X}_i} \psi_{ij}(x_i, x_j)\, dx_i \tag{3.26}$$

If $\psi_{ij}(x_i, x_j)$ is specified by a Gaussian mixture, $\varphi_{ij}(x_j)$ is simply the mixture attained by marginalizing each component. We discuss more general cases in Sec. 3.3.3.

Marginal and Conditional Sampling

Given the preceding analysis, NBP approximates the message update of eq. (3.20) in two stages. We first draw L independent samples $\tilde{x}_j^{(\ell)}$ from the partial belief estimate of eq. (3.21), reweighted by the marginal influence function of eq. (3.26):

$$\tilde{x}_j^{(\ell)} \sim \frac{1}{Z_j} \varphi_{ij}(x_j)\, q_{j\setminus i}(x_j) = \frac{1}{Z_j} \varphi_{ij}(x_j)\, \psi_j(x_j, y) \prod_{k \in \Gamma(j)\setminus i} m_{kj}(x_j) \tag{3.27}$$

For clique potentials which are mixtures of Gaussians, this density is a product of Gaussian mixtures, similar to that arising in the belief update of eq. (3.19). Samples may thus be efficiently drawn via the Monte Carlo methods of Sec. 3.4. We then propagate these auxiliary particles to node i via the pairwise clique potential:

$$x_{ji}^{(\ell)} \sim \frac{1}{Z_i^\ell}\, \psi_{ij}(x_i, x_j = \tilde{x}_j^{(\ell)}) \qquad\qquad Z_i^\ell = \int_{\mathcal{X}_i} \psi_{ij}(x_i, x_j = \tilde{x}_j^{(\ell)})\, dx_i \tag{3.28}$$

This message update treats the clique potential as a joint distribution, and samples particles $x_{ji}^{(\ell)} \in \mathcal{X}_i$ from the conditional densities induced by each fixed auxiliary particle $\tilde{x}_j^{(\ell)}$. Because the clique potential's marginal influence on x_j is incorporated into eq. (3.27), these particles consistently approximate the target message $m_{ji}(x_i)$. When $\psi_{ij}(x_i, x_j)$ is a mixture of K Gaussians, the conditional distribution of eq. (3.28) is another, reweighted K–component mixture.

Bandwidth Selection

Given L particles $x_{ji}^{(\ell)}$ sampled as in eq. (3.28), a nonparametric estimate of the updated message $m_{ji}(x_i)$ is constructed via eq. (3.17). In applications considered by this thesis, we determine the bandwidth Λ_{ji} of this density estimate via one of the automatic methods described in Sec. 2.4.2. When $\psi_{ij}(x_i, x_j)$ is a Gaussian mixture with few components, alternative methods are possible which replace the Monte Carlo propagation of eq. (3.28) by a set of deterministically placed kernels [141, 145, 261, 262].

■ 3.2.4 Belief Sampling Message Updates

As illustrated in Fig. 2.16, the BP update of message $m_{ji}(x_i)$ is most often expressed as a transformation of the incoming messages from all *other* neighboring nodes $k \in \Gamma(j)\backslash i$. In some situations, however, it is useful to consider the following alternative form:

$$m_{ji}(x_i) \propto \int_{\mathcal{X}_j} \psi_{ij}(x_i, x_j)\, \psi_j(x_j, y) \prod_{k \in \Gamma(j)\backslash i} m_{kj}(x_j)\, dx_j \tag{3.29}$$

$$\propto \int_{\mathcal{X}_j} \psi_{ij}(x_i, x_j)\, \frac{q_j(x_j)}{m_{ij}(x_j)}\, dx_j \tag{3.30}$$

If the message update integral is evaluated exactly, the equivalence of these expressions follow directly from eq. (3.16). However, eq. (3.30) suggests an alternative *belief sampling* form of the NBP algorithm, in which the latest belief estimate provides a proposal distribution for auxiliary particles $\tilde{x}_j^{(\ell)} \sim q_j(x_j)$. As summarized in Alg. 3.2, overcounting of the incoming message $m_{ij}(x_j)$ may be avoided via importance weights $\tilde{w}_j^{(\ell)} \propto 1/m_{ij}(\tilde{x}_j^{(\ell)})$. These weighted samples are then propagated as in the *message sampling* NBP algorithm described in Sec. 3.2.3.

Computationally, the belief sampling form of the NBP algorithm offers clear advantages. In particular, for a node with d neighbors, computation of new outgoing messages via Alg. 3.1 requires L samples drawn from each of d different products of $(d-1)$ Gaussian mixtures. Depending on the selected sampling algorithm (see Sec. 3.4), these sampling updates typically require either $\mathcal{O}(d^2L)$ or $\mathcal{O}(d^2L^2)$ operations. In contrast, belief sampling NBP updates share a single set of L samples, drawn from a product of d Gaussian mixtures, among all outgoing messages (Alg. 3.2, steps 1–2). The computational cost is then reduced to $\mathcal{O}(dL)$ or $\mathcal{O}(dL^2)$ operations, a potentially substantial

Given input messages $m_{kj}(x_j)$ for every $k \in \Gamma(j)$, which may be either analytic functions or kernel densities $m_{kj}(x_j) = \{x_{kj}^{(\ell)}, w_{kj}^{(\ell)}, \Lambda_{kj}^{(\ell)}\}_{\ell=1}^{L}$, construct an output message $m_{ji}(x_i)$ as follows:

1. Draw L independent, importance weighted samples from the product distribution

$$(\tilde{x}_j^{(\ell)}, \tilde{w}_j^{(\ell)}) \sim \frac{1}{Z_j}\psi_j(x_j, y) \prod_{k \in \Gamma(j)} m_{kj}(x_j) \qquad \ell = 1, \ldots, L$$

   As in Sec. 3.4, sample particles $\tilde{x}_j^{(\ell)}$ from the product of those terms which are Gaussian mixtures, and evaluate the remaining terms to determine importance weights $\tilde{w}_j^{(\ell)}$.

2. Evaluate the effective sample size L_{eff} of these auxiliary particles via eq. (3.10). If L_{eff} is small, construct a kernel–based density estimate $q_j(x_j) = \{\tilde{x}_j^{(\ell)}, \tilde{w}_j^{(\ell)}, \Lambda_j^{(\ell)}\}_{\ell=1}^{L}$ via any bandwidth selection method (see [263]), and draw L new auxiliary particles from $q_j(x_j)$.

3. Determine the marginal influence function $\varphi_{ij}(x_j)$ defined by eq. (3.26):

   (a) If $\psi_{ij}(x_i, x_j)$ is a Gaussian mixture, $\varphi_{ij}(x_j)$ is the marginal distribution of x_j.

   (b) For analytic $\psi_{ij}(x_i, x_j)$, determine $\varphi_{ij}(x_j)$ by symbolic or numeric integration.

4. Reweight the auxiliary particles from steps 1–2 as follows:

$$\tilde{w}_j^{(\ell)} \leftarrow \tilde{w}_j^{(\ell)} \cdot \frac{\varphi_{ij}\left(\tilde{x}_j^{(\ell)}\right)}{m_{ij}\left(\tilde{x}_j^{(\ell)}\right)}$$

5. Using these weighted auxiliary particles $\{\tilde{x}_j^{(\ell)}, \tilde{w}_j^{(\ell)}\}_{\ell=1}^{L}$, determine an outgoing message $m_{ji}(x_i)$ as in steps 4–5 of Alg. 3.1.

Algorithm 3.2. Belief sampling variant of the nonparametric BP update for the message $m_{ji}(x_i)$ sent from node j to node i as in eq. (3.30). Input and output messages may be either kernel density estimates or analytic functions, depending on the form of the corresponding pairwise potentials. If desired, the auxiliary particles sampled in steps 1–2 may be reused when updating outgoing messages $m_{ji}(x_i)$ for several neighbors $i \in \Gamma(j)$.

savings even for moderately sized neighborhoods. Belief sampling offers further advantages in distributed implementations of NBP, where communications constraints favor algorithms which share a common message among all neighboring nodes [142].

Statistically, the relative merits of the message and belief sampling forms of NBP are less clear. When discussing algorithms related to NBP, several authors have argued that belief sampling better concentrates particles in important regions of the state space [145, 171]. In particular, consider the sequences of forward and backward messages computed when applying BP to an HMM. Using the message sampling form of NBP, these messages are computed independently, and future observations cannot affect messages sent from past times. In contrast, belief sampling NBP updates allow the latest belief estimates to guide forward message construction, focusing particles on those states most relevant to future observations. This approach is heuristically similar to the expectation propagation (EP) algorithm [135, 213], which iteratively projects belief estimates, rather than messages, to tractable exponential families.

While these arguments are intuitively compelling, the empirical behavior of NBP is

more subtle. In particular, when NBP uses few particles L to approximate messages, errors in early belief estimates may inappropriately bias the messages computed by Alg. 3.2, and lead to inferior performance [142]. Later sections discuss this issue, and potential solutions, in more detail.

3.3 Analytic Messages and Potentials

In many of the applications motivating the NBP algorithm, the graphical model's potentials are not Gaussian mixtures, but more general analytic functions. Adapting importance sampling methods, we now extend the NBP updates of Sec. 3.2 to this case. As with particle filters, we assume that observation potentials $\psi_i(x_i, y)$ can be evaluated, up to some normalization constant, at any particle location $x_i^{(\ell)}$. We discuss analogous conditions for pairwise potentials $\psi_{ij}(x_i, x_j)$ in Sec. 3.3.3.

Particle–based representations are only meaningful when the corresponding BP messages $m_{ji}(x_i)$ are finitely integrable. As discussed in Sec. 3.1, the causal factorization underlying HMMs leads to messages which take the form of conditional densities (see eq. (3.4)). However, for more general parameterizations like that of Prop. 2.3.1, BP messages instead equal likelihood functions [276]. In such models, uninformative or weakly informative messages with infinite measure are possible.

In the simplest case, we assume that the pairwise MRF of eq. (3.15) is parameterized by clique potentials satisfying

$$\int_{\mathcal{X}_i} \psi_{i,j}(x_i, x_j = \bar{x})\, dx_i < \infty \qquad \text{for all } (i,j) \in \mathcal{E} \tag{3.31}$$

$$\int_{\mathcal{X}_i} \psi_i(x_i, y = \bar{y})\, dx_i < \infty \qquad \text{for all } i \in \mathcal{V} \tag{3.32}$$

A simple induction argument then shows that all messages are finitely integrable, and hence *normalizable*. Heuristically, eqs. (3.31, 3.32) require each potential to be sufficiently informative to localize variables given observations of their neighbors. In many applications, these conditions are easily satisfied by constraining all variables to a (possibly large) bounded range. The following sections also describe methods for constructing analytic NBP messages, which relax these normalization conditions.

3.3.1 Representation of Messages and Beliefs

As motivated in Sec. 3.2, we again estimate beliefs via a weighted mixture of Gaussian kernels (eq. (3.18)). For NBP messages, however, we consider two complementary representations. As before, messages corresponding to edges satisfying the normalization condition of eq. (3.31) are described nonparametrically (eq. (3.17)). In some models, however, there are weakly informative potentials which are not normalizable, but nevertheless encode important constraints. For example, in the NBP hand tracker developed in Chap. 4, pairwise potentials are used to prevent two fingers from occupying the same three–dimensional space. Related potentials, which encode the constraint that certain

hidden variables do *not* take similar values, also arise in algorithms for distributed sensor network localization [141, 142].

In high–dimensional spaces, mixture models provide an extremely inefficient representation of messages encoding such *dissimilarities*. The NBP algorithm thus describes these messages implicitly, as analytic functions. As we show next, belief updates are then feasible assuming these message functions are easily evaluated.

■ 3.3.2 Message Fusion

Consider the marginal belief estimate $q_i(x_i)$, which is determined according to eq. (3.16). Of the $d = (|\Gamma(i)| + 1)$ messages and potentials composing this product, some are represented by Gaussian mixtures, and others via analytic functions. Using the efficient algorithms developed in Sec. 3.4, we first draw L samples $x_i^{(\ell)}$ from the product distribution determined solely by the Gaussian mixtures. Any analytic messages or potentials are then evaluated at these particle locations, providing importance weights $w_i^{(\ell)}$. Finally, a bandwidth selection method (see Sec. 2.4.2) is used to construct a nonparametric density estimate, as in eq. (3.18).

This belief update procedure assumes that at least one message or observation potential is a Gaussian mixture, to provide a proposal distribution for importance sampling. In many applications, this can be guaranteed via an appropriate message schedule. More generally, an application–specific importance distribution could be used. For high–dimensional problems, however, good proposal design can be extremely challenging. Indeed, one of NBP's strengths is that message proposal distributions are automatically refined as information propagates thoughout the graph.

■ 3.3.3 Message Propagation

To update the NBP message $m_{ji}(x_i)$ associated with an analytic pairwise potential, we must determine the marginal influence function $\varphi_{ij}(x_j)$ of eq. (3.26). In many applications, pairwise potentials depend only on the difference in their arguments, so that $\psi_{ij}(x_i, x_j) = \widetilde{\psi}_{ij}(x_i - x_j)$. For such models, the marginal influence function is constant and may be neglected:

$$\varphi_{ij}(x_j) = \int_{\mathcal{X}_i} \widetilde{\psi}_{ij}(x_i - x_j)\, dx_i = \int_{\mathcal{X}_i} \widetilde{\psi}_{ij}(x_i)\, dx_i \qquad \text{for all } x_j \in \mathcal{X}_j \tag{3.33}$$

More generally, symbolic or numeric integration may be required, depending on the particular form of $\psi_{ij}(x_i, x_j)$.

Given this influence function, we first draw L auxiliary samples $\tilde{x}_j^{(\ell)}$ according to the product distribution of eq. (3.27). As in Sec. 3.3.2, we use the product of those messages which are Gaussian mixtures as a proposal distribution, and reweight by any analytic messages, observation potentials, or marginal influences. For normalizable pairwise potentials, we then sample outgoing message particles via the conditional distribution of eq. (3.28). If needed, additional importance sampling or MCMC methods may be

used in this propagation step. Finally, a bandwidth Λ_{ji} is determined as described in Sec. 2.4.2, producing the nonparametric message estimate of eq. (3.17). See Alg. 3.1 for a summary of these computations, which also includes an optional resampling update (step 3) to avoid propagating low–weight auxiliary particles.

If $\psi_{ij}(x_i, x_j)$ does not satisfy the normalization condition of eq. (3.31), we instead treat $m_{ji}(x_i)$ as an analytic function. Using L auxiliary particles $\tilde{x}_j^{(\ell)}$ sampled as before, with associated importance weights $\tilde{w}_j^{(\ell)}$, the outgoing message equals

$$m_{ji}(x_i) \propto \int_{\mathcal{X}_j} \psi_{ij}(x_i, x_j) \left[\sum_{\ell=1}^{L} \tilde{w}_j^{(\ell)} \delta(x_j, \tilde{x}_j^{(\ell)}) \right] dx_j \propto \sum_{\ell=1}^{L} \tilde{w}_j^{(\ell)} \psi_{ij}(x_i, \tilde{x}_j^{(\ell)}) \tag{3.34}$$

In subsequent message and marginal updates, we evaluate this analytic message to determine importance weights for each candidate value of x_i.

We note that in principle, eq. (3.34) could be used to treat every NBP message analytically. However, we would then have the difficult task of hand–designing proposal distributions for each message and belief update. By instead representing most messages as Gaussian mixtures, we develop efficient sampling algorithms which directly fuse information, and avoid unnecessary reliance on potentially inaccurate importance sampling schemes.

■ 3.3.4 Belief Sampling Message Updates

The preceding approach to analytic message updates may also be applied to the belief sampling form of the NBP algorithm (see Sec. 3.2.4). As summarized in Alg. 3.2, we first use importance sampling methods to draw particles from the marginal belief $q_j(x_j)$, as in Sec. 3.3.2. These auxiliary particles $\tilde{x}_j^{(\ell)}$ are then assigned weight $\tilde{w}_j^{(\ell)} \propto \varphi_{ij}(\tilde{x}_j^{(\ell)}) / m_{ij}(\tilde{x}_j^{(\ell)})$, correcting for the overcounted input message $m_{ij}(x_j)$ as well as the marginal influence of $\psi_{ij}(x_i, x_j)$. Once appropriately reweighted, the conditional propagation of these particles proceeds exactly as in Sec. 3.3.3.

■ 3.3.5 Related Work

Several authors have used junction tree representations (see Sec. 2.3.2) to develop structured approximate inference methods for general graphs. These algorithms begin by grouping nodes into cliques chosen to break the original graph's cycles. In many applications, the high dimensionality of these cliques makes exact manipulation of their true marginal distributions intractable. However, a wide range of algorithms can be developed by coupling an approximate clique representation with corresponding local computational methods [58, 131, 171]. For example, distributions over large discrete cliques can be approximated by a set of weighted point samples, and then related to neighboring nodes using standard message–passing recursions [170]. Koller et al. [171] propose a more sophisticated framework in which current marginal estimates are used to guide message computations, as in the "belief sampling" form of NBP (Alg. 3.2).

However, the sample algorithm they provide is limited to networks containing discrete or conditionally Gaussian latent variables.

The NBP algorithm differs from these previous approaches in three key ways. First, for graphs with cycles we do *not* form a junction tree, but instead iterate local message updates as in loopy BP. This advantageously leads to beliefs defined in lower–dimensional spaces, where nonparametric density estimation is more reliable. Second, NBP message updates can be adapted to graphs containing an extremely broad range of continuous, non–Gaussian potentials. The *particle message passing* (PAMPAS) algorithm [145] provides a related extension of particle filters to general graphs, which was developed concurrently with NBP. However, its message updates are specialized to graphs whose potentials are small Gaussian mixtures, rather than the general analytic functions considered by Sec. 3.3.3. More recently, particle–based message updates have also been applied in factor graphs arising in digital communications [57].

A final advantage of NBP is provided by our use of Gaussian kernel density estimates to approximate continuous beliefs. As we show in the following section, efficient multiscale sampling methods may then be used to accurately fuse the information provided by several NBP messages. In contrast, the density tree representations considered by several other approximation frameworks [171, 294] seem limited to simpler, importance sampling–based message updates.

■ 3.4 Efficient Multiscale Sampling from Products of Gaussian Mixtures

In many applications, NBP's computations are dominated by the cost of sampling from the product of several Gaussian mixtures. These mixture products are the mechanism by which NBP fuses information from different messages, and arise in the computation of both belief updates and message updates (see Alg. 3.1). In this section, we describe several algorithms for sampling from such products, including multiscale methods derived from KD–tree density representations. We conclude with an empirical comparison indicating the situations in which each approach is most effective.

We begin by considering a set of d Gaussian mixtures $\{p_1(x), \ldots, p_d(x)\}$, whose components we denote by

$$p_i(x) = \sum_{\ell_i \in \mathbb{L}_i} w_{\ell_i} \mathcal{N}(x; \mu_{\ell_i}, \Lambda_i) \tag{3.35}$$

Here, ℓ_i is an abstract *label* indexing the set $\mathbb{L}_i$ of mixture components composing $p_i(x)$, and w_{ℓ_i} equals the normalized weight of component ℓ_i. For notational simplicity, we assume that all input mixtures are of equal size L, and that the diagonal covariance matrices Λ_i are uniform within each mixture. However, the algorithms which follow may be readily extended to problems where this is not the case. Our goal is to efficiently sample from the L^d component mixture density $p(x) \propto \prod_{i=1}^{d} p_i(x)$. As in NBP message updates, we assume that L samples from this product distribution are desired.

3.4.1 Exact Sampling

The task of sampling from the product distribution can be decomposed into two stages: randomly select one of the product density's L^d components, and then draw a sample from the corresponding Gaussian. Note that there is a single Gaussian in the product density for each combination of input density labels. Throughout this section, we use lowercase script ℓ_i to label input density components, and capital script $\mathcal{L} = [\ell_1, \ldots, \ell_d]$ to indicate corresponding members of the product distribution. The relative weights assigned to components of the product distribution can be shown to equal

$$w_{\mathcal{L}} \propto \frac{\prod_{i=1}^{d} w_{\ell_i} \mathcal{N}(x; \mu_{\ell_i}, \Lambda_i)}{\mathcal{N}(x; \mu_{\mathcal{L}}, \Lambda_{\mathcal{L}})} \qquad \Lambda_{\mathcal{L}}^{-1} = \sum_{i=1}^{d} \Lambda_i^{-1} \qquad \Lambda_{\mathcal{L}}^{-1} \mu_{\mathcal{L}} = \sum_{i=1}^{d} \Lambda_i^{-1} \mu_{\ell_i} \quad (3.36)$$

where $\mu_{\mathcal{L}}$ and $\Lambda_{\mathcal{L}}$ are the mean and covariance of product component $\mathcal{L}$. The weight $w_{\mathcal{L}}$ is constant, and may be evaluated at any x (the value $x = \mu_{\mathcal{L}}$ may be numerically convenient). Furthermore, when all input mixtures share a common covariance $\Lambda_i = \Lambda$, additional simplifications are possible. To form the product distribution, these weights are normalized by the *weight partition function* $Z \triangleq \sum_{\mathcal{L}} w_{\mathcal{L}}$.

Determining Z exactly takes $\mathcal{O}(L^d)$ operations. Given this constant, we can draw L samples from the product distribution in $\mathcal{O}(L^d)$ time and $\mathcal{O}(L)$ storage. To do this, we first sample and sort L independent variables which are uniformly distributed on the unit interval. We then compute the cumulative distribution of $p(\mathcal{L}) = w_{\mathcal{L}}/Z$ to determine which, if any, samples are drawn from each component $\mathcal{L}$. Finally, samples are drawn from the chosen Gaussians, whose mean and covariance are as in eq. (3.36).

3.4.2 Importance Sampling

While the previous section's exact sampler is accurate, its exponential cost is often intractable. To develop more efficient algorithms, we consider the importance sampling framework described in Sec. 2.4.1. To draw L approximate samples from the mixture product $p(x)$, an importance sampler first draws $M > L$ auxiliary samples $\tilde{x}^{(m)}$ from some proposal distribution $r(x)$. The m^{th} sample is then given importance weight

$$\tilde{w}^{(m)} \propto \frac{p(\tilde{x}^{(m)})}{r(\tilde{x}^{(m)})} \qquad m = 1, \ldots, M \quad (3.37)$$

Normalizing these weights, L output samples $x^{(\ell)}$ are generated by taking $x^{(\ell)} = \tilde{x}^{(m)}$ with probability proportional to $\tilde{w}^{(m)}$. Note that unless $M \gg L$, this importance sampler is likely to choose some auxiliary samples multiple times.

The accuracy of such importance samplers depends critically on the chosen proposal distribution [192]. For products of Gaussian mixtures, we consider two canonical importance densities. The first, which we refer to as *mixture importance sampling*, draws each sample by randomly selecting one of the d input mixtures, and sampling from its

Given d Gaussian mixtures, where $\{\mu_{\ell_i}, w_{\ell_i}, \Lambda_i\}$ denote the components of the i^{th} mixture:

1. For each $i \in \{1, \ldots, d\}$, choose a starting label ℓ_i by sampling $p(\ell_i = \ell) \propto w_\ell$, $\ell \in \mathbb{L}_i$.
2. Fixing the input mixture labels ℓ_i, use eq. (3.36) to draw a single sample x from the corresponding component $\mathcal{L}$ of the product mixture:
$$x \sim \mathcal{N}(\mu_\mathcal{L}, \Lambda_\mathcal{L}) \qquad \mathcal{L} = [\ell_1, \ldots, \ell_d]$$
3. Fixing x, independently sample a new label ℓ_i for each $i \in \{1, \ldots, d\}$:
$$p(\ell_i = \ell \mid x) \propto w_\ell \mathcal{N}(x; \mu_\ell, \Lambda_i) \qquad \ell \in \mathbb{L}_i$$
4. Repeat steps 2–3 for T iterations, and output the sample x from the final iteration.

Algorithm 3.3. Parallel Gibbs sampling from the product of d Gaussian mixtures. Accuracy is implicitly determined by the number of sampling iterations, T. If the input mixtures have L components, each iteration requires $\mathcal{O}(Ld)$ operations.

L components. This procedure is equivalent to the following proposal distribution:

$$r(x) = \frac{1}{d} \sum_{i=1}^{d} p_i(x) \tag{3.38}$$

The importance weight of each sample $\tilde{x}^{(m)} \sim r(x)$ then equals

$$\tilde{w}^{(m)} \propto \frac{\prod_i p_i(\tilde{x}^{(m)})}{\sum_i p_i(\tilde{x}^{(m)})} \tag{3.39}$$

A similar approach was used to combine density trees in [294]. Alternatively, we can approximate each input mixture $p_i(x)$ by a single Gaussian density $r_i(x)$, and choose

$$r(x) \propto \prod_{i=1}^{d} r_i(x) \qquad r_i(x) = \mathcal{N}(x; E_{p_i}[x], \mathrm{Cov}_{p_i}[x]) \tag{3.40}$$

We call this procedure *Gaussian importance sampling*. While this proposal distribution desirably interpolates among input mixtures, we expect it to be ineffective for multimodal product distributions (see, for example, Fig. 2.17).

■ 3.4.3 Parallel Gibbs Sampling

Direct sampling from Gaussian mixture products is difficult because the distribution of product density labels $\mathcal{L}$, as defined by eq. (3.36), has a complex form. In this section, we instead consider the joint distribution of the input mixture labels ℓ_i and a corresponding output sample x:

$$p(x, \ell_1, \ldots, \ell_d) \propto \prod_{i=1}^{d} \sum_{\ell_i \in \mathbb{L}_i} w_{\ell_i} \mathcal{N}(x; \mu_{\ell_i}, \Lambda_i) \tag{3.41}$$

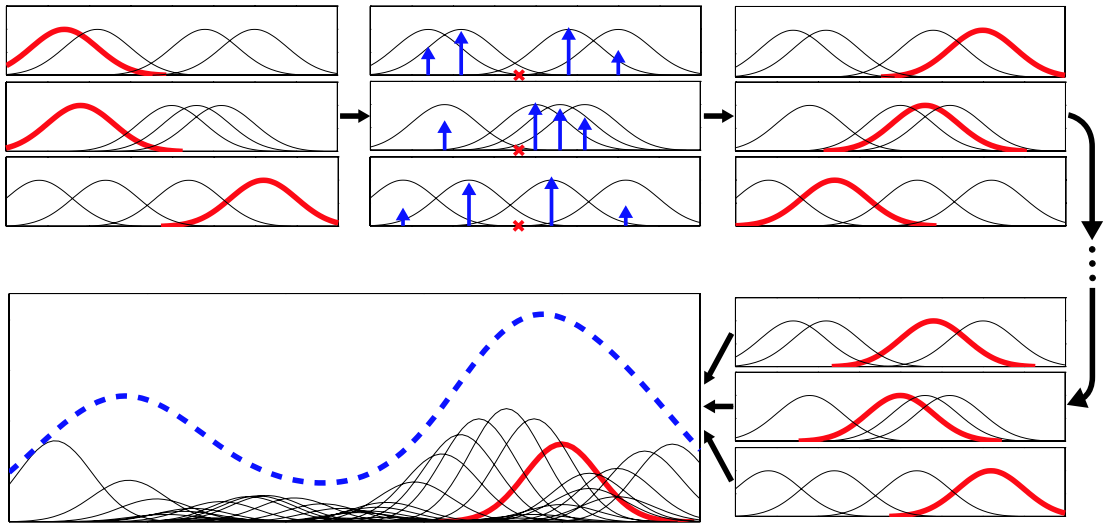


Figure 3.2. Parallel Gibbs sampling from a product of three mixtures of $L = 4$ Gaussian kernels. *Top:* Fixing the values for all mixture labels (solid red), we sample a single observation x (red, center). We then independently sample a new label ℓ_i for each mixture according to weights (arrows) determined by x. *Bottom:* After T iterations, the final labeled Gaussians for each mixture (right, solid red) are multiplied together to identify one (left, solid red) of the 4^3 components (left, thin black) of the product density (left, dashed blue).

Given fixed values for the input labels, x follows the Gaussian distribution of product component $\mathcal{L} = [\ell_1, \ldots, \ell_d]$ (see eq. (3.36)). Conversely, given x the labels are conditionally independent, with posterior distributions of the following form:

$$p(\ell_i = \ell \mid x) \propto w_\ell \mathcal{N}(x; \mu_\ell, \Lambda_i) \qquad \ell \in \mathbb{L}_i \tag{3.42}$$

As summarized by Alg. 3.3, we may thus define a *parallel Gibbs sampler* which alternates between choosing an output sample x conditioned on the current input mixture labels, and parallel sampling of the mixture labels given x.

Figure 3.2 illustrates the parallel Gibbs sampler for a product of three Gaussian mixtures. In this approach, the input labels are treated as auxiliary variables [9] which reveal tractable structure in the product distribution (see Sec. 2.4.3). Each parallel sampling iteration requires $\mathcal{O}(dL)$ operations, so that the complexity of choosing L product density samples is $\mathcal{O}(dL^2)$. We note that a similar Gibbs sampler has been used to train product of experts models [137, 138].

Although formal verification of the Gibbs sampler's convergence is difficult, our empirical results indicate that accurate Gibbs sampling typically requires far fewer computations than direct sampling. Note that NBP uses the Gibbs sampling method differently from classic simulated annealing algorithms [108]. In particular, those approaches update a single Markov chain whose state dimension is proportional to the size of the graph. In contrast, NBP uses a variational approximation to define many *local* Gibbs samplers, each involving only a few nodes.

Given d Gaussian mixtures, where $\{\mu_{\ell_i}, w_{\ell_i}, \Lambda_i\}$ denote the components of the i^{th} mixture:

1. For each $i \in \{1, \dots, d\}$, choose a starting label ℓ_i by sampling $p(\ell_i = \ell) \propto w_\ell$, $\ell \in \mathbb{L}_i$.
2. Incrementally resample the label ℓ_i associated with each input mixture $i \in \{1, \dots, d\}$:
   (a) Using eq. (3.36), determine the mean $\bar{\mu}$ and covariance $\bar{\Lambda}$ of the product distribution
   $$\mathcal{N}(x; \bar{\mu}, \bar{\Lambda}) \propto \prod_{j \neq i} \mathcal{N}(x; \mu_{\ell_j}, \Lambda_j)$$
   (b) Using any convenient x, determine the unnormalized probability of each $\ell_i \in \mathbb{L}_i$:
   $$\bar{w}_{\ell_i} = w_{\ell_i} \frac{\mathcal{N}(x; \mu_{\ell_i}, \Lambda_i)\mathcal{N}(x; \bar{\mu}, \bar{\Lambda})}{\mathcal{N}(x; \mu_{\mathcal{L}}, \Lambda_{\mathcal{L}})} \qquad \mathcal{L} = [\ell_1, \dots, \ell_d]$$
   (c) Sample a new label ℓ_i according to $p(\ell_i = \ell) \propto \bar{w}_\ell$, $\ell \in \mathbb{L}_i$.
3. Repeat step 2 for T iterations.
4. Draw a single sample $x \sim \mathcal{N}(\mu_{\mathcal{L}}, \Lambda_{\mathcal{L}})$, where $\mathcal{L} = [\ell_1, \dots, \ell_d]$ indicates the product mixture component selected on the final incremental sampling iteration.

Algorithm 3.4. Sequential Gibbs sampling from the product of d Gaussian mixtures. Accuracy is implicitly determined by the number of sampling iterations, T. If the input mixtures have L components, each iteration requires $\mathcal{O}(Ld)$ operations.

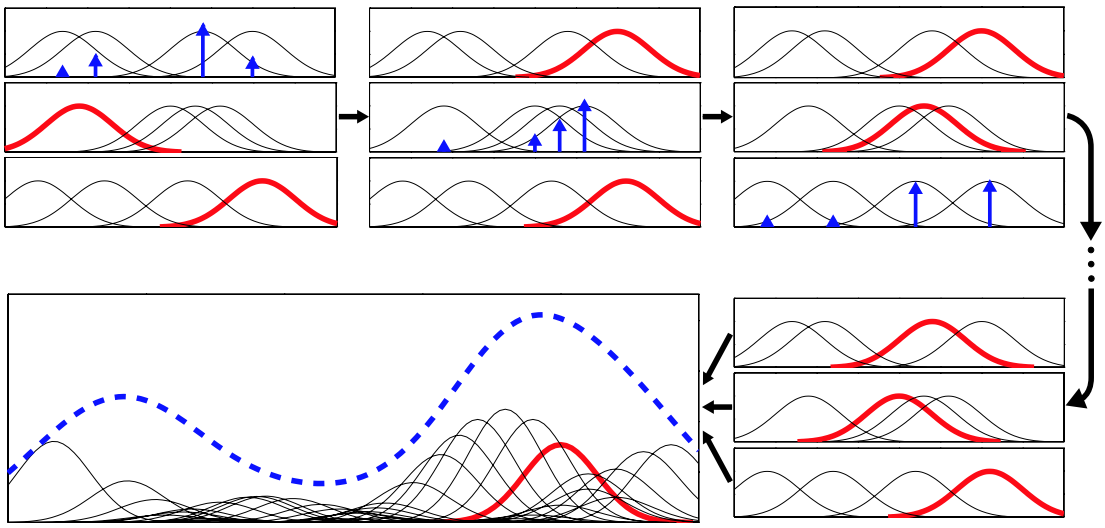


Figure 3.3. Sequential Gibbs sampling from a product of three mixtures of $L = 4$ Gaussian kernels. *Top:* Mixture labels are resampled according to weights (arrows) determined by the two fixed components (solid red). The Gibbs sampler iteratively chooses a new label ℓ_i for one density conditioned on the Gaussians selected by the other labels $\{\ell_j\}_{j \neq i}$. *Bottom:* After T iterations, the final labeled Gaussians for each mixture (right, solid red) are multiplied together to identify one (left, solid red) of the 4^3 components (left, thin black) of the product density (left, dashed blue).

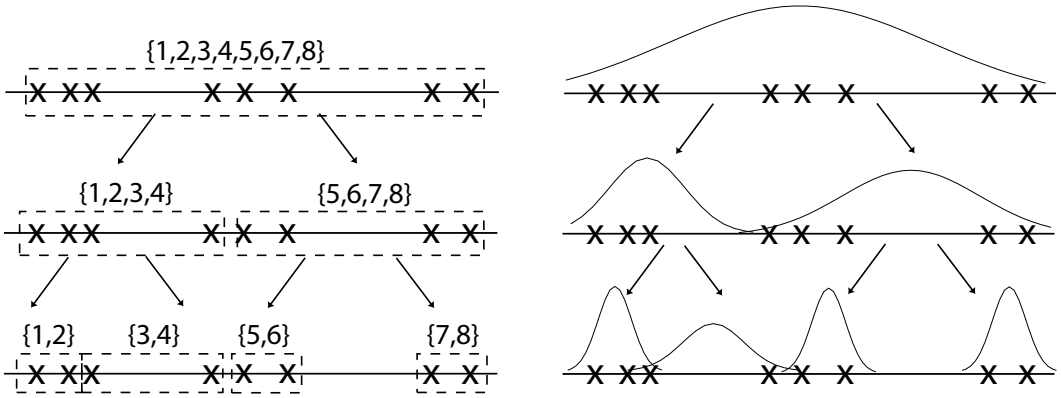


Figure 3.4. Two KD-tree representations of the same one–dimensional point set (finest scale not shown). (a) Each node maintains a bounding box containing all associated data points. Label sets $\mathfrak{l}$ are shown in braces. (b) Each node maintains mean and variance statistics for all associated data points.

■ 3.4.4 Sequential Gibbs Sampling

We can also construct an alternative, *sequential Gibbs sampler* by marginalizing the output sample x from the joint distribution of eq. (3.41). Given fixed labels $\{\ell_j \mid j \neq i\}$ for all but one input mixture, the conditional distribution of the remaining label ℓ_i is given by eq. (3.36). As detailed in Alg. 3.4, we may thus sequentially resample all d input labels in $\mathcal{O}(dL)$ operations. After a fixed number of Gibbs iterations, a single output sample x is drawn from the product mixture component $\mathcal{L} = [\ell_1, \dots, \ell_d]$ corresponding to the chosen input mixture labels.

Figure 3.3 illustrates the sequential Gibbs sampler for the same Gaussian mixture product considered in Fig. 3.2. Both Gibbs samplers require $\mathcal{O}(dL)$ operations per iteration. Note that the sequential Gibbs sampler can be interpreted as a Rao–Blackwellized [9, 39] modification of the parallel sampler (see Sec. 2.4.4). As we confirm empirically in Sec. 3.4.8, it should thus provide improved sampling accuracy.

■ 3.4.5 KD Trees

A *KD–tree* is a hierarchical data structure which caches statistics of subsets of a collection of points, thereby making later computations more efficient [19, 66]. KD–trees are typically binary trees constructed by successively splitting the data along cardinal axes, grouping points by spatial location. A variety of heuristics have been proposed for finding trees which best capture the coarse–scale structure of a given dataset. Empirically, the fast sampling algorithms developed in the following sections seem insensitive to the details of the chosen KD–tree. Our simulations employ a simple top–down construction algorithm, recursively partitioning points along the highest–variance dimension [226].

Each leaf node of the KD–tree is associated with a single observation, which as before we index by a label $\ell \in \mathbb{L}$. As illustrated in Fig. 3.4, coarser scale nodes are then associated with subsets of these observations, denoted by corresponding label sets $\mathfrak{l}$. In

the KD–tree of Fig. 3.4(a), coarse scale nodes maintain a bounding box containing the corresponding fine scale observations; similar trees are used in Sec. 3.4.7. Alternatively, the KD–tree of Fig. 3.4(b) caches the mean and variance of point clusters, providing a multi–scale Gaussian mixture representation used in Sec. 3.4.6. In either case, cached statistics may be efficiently computed via a simple bottom–up recursion [66].

■ 3.4.6 Multiscale Gibbs Sampling

Although the pair of Gibbs samplers discussed in Sec. 3.4.3 and 3.4.4 are often effective, they sometimes require a very large number of iterations to produce accurate samples. The most difficult densities are those for which there are multiple widely separated modes, each of which is associated with disjoint subsets of the input mixture labels. In this case, conditioned on a set of labels corresponding to one mode, it is very unlikely that a label or data point corresponding to a different mode will be sampled, leading to slow convergence.

Similar problems have been observed with Gibbs samplers on Markov random fields [108]. In these cases, convergence can often be accelerated by constructing a series of "coarser scale" approximate models in which the Gibbs sampler can move between modes more easily [184]. The primary challenge in developing these algorithms is to determine procedures for constructing accurate coarse scale approximations. For products of Gaussian mixtures, KD–trees provide a simple, intuitive, and easily constructed set of coarser scale models.

As in Fig. 3.4(b), we describe each input mixture by a KD–tree storing the mean and variance (biased by kernel size) of subsets of that density's Gaussian kernels. We start at the same coarse scale for all input mixtures, and perform standard Gibbs sampling on that scale's summary Gaussians. After several iterations, we condition on a data sample (as in the parallel Gibbs sampler of Alg. 3.3) to infer labels at the next finest scale. Repeating this process, we eventually arrive at the finest scale, and draw an output sample from the chosen Gaussian component of the product distribution.

Intuitively, by gradually moving from coarse to fine scales, multiscale sampling can better explore all of the product density's important modes. As the number of sampling iterations approaches infinity, multiscale samplers have the same asymptotic properties as standard Gibbs samplers. Unfortunately, there is no guarantee that multiscale sampling will improve performance. However, our simulation results indicate that it is usually very effective (see Sec. 3.4.8).

■ 3.4.7 Epsilon–Exact Sampling

As illustrated in Fig. 3.5, KD–trees may be used to efficiently bound the minimum and maximum pairwise distance separating sets of summarized observations. This approach has been previously used to develop *dual–tree* algorithms for fast evaluation of multivariate kernel density estimates [120]. In this section, we adapt these ideas to efficiently compute an approximation to the weight partition function Z. This leads to an ϵ-*exact* sampler for which a label $\mathcal{L} = [\ell_1, \ldots, \ell_d]$, with true probability $p_{\mathcal{L}}$, is guaranteed to be

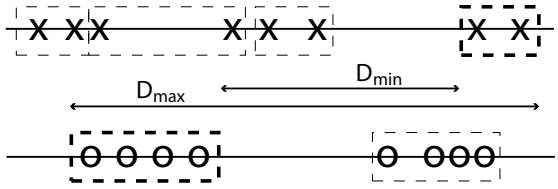


Figure 3.5. KD–tree representations of two sets of points (crosses and circles) may be combined to efficiently bound the maximum ($D_{\max}$) and minimum ($D_{\min}$) pairwise distance separating subsets (bold) of the summarized observations.

sampled with some probability $\hat{p}_{\mathcal{L}} \in [p_{\mathcal{L}} - \epsilon, p_{\mathcal{L}} + \epsilon]$.

We begin by constructing a KD–tree representation of each input mixture $p_i(x)$. As in Fig. 3.4(a), each coarse–scale node is associated with some subset $\mathfrak{l}_i$ of the observation labels ℓ_i. To implement the proposed *multi–tree* recursion, we cache bounding boxes for each set $\{\mu_{\ell_i} \mid \ell_i \in \mathfrak{l}_i\}$ of summarized input kernels, as well as their associated weight $w_{\mathfrak{l}_i} \triangleq \sum_{\ell_i \in \mathfrak{l}_i} w_{\ell_i}$. Choosing a coarse–scale node in each of these KD–trees then identifies some subset of the product density labels, which we denote by $\mathfrak{L} = \mathfrak{l}_1 \times \cdots \times \mathfrak{l}_d$.

The approximate sampling procedure is structurally similar to the exact sampler of Sec. 3.4.1, but uses bounds derived from the KD–trees to reduce computation. We first use a multi–tree recursion to identify sets of product density components $\mathfrak{L}$ whose elements $\mathcal{L} \in \mathfrak{L}$ have nearly constant weight $\hat{w}_{\mathcal{L}}$. Summarizing all product density components in this fashion, we may then approximate the weight partition function as $\hat{Z} = \sum \hat{w}_{\mathfrak{L}}$, where $\hat{w}_{\mathfrak{L}} = \sum_{\mathcal{L} \in \mathfrak{L}} \hat{w}_{\mathcal{L}}$. Finally, we compute the cumulative distribution of these same *sets* of labels, assigning total probability $\hat{w}_{\mathfrak{L}}/\hat{Z}$ to those components $\mathcal{L} \in \mathfrak{L}$.

Approximate Evaluation of the Weight Partition Function

We first note that the weights assigned to components of the product distribution (eq. (3.36)) can be expressed using terms which involve only pairwise distances:

$$w_{\mathcal{L}} = \Big(\prod_{i=1}^{d} w_{\ell_i}\Big) \cdot \prod_{(\ell_i, \ell_{j>i})} \mathcal{N}\big(\mu_{\ell_i}; \mu_{\ell_j}, \Lambda_{(i,j)}\big) \qquad \text{where} \qquad \Lambda_{(i,j)} = \Lambda_i \Lambda_{\mathcal{L}}^{-1} \Lambda_j \tag{3.43}$$

This equation may be divided into two parts: a weight contribution $\prod_i w_{\ell_i}$, and a distance contribution (which we denote by $K_{\mathcal{L}}$) expressed in terms of the pairwise distances between kernel centers. We use the KD-trees' bounding boxes to compute bounds on each of these pairwise distance terms for a collection of labels $\mathfrak{L} = \mathfrak{l}_1 \times \cdots \times \mathfrak{l}_d$. The product of these upper (lower) pairwise bounds is itself an upper (lower) bound on the total distance contribution for any label $\mathcal{L} \in \mathfrak{L}$. Let these bounds by denoted by $K_{\mathfrak{L}}^{+}$ and $K_{\mathfrak{L}}^{-}$, respectively. We could also use multipole methods such as the Fast Gauss Transform [275] to efficiently compute other, potentially tighter bounds on these pairwise interactions.

By using the mean $K_{\mathfrak{L}}^{*} = \frac{1}{2}\left(K_{\mathfrak{L}}^{+} + K_{\mathfrak{L}}^{-}\right)$ to approximate $K_{\mathcal{L}}$, we incur an error of at most $\frac{1}{2}\left(K_{\mathfrak{L}}^{+} - K_{\mathfrak{L}}^{-}\right)$ for any label $\mathcal{L} \in \mathfrak{L}$. Let δ be a small tolerance parameter, whose

relationship to ϵ is quantified by Thm. 3.4.1. If this error is less than $Z\delta$ (which we ensure by comparing to a running lower bound Z_{min} on Z), we treat it as constant over the set $\mathfrak{L}$ and approximate the contribution to Z by

$$\hat{w}_{\mathfrak{L}} = \sum_{\mathcal{L}\in\mathfrak{L}} \hat{w}_{\mathcal{L}} = K^*_{\mathfrak{L}} \sum_{\mathcal{L}\in\mathfrak{L}} \Big(\prod_i w_{\ell_i} \Big) = K^*_{\mathfrak{L}} \prod_i \Big(\sum_{\ell_i \in \mathfrak{l}_i} w_{\ell_i} \Big) = K^*_{\mathfrak{L}} \prod_i w_{\mathfrak{l}_i} \tag{3.44}$$

This quantity is easily calculated via cached statistics of the weight contained in each point set. If the error is larger than $Z\delta$, we need to refine at least one of the label sets. We use a simple heuristic to do this, finding the pair of trees with the largest discrepancy in their upper and lower pairwise distance bounds.

This recursion is summarized in Alg. 3.5. Note that all of the quantities required by this algorithm may be stored within the KD–trees, avoiding searches over the label sets $\mathfrak{l}_i$. At the algorithm's termination, the total error is bounded by

$$|Z - \hat{Z}| \leq \sum_{\mathcal{L}} |w_{\mathcal{L}} - \hat{w}_{\mathcal{L}}| \leq \sum_{\mathcal{L}} \frac{1}{2} \left(K^+_{\mathfrak{L}} - K^-_{\mathfrak{L}} \right) \prod_i w_{\ell_i} \leq Z\delta \sum_{\mathcal{L}} \prod_i w_{\ell_i} \leq Z\delta \tag{3.45}$$

where the last inequality follows because each input mixture's weights are normalized. This guarantees that our estimate $\hat{Z}$ is within a fractional tolerance δ of its true value.

Approximate Sampling from the Cumulative Distribution

To use the partition function estimate $\hat{Z}$ for approximate sampling, we repeat the multi–tree recursion in a manner analogous to the exact sampler. In particular, we first draw L sorted uniform random variables, and then locate these samples in the cumulative distribution. We do not explicitly construct the cumulative distribution, but instead use the same approximate partial weight sums used to determine $\hat{Z}$ (see eq. (3.44)) to find the block of labels $\mathfrak{L} = \mathfrak{l}_1 \times \cdots \times \mathfrak{l}_d$ associated with each sample. Since all labels $\mathcal{L} \in \mathfrak{L}$ within this block have nearly equal distance contribution $K_{\mathcal{L}} \approx K^*_{\mathfrak{L}}$, we independently sample labels $\ell_i \in \mathfrak{l}_i$ for each input mixture according to their weights w_{ℓ_i}.

This multi–tree sampling algorithm is summarized in Alg. 3.6. Note that, to be consistent about when approximations are made and thus produce weights $\hat{w}_{\mathfrak{L}}$ which still sum to $\hat{Z}$, we repeat the procedure for computing $\hat{Z}$ exactly, including recomputing the running lower bound Z_{min}. The accuracy of this sampling algorithm is then characterized by the following result.

Theorem 3.4.1. *Consider a sample from a product of Gaussian mixtures drawn via Alg. 3.6, using a partition function estimate $\hat{Z}$ determined from Alg. 3.5 with tolerance parameter δ. This sample is then guaranteed to be drawn from product component $\mathcal{L}$ with probability $\hat{p}_{\mathcal{L}} \in [p_{\mathcal{L}} - \epsilon, p_{\mathcal{L}} + \epsilon]$, where*

$$|\hat{p}_{\mathcal{L}} - p_{\mathcal{L}}| = \left| \frac{\hat{w}_{\mathcal{L}}}{\hat{Z}} - \frac{w_{\mathcal{L}}}{Z} \right| \leq \frac{2\delta}{1-\delta} \triangleq \epsilon \tag{3.46}$$

MultiTree($\mathfrak{l}_1, \ldots, \mathfrak{l}_d$)

1. For each pair of distributions $(i, j > i)$, use their bounding boxes to compute
$$K_{\max}^{(i,j)} \geq \max_{\ell_i \in \mathfrak{l}_i, \ell_j \in \mathfrak{l}_j} \mathcal{N}\big(\mu_{\ell_i}; \mu_{\ell_j}, \Lambda_{(i,j)}\big)$$
$$K_{\min}^{(i,j)} \leq \min_{\ell_i \in \mathfrak{l}_i, \ell_j \in \mathfrak{l}_j} \mathcal{N}\big(\mu_{\ell_i}; \mu_{\ell_j}, \Lambda_{(i,j)}\big)$$
2. Determine overall upper and lower bounds on the pairwise distance weights:
$$K_{\max} = \prod_{(i,j>i)} K_{\max}^{(i,j)} \qquad K_{\min} = \prod_{(i,j>i)} K_{\min}^{(i,j)}$$
3. If $\frac{1}{2}(K_{\max} - K_{\min}) \leq Z_{\min}\delta$, approximate this combination of label sets as follows:
   (a) $\hat{w}_{\mathfrak{L}} = \frac{1}{2}(K_{\max} + K_{\min})\left(\prod w_{\mathfrak{l}_i}\right)$ where $w_{\mathfrak{l}_i} = \sum_{\ell_i \in \mathfrak{l}_i} w_{\ell_i}$ is cached by the KD-trees.
   (b) $Z_{\min} = Z_{\min} + K_{\min}\left(\prod w_{\mathfrak{l}_i}\right)$
   (c) $\hat{Z} = \hat{Z} + \hat{w}_{\mathfrak{L}}$
4. Otherwise, refine one of the given label sets:
   (a) Find $\arg\max_{(i,j)} K_{\max}^{(i,j)}/K_{\min}^{(i,j)}$ such that $\text{range}(\mathfrak{l}_i) \geq \text{range}(\mathfrak{l}_j)$.
   (b) Recursively call the following methods:
   MultiTree($\mathfrak{l}_1, \ldots, \text{Nearer}(\text{Left}(\mathfrak{l}_i), \text{Right}(\mathfrak{l}_i), \mathfrak{l}_j), \ldots, \mathfrak{l}_d$)
   MultiTree($\mathfrak{l}_1, \ldots, \text{Farther}(\text{Left}(\mathfrak{l}_i), \text{Right}(\mathfrak{l}_i), \mathfrak{l}_j), \ldots, \mathfrak{l}_d$)

   Here, Nearer(Farther) returns the nearer (farther) of the first two arguments to the third.

Algorithm 3.5. Recursive multi-tree algorithm for approximately evaluating the partition function Z for a product of d Gaussian mixtures represented by KD–trees. $Z_{\min}$ denotes a running lower bound on the partition function, while $\hat{Z}$ is the current estimate. Initialize by setting $Z_{\min} = \hat{Z} = 0$.

Proof. From the bound of eq. (3.45) on the error associated with $K_{\mathfrak{L}}^*$, it follows that

$$\frac{\hat{w}_{\mathcal{L}}}{Z} - \frac{\hat{w}_{\mathcal{L}}}{\hat{Z}} = \frac{\hat{w}_{\mathcal{L}}}{Z}\left(1 - \frac{1}{\hat{Z}/Z}\right) \leq \frac{\hat{w}_{\mathcal{L}}}{Z}\left(1 - \frac{1}{1+\delta}\right) \leq \frac{\hat{w}_{\mathcal{L}}}{Z}\left(\frac{\delta}{1+\delta}\right) \leq \delta$$

where the last inequality uses $\hat{w}_{\mathcal{L}} \leq \hat{Z} \leq Z(1+\delta)$. A comparable lower bound can then be determined as follows:

$$\frac{\hat{w}_{\mathcal{L}}}{Z} - \frac{\hat{w}_{\mathcal{L}}}{\hat{Z}} \geq \frac{\hat{w}_{\mathcal{L}}}{Z}\left(1 - \frac{1}{1-\delta}\right) \geq \frac{\hat{w}_{\mathcal{L}}}{Z}\left(\frac{-\delta}{1-\delta}\right) \geq -\frac{1+\delta}{1-\delta}\delta$$

Combining these expressions, it then follows that

$$\left|\frac{\hat{w}_{\mathcal{L}}}{Z} - \frac{\hat{w}_{\mathcal{L}}}{\hat{Z}}\right| \leq \frac{1+\delta}{1-\delta}\delta$$

Thus, the estimated probability of choosing label $\mathcal{L}$ has at most error

$$\left|\frac{w_{\mathcal{L}}}{Z} - \frac{\hat{w}_{\mathcal{L}}}{\hat{Z}}\right| \leq \left|\frac{w_{\mathcal{L}}}{Z} - \frac{\hat{w}_{\mathcal{L}}}{Z}\right| + \left|\frac{\hat{w}_{\mathcal{L}}}{Z} - \frac{\hat{w}_{\mathcal{L}}}{\hat{Z}}\right| \leq \frac{2\delta}{1-\delta}$$

Given the final partition function estimate $\hat{Z}$, repeat Alg. 3.5 with the following modifications:

3. (c) If $\hat{c} \leq \hat{Z} u_m < \hat{c} + \hat{w}_{\mathfrak{L}}$ for any m, draw $\mathcal{L} \in \mathfrak{L}$ by sampling $\ell_i \in \mathfrak{l}_i$ with weight $w_{\ell_i}/w_{\mathfrak{l}_i}$

3. (d) $\hat{c} = \hat{c} + \hat{w}_{\mathfrak{L}}$

Algorithm 3.6. Recursive multi-tree algorithm for approximate sampling from a product of d Gaussian mixtures. We let $\hat{c}$ denote the cumulative sum of weights $\hat{w}_{\mathfrak{L}}$. Initialize by sorting L uniform $[0, 1]$ random variables $\{u_m\}_{m=1}^{L}$, and set $Z_{\min} = \hat{c} = 0$.

which matches the definition of eq. (3.46). □

In contrast with the Gibbs samplers of Algs. 3.3 and 3.4, this epsilon–exact sampler provides explicit guarantees on the quality of its generated samples. However, its computation time is implicitly determined by the structure of the input Gaussian mixtures, and may vary widely depending on their complexity and dimensionality. The empirical examples of the following section examine these issues in more detail.

■ 3.4.8 Empirical Comparisons of Sampling Schemes

In this section, we compare the proposed sampling methods on three one–dimensional examples, each involving products of mixtures of 100 Gaussians (see Fig. 3.6). We measure performance by drawing 100 approximate samples, constructing a kernel density estimate using likelihood cross–validation [263], and calculating the KL divergence from the true product density. We repeat this test 250 times for each of a range of parameter settings of each algorithm, and plot the average KL divergence versus computation time. We focus on one–dimensional test cases so that discretization may be used to efficiently evaluate these KL divergence scores.

Exact sampling from products of Gaussian mixtures is typically very slow, due to the exponential number of components in the product density (see Sec. 3.4.1). Thus, for all examples except the two–mixture product of Fig. 3.6(c), the computation time of exact sampling is too large to be shown on our result plots. In these cases, we use a horizontal line to indicate the accuracy of exact sampling, and list the corresponding computation time in the caption. Note that, because we estimate the product density from 100 samples, even exact sampling has a non–zero KL divergence score.

For the product of three mixtures in Fig. 3.6(a), the multiscale (MS) Gibbs samplers dramatically outperform standard Gibbs sampling. In addition, we see that sequential Gibbs sampling is more accurate than parallel. Both of these differences can be attributed to the bimodal product density. However, the most effective algorithm is the ϵ–exact sampler, which matches exact sampling's performance in far less time (0.05 versus 2.75 seconds). For a product of five densities (Fig. 3.6(b)), the cost of exact sampling increases to 7.6 hours, but the ϵ–exact sampler matches its performance in less than one minute. Even faster, however, is the sequential MS Gibbs sampler, which takes only 0.3 seconds.

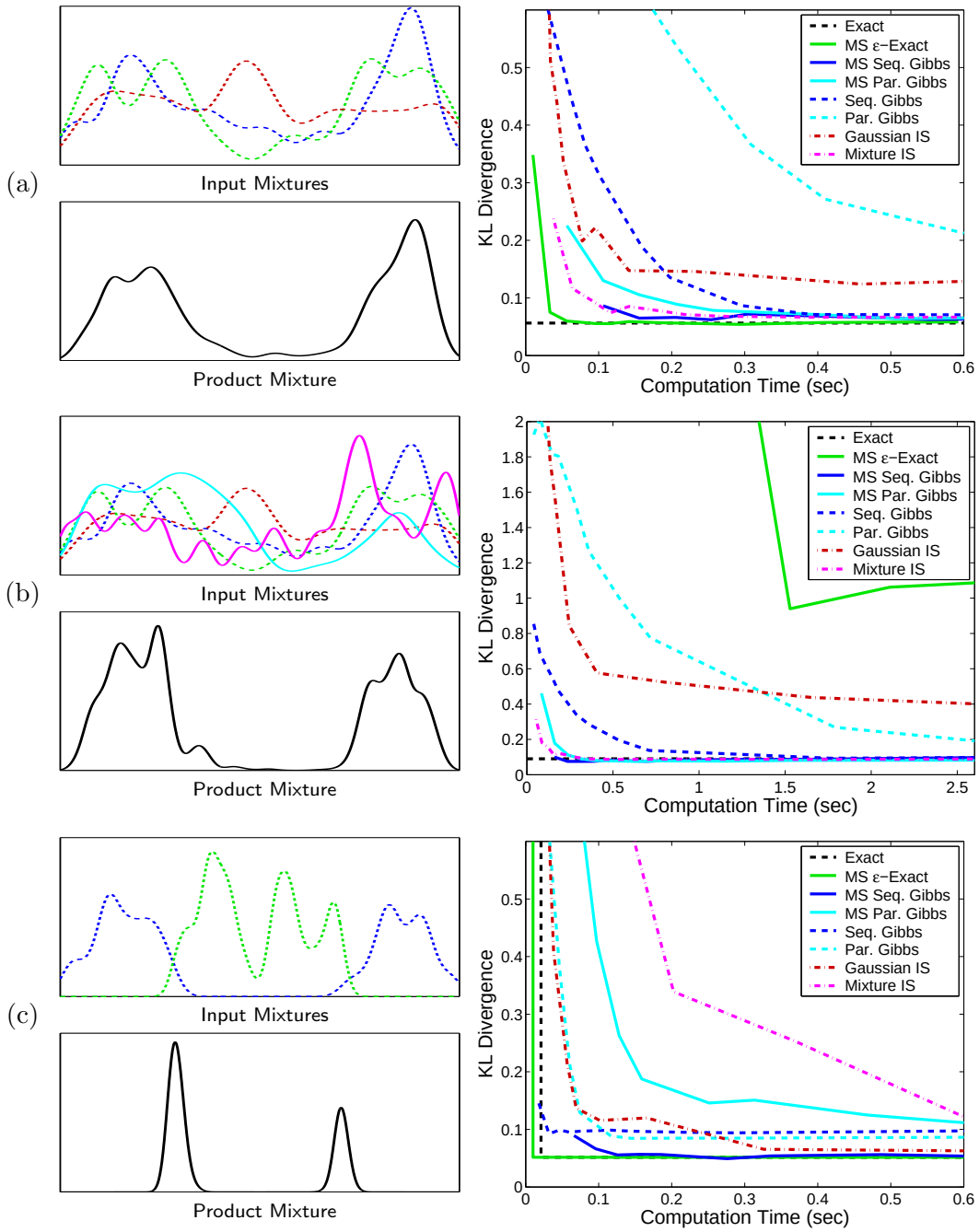


Figure 3.6. Comparison of average sampling accuracy versus computation time for different algorithms (see text). (a) Product of 3 mixtures (exact requires 2.75 sec). (b) Product of 5 mixtures (exact requires 7.6 hours). (c) Product of 2 mixtures (exact requires 0.02 sec). All computation times measured on an 800 MHz Intel Pentium III workstation.

For the previous two examples, mixture importance sampling (IS) is nearly as accurate as the best multiscale methods (Gaussian IS seems ineffective). However, in cases where all of the input densities have little overlap with the product density, mixture IS performs very poorly (see Fig. 3.6(c)). In contrast, multiscale samplers perform very well in such situations, because they can discard large numbers of low weight product density kernels.

3.5 Applications of Nonparametric BP

This section considers a pair of simple applications which explore the accuracy and effectiveness of the NBP algorithm. In all cases, we sample from Gaussian mixture products via the multiscale, sequential Gibbs sampler of Sec. 3.4.6, and update NBP messages as in Alg. 3.1. Chap. 4 explores the belief sampling updates of Alg. 3.2 in the context of a more challenging visual tracking application.

3.5.1 Gaussian Markov Random Fields

Gaussian graphical models provide one of the only families of continuous distributions for which the BP algorithm may be implemented exactly [276, 321]. For this reason, Gaussian models may be used to test the accuracy of the nonparametric approximations made by NBP. Note that we cannot hope for NBP to outperform algorithms, like Gaussian BP, designed to take advantage of the linear structure underlying Gaussian problems. Instead, our goal is to verify NBP's accuracy in a situation where exact comparisons are possible.

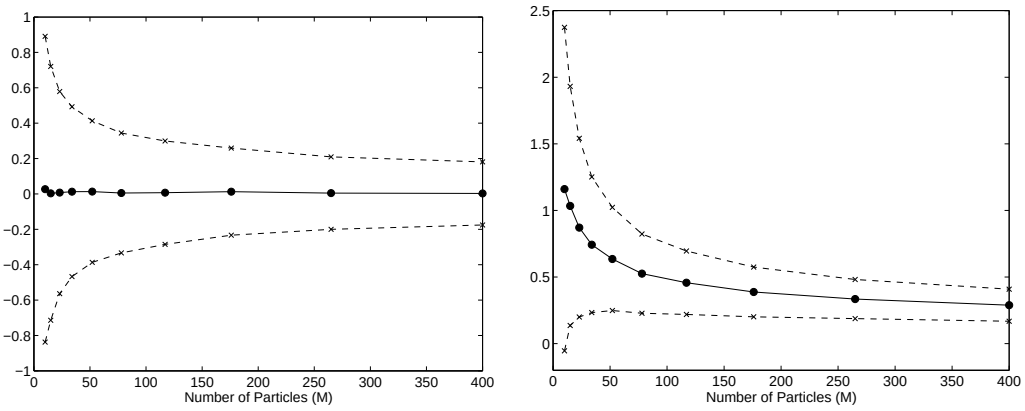
We consider NBP's performance on a 5×5 nearest–neighbor grid, as in Fig. 2.13(a), with randomly chosen inhomogeneous potentials. We note, however, that qualitatively similar results have also been observed on tree–structured graphs. Exploiting the tractable form of Gaussian potentials, we could use a simplified, analytic message propagation step [145]. To allow a more general comparison, however, we instead define our test model using Gaussian mixture potentials, which are in turn kernel density estimates based on samples drawn from true, correlated Gaussian potentials.

For each node $i \in \mathcal{V}$, Gaussian BP converges to a steady–state estimate of the marginal mean μ_i and variance σ_i^2 after about 15 iterations. To evaluate NBP, we performed 15 iterations of the NBP message updates (see Alg. 3.1) using several different particle set sizes $L \in [10, 400]$. We then found the marginal mean $\hat{\mu}_i$ and variance $\hat{\sigma}_i^2$ estimates implied by the final NBP density estimates. For each tested particle set size, the NBP comparison was repeated 100 times.

Combining the data from these NBP trials, we computed the error in the mean and variance estimates, normalized so each node behaved like a unit–variance Gaussian:

$$\tilde{\mu}_i = \frac{\hat{\mu}_i - \mu_i}{\sigma_i} \qquad \tilde{\sigma}_i^2 = \frac{\hat{\sigma}_i^2 - \sigma_i^2}{\sqrt{2}\sigma_i^2} \tag{3.47}$$

Figure 3.7 shows the mean and variance of these error statistics, across all nodes and



Mean ($\tilde{\mu}_i$) Variance ($\tilde{\sigma}_i^2$)

Figure 3.7. NBP performance on a 5×5 nearest–neighbor grid with Gaussian potentials. Plots show the mean (solid) and standard deviation (dashed) of the normalized error measures of eq. (3.47), following 15 iterations of NBP using different particle set sizes L.

trials, for different particle set sizes L. The NBP algorithm always provides unbiased mean estimates, but overly large variances. This bias, which decreases as more particles are used, is due to the smoothing inherent in kernel–based density estimates. As expected for samples drawn from Gaussian distributions, the standard deviation of both error measures falls as $L^{-1/2}$.

■ 3.5.2 Part–Based Facial Appearance Models

In this section, we use NBP to infer relationships among the PCA coefficients defining a part–based model of the human face. Several related facial appearance models have demonstrated robustness to partial occlusions [80, 215, 318]. We propose a richer model which captures statistical dependencies induced by facial expressions, and thus allows NBP to infer the *appearance* of occluded parts.

Model Construction

We begin by using training data to construct a nonparametric graphical prior for the location and appearance of five different facial features (see Fig. 3.9). A 10–dimensional linear basis for each feature's appearance is first determined via principal components analysis (PCA) [215]. The hidden variable x_i at each of the graphical model's five nodes is then defined to be a 12–dimensional (10 PCA coefficients plus image position) representation of the corresponding feature. We assume that the face's orientation and scale are known, although the model could be easily extended to describe other alignment parameters [61].

Our appearance model is based on training images from the AR face database [198]. For each of 94 people, we chose four poses showing a range of expressions and lighting conditions (see Fig. 3.8). Five manually selected feature points (eyes, nose and

Figure 3.8. Two of the 94 training subjects from the AR face database [198]. Each was photographed in these four poses, which show neutral and smiling expressions and three lighting conditions.

mouth corners) were used to align each image and position the feature masks shown in Fig. 3.9(a). Subtracting the mean features shown in Fig. 3.9(b), a PCA basis for each feature's appearance may then be determined via a singular value decomposition [215].

Using the PCA representation of each training subject, we determined a kernel–based nonparametric density estimate of the joint probability of those pairs of facial features which are adjacent in the graph of Figure 3.9(c). Figure 3.10 shows several marginalizations of these 20–dimensional densities, each relating a pair of PCA coefficients (e.g., the first nose and second left mouth coefficients). Clearly, simple Gaussian approximations would obscure most of this dataset's structure. Finally, we approximate the true pairwise potentials relating neighboring PCA coefficients by corresponding pairwise kernel density estimates [95]. Differences in the positions of neighboring features are then modeled by a two–dimensional Gaussian distribution, with mean and covariance estimated from the training set.

Estimation of Occluded Features

In this section, we apply the graphical model of Fig. 3.9 to the simultaneous location and reconstruction of partially occluded faces. Given an input image, we first identify the region most likely to contain a face using a standard eigenface detector [215] trained on partial face images. This step helps to prevent spurious detection of background detail by the individual components. Chap. 5 describes other representations of visual features which are more discriminant than linear PCA bases. For each part, we scan this region with the feature mask, producing the best PCA reconstruction $\hat{v}$ of each pixel window v. The observation potential is created by defining a Gaussian mixture component, with mean $\hat{v}$ and weight $\exp\{-||v - \hat{v}||^2/2\sigma^2\}$, for each v. To allow for outliers due to occlusion, the observation potential is augmented by a zero mean, high–variance Gaussian weighted to represent 20% of the total likelihood.

We tested the NBP algorithm on images of individuals not found in the training set.

(a) (b) (c)

Figure 3.9. Part–based model of the position and appearance of five facial features. (a) Masks defining the pixel regions corresponding to each feature. Note that the two mouth masks overlap. (b) Mean intensities of each feature, used to construct a PCA–based appearance model. (c) Graphical prior relating the position and PCA coefficients of different features.

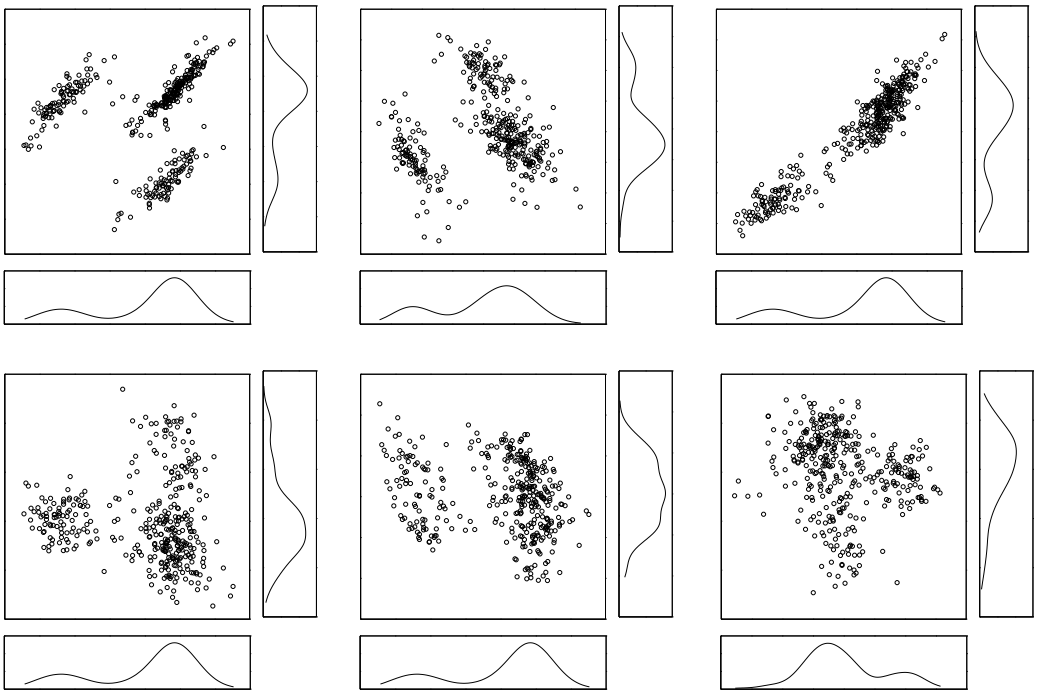


Figure 3.10. Empirical joint distributions of six different pairs of PCA coefficients. Each plot shows the corresponding marginal distributions along the bottom and right edges. Note the multimodal, non–Gaussian relationships underlying these facial features.

Each message was represented by $L = 100$ particles. Due to the high dimensionality of the variables in this model, and the presence of the occlusion process, discretization is intractable. We instead compare NBP's position and appearance estimates to the closed form solution obtained by fitting a single Gaussian to each of the nonparametric prior and observation potentials.

Figure 3.11 shows inference results for two images of a man concealing his mouth. In one image he is smiling, while in the other he is not. Using the relationships between eye and mouth shape learned from the training set, NBP correctly infers the concealed mouth's expression. In contrast, the Gaussian approximation distorts the relationships shown in Fig. 3.10, and produces results which are indistinguishable from the mean mouth shape. Note that these results were obtained in an unsupervised fashion, without any manual labeling of the training image expressions.

Figure 3.12 shows inference results for two images of a woman concealing one eye. In one image, she is seen under normal illumination, while in the second she is illuminated from the left by a bright light. In both cases, the structure of the concealed eye mirrors the visible eye. In addition, NBP correctly modifies the illumination of the occluded eye to match the intensity of the corresponding mouth corner. This example shows NBP's ability to integrate information from multiple nodes, producing globally consistent estimates.

■ 3.6 Discussion

This chapter developed a nonparametric, sampling–based extension of the belief propagation algorithm for graphical models containing continuous, non–Gaussian random variables. Via importance sampling methods, we have shown how NBP may be flexibly adapted to a broad range of analytic potential functions. Multiscale sampling methods then improve the accuracy and efficiency of message updates. Motivated by NBP's success in using part–based models to estimate facial appearance, Chap. 4 considers a more challenging application to the visual tracking of articulated hand motion. Chap. 5 then revisits object recognition problems, and develops methods for robustly learning nonparametric, part–based models.

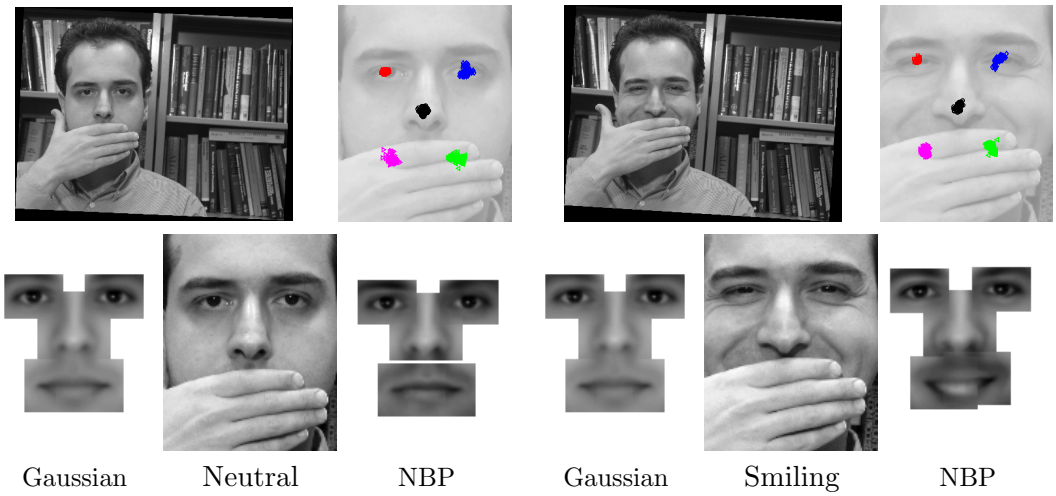


Figure 3.11. Simultaneous estimation of location (top row) and appearance (bottom row) of an occluded mouth. Results for the Gaussian approximation are on the left of each panel, and for NBP on the right. By observing the squinting eyes of the subject (right), and exploiting the feature interrelationships represented in the trained graphical model, the NBP algorithm correctly infers that the occluded mouth should be smiling. A parametric Gaussian model fails to capture these relationships.

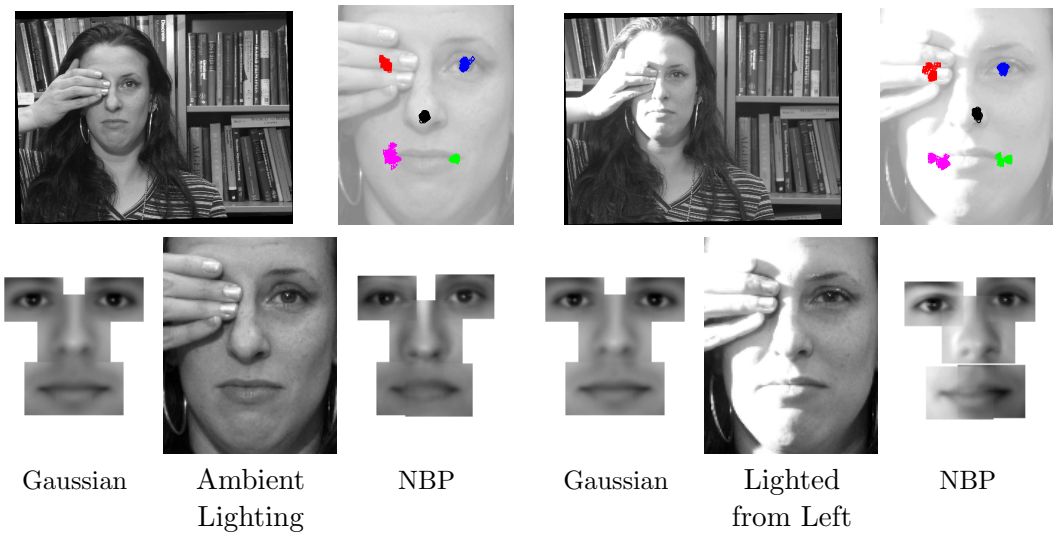


Figure 3.12. Simultaneous estimation of location (top row) and appearance (bottom row) of an occluded eye. NBP combines information from the visible eye and mouth to determine both shape and illumination of the occluded eye, correctly inferring that the left eye should brighten under the lighting conditions shown at right.

Chapter 4

Visual Hand Tracking

ACCURATE visual tracking of articulated objects is a challenging problem with applications in human–computer interfaces, motion capture, and scene understanding [103, 214]. In this chapter, we develop probabilistic methods for estimating three–dimensional hand motion from video sequences. Even coarse models of the hand's geometry have 26 continuous degrees of freedom [333], making direct search over all possible 3D poses intractable. Instead, we adapt the nonparametric belief propagation (NBP) algorithm developed in Chap. 3 to this hand tracking task.

To develop a graphical model describing the hand tracking problem, we consider a redundant *local* representation in which each hand component is described by its global 3D position and orientation. Sec. 4.1 shows that the model's kinematic constraints, including self–intersection constraints not captured by joint angle representations, take a simple form in this local representation. In Sec. 4.2, we then develop an appearance model which incorporates color and edge–based image evidence. In cases where there is no self–occlusion among the hand's fingers, this appearance model factorizes across the hand's components. As we show in Sec. 4.3, efficient distributed inference is then possible using the NBP algorithm. To consistently estimate 3D orientations, our tracker adapts quaternion representations to density estimation on the rotation group.

Realistic hand motion typically induces significant self–occlusion. To address this, we introduce a set of binary auxiliary variables specifying the occlusion state of each pixel. Sec. 4.4 then uses an analytic approximation to marginalize these occlusion masks, producing an NBP hand tracker which infers occlusion events in a distributed fashion. We conclude in Sec. 4.5 with simulations demonstrating refinement of coarse initial pose estimates, and tracking of extended motion sequences. Portions of these results were presented at the 2004 CVPR Workshop on Generative Model Based Vision [278], and the 2004 Conference on Neural Information Processing Systems [279].

■ 4.1 Geometric Hand Modeling

Structurally, the hand is composed of sixteen approximately rigid components: three phalanges or links for each finger and thumb, as well as the palm [333]. As proposed by [240, 270], we model each rigid body by one or more truncated quadrics (ellipsoids, cones, and cylinders) of fixed size. These geometric primitives are well matched to the

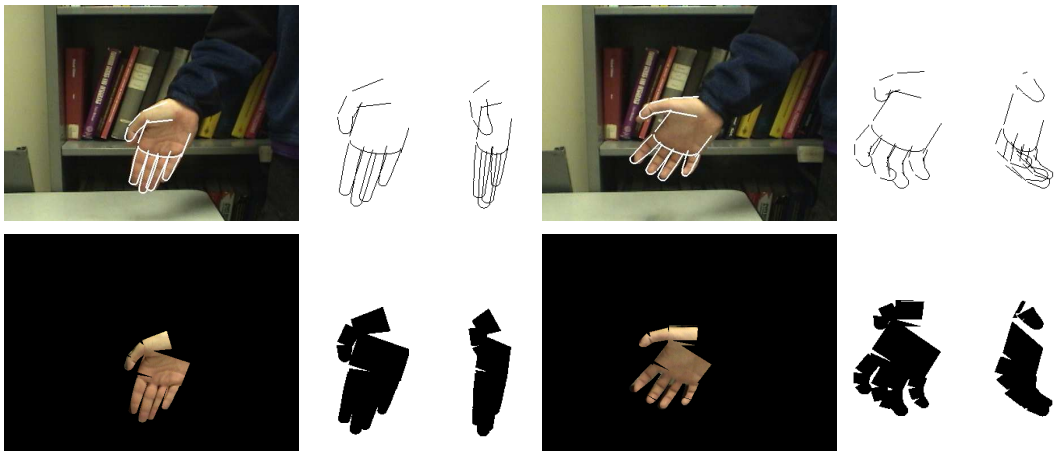


Figure 4.1. Projected edges (top row) and silhouettes (bottom row) for two configurations (left and right blocks) of the 3D structural hand model. To aid visualization, the model joint angles are set to match the images (left), and then also projected following rotations by 35° (center) and 70° (right) about the vertical axis.

true geometry of the hand, and in contrast to 2.5D "cardboard" models [332, 334], allow tracking from arbitrary orientations. Furthermore, they permit efficient computation of projected boundaries and silhouettes [32, 270].

Figure 4.1 shows the edges and silhouettes corresponding to a sample hand model configuration. Because our model is designed for estimation, not visualization, precise modeling of all parts of the hand is unnecessary. As our tracking results demonstrate, it is sufficient to capture the coarse structural features which are most relevant to the observation model described in Sec. 4.2.

■ 4.1.1 Kinematic Representation and Constraints

The kinematic constraints between different hand model components are well described by revolute joints [333]. Figure 4.2(a) shows a graph depicting this kinematic structure, in which nodes correspond to rigid bodies and edges to joints. The two joints connecting the phalanges of each finger and thumb have a single rotational degree of freedom, while the joints connecting the base of each finger to the palm have two degrees of freedom (corresponding to grasping and spreading motions). These twenty angles, combined with the palm's global position and orientation, provide 26 degrees of freedom.

To determine the image evidence for a given hand configuration, the 3D position and orientation, or pose, of each hand component must be determined. This forward kinematics problem may be solved via a series of transformations derived from the position and orientation of each joint axis, along with the corresponding joint angles (see, for example, [204] for details). While most model–based hand trackers use this joint angle parameterization, we instead explore a redundant representation in which the i^{th} rigid body is described by its position u_i and orientation r_i (a unit quaternion).

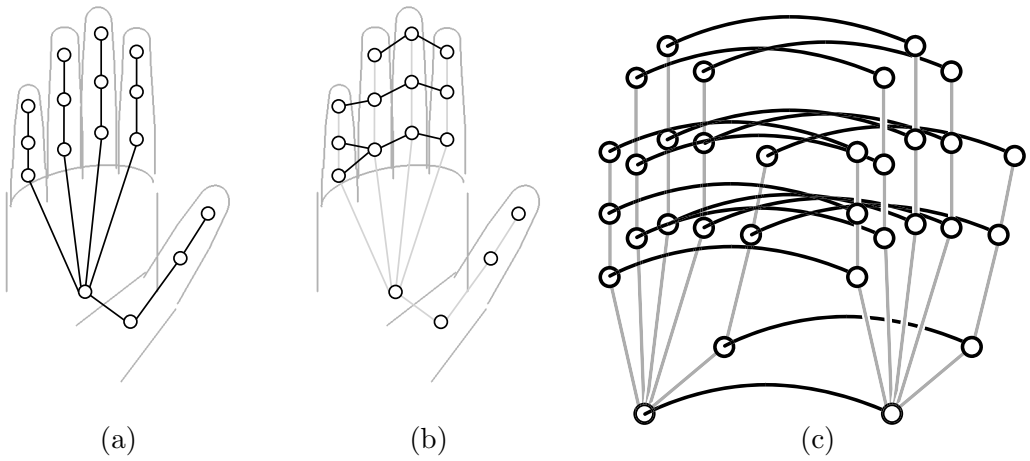


Figure 4.2. Graphs describing the hand model's constraints. (a) Kinematic constraints ($\mathcal{E}_K$) derived from revolute joints. (b) Structural constraints ($\mathcal{E}_S$) preventing 3D component intersections. (c) Dynamics relating two consecutive time steps (structural edges not shown for clarity).

Let $x_i = (u_i, r_i)$ denote this *local* description of each component, and $x = \{x_1, \ldots, x_{16}\}$ the overall hand configuration. Sec. 4.3.1 discusses quaternion representations of 3D orientation in more detail.

Clearly, there are dependencies among the elements of x implied by the kinematic constraints. Let $\mathcal{E}_K$ be the set of all pairs of rigid bodies which are connected by joints, or equivalently the edges in the kinematic graph of Fig. 4.2(a). For each joint $(i, j) \in \mathcal{E}_K$, define an indicator function $\psi^K_{i,j}(x_i, x_j)$ which is equal to one if the pair (x_i, x_j) are valid rigid body configurations associated with *some* setting of the angles of joint (i, j), and zero otherwise. Viewing the component configurations x_i as random variables, the following prior explicitly enforces all constraints implied by the original joint angle representation:

$$p_K(x) \propto \prod_{(i,j)\in\mathcal{E}_K} \psi^K_{i,j}(x_i, x_j) \tag{4.1}$$

Equation (4.1) shows that $p_K(x)$ is defined by an undirected graphical model, whose Markov structure is described by the graph representing the hand's kinematic structure (Fig. 4.2(a)). Intuitively, this graph expresses the fact that conditioned on the pose of the palm, the position and orientation of each finger are determined by an independent set of joint angles, and are thus statistically independent.

At first glance, the local representation described in this section may seem unattractive: the state dimension has increased from 26 to 96, and inference algorithms must now explicitly deal with the prior constraints encoded by $p_K(x)$. However, as we show in the following sections, local encoding of the model state greatly simplifies many other aspects of the tracking problem.

4.1.2 Structural Constraints

In reality, the hand's joint angles are coupled because different fingers can never occupy the same physical volume. This constraint is complex in a joint angle parameterization, but simple in our local representation: the position and orientation of every pair of rigid bodies must be such that their component quadric surfaces do not intersect.

We approximate this ideal constraint in two ways. First, we only explicitly constrain those pairs of rigid bodies which are most likely to intersect, corresponding to the edges $\mathcal{E}_S$ of the graph in Fig. 4.2(b). Furthermore, we note that the kinematic prior $p_K(x)$ implicitly constrains the quadrics composing each finger to have similar relative orientations. We may thus detect most intersections based on the distance between object centroids u_i, so that the structural prior becomes

$$p_S(x) \propto \prod_{(i,j)\in\mathcal{E}_S} \psi^S_{i,j}(x_i, x_j) \qquad \psi^S_{i,j}(x_i, x_j) = \begin{cases} 1 & ||u_i - u_j|| > \varepsilon_{i,j} \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

Here, $\varepsilon_{i,j}$ is determined from the fixed dimensions of the quadrics composing rigid bodies i and j. Empirically, we find that this constraint helps prevent different fingers from tracking the same image data.

4.1.3 Temporal Dynamics

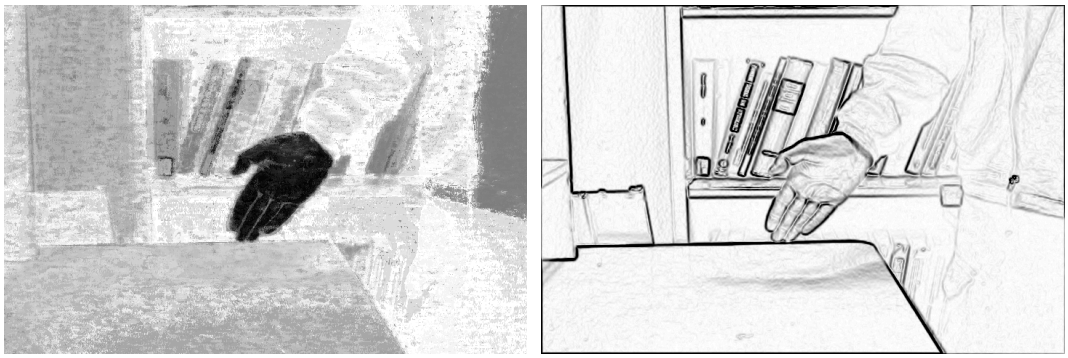
In order to exploit the temporal information encoded in video sequences, we construct a simple model of the hand's dynamics. Let x_i^t denote the position and orientation of the i^{th} hand component at time t, and $x^t = \{x_1^t, \ldots, x_{16}^t\}$. For each component at time t, our dynamical model adds a Gaussian potential connecting it to the corresponding component at the previous time step (see Fig. 4.2(c)):

$$p_T\left(x^t \mid x^{t-1}\right) = \prod_{i=1}^{16} \mathcal{N}\left(x_i^t; x_i^{t-1}, \Lambda_i\right) \tag{4.3}$$

Although this temporal model is factorized, the kinematic constraints at the following time step implicitly couple the corresponding random walks. Via results in Sec. 2.1.1, these dynamics can be justified as the maximum entropy model given observations of the marginal variance Λ_i of each hand component.

4.2 Observation Model

Our hand tracking system is based on a set of efficiently computed edge and color cues. Previous work has demonstrated the effectiveness of similar features in visual tracking applications [260, 333]. For notational simplicity, we focus on a single video frame for the remainder of this section, and denote the hand's pose by $x = \{x_1, \ldots, x_{16}\}$. We then let v represent the color and intensity of an individual pixel, and $\mathbf{v} = \{v \mid v \in \Upsilon\}$ the full image defined by some rectangular pixel lattice Υ.



(a) (b)

Figure 4.3. Image evidence used for visual hand tracking. (a) Likelihood ratios for a color–based skin model (dark pixels are likely to be skin). (b) Likelihood ratios for derivative of Gaussian filter response magnitudes (dark pixels are likely hand boundaries).

■ 4.2.1 Skin Color Histograms

Skin colored pixels have predictable statistics, which we model using a histogram distribution p_{skin} estimated from manually labeled training patches. Following [157], the red, green, and blue color channels were each discretized to 32 levels. A small positive constant, corresponding to a Dirichlet prior biased towards sparse appearance distributions (see Sec. 2.1.3), was added to each bin total to avoid overfitting. Images not depicting hands or skin were then used to create a comparably binned background histogram model p_{bkgd}. Empirically, we find that these histograms better capture saturation and lighting effects than Gaussian color models [278].

As in Fig. 4.1, let $\Omega(x)$ denote the set of pixels in the projected silhouette of 3D hand pose x. For simplicity, we assume that the colors associated with different pixels $v \in \Upsilon$ are independent given x, so that an image $\mathbf{v}$ has color likelihood

$$p_C(\mathbf{v} \mid x) = \prod_{v \in \Omega(x)} p_{\text{skin}}(v) \prod_{v \in \Upsilon \setminus \Omega(x)} p_{\text{bkgd}}(v) \propto \prod_{v \in \Omega(x)} \frac{p_{\text{skin}}(v)}{p_{\text{bkgd}}(v)} \tag{4.4}$$

The final expression neglects the proportionality constant $\prod_{v \in \Upsilon} p_{\text{bkgd}}(v)$, which is independent of x, and thereby limits computation to the silhouette region [48, 260]. To make likelihood evaluation efficient, we precompute the cumulative sum of log likelihood ratios along each row of pixels. Analogously to the integral images widely used for object detection [304], we may then quickly determine the likelihood of each hypothesized silhouette given only its boundary. Figure 4.3(a) shows the value of this likelihood ratio for the pixels in a typical test image.

The silhouette $\Omega(x)$ of the overall hand is formed from the union of the silhouettes of individual hand components $\Omega(x_i)$. For 3D poses x in which there is no self–occlusion,

these component silhouettes are disjoint, and the overall color likelihood factorizes as

$$p_C(\mathbf{v} \mid x) \propto \prod_{i=1}^{16} \prod_{v \in \Omega(x_i)} \frac{p_{\text{skin}}(v)}{p_{\text{bkgd}}(v)} = \prod_{i=1}^{16} p_C(\mathbf{v} \mid x_i) \tag{4.5}$$

Note that this decomposition, which provides independent evidence for each hand component, is *not* possible in the original joint angle representation.

To allow a similar decomposition, and hence distributed inference, when there is occlusion, we augment the configuration x_i of each node with a set of binary hidden variables $z_i = \{z_{i(v)} \mid v \in \Upsilon\}$. Letting $z_{i(v)} = 0$ if pixel v in the projection of rigid body i is occluded by *any* other body, and 1 otherwise, the color likelihood may be written as

$$p_C(\mathbf{v} \mid x, z) \propto \prod_{i=1}^{16} \prod_{v \in \Omega(x_i)} \left(\frac{p_{\text{skin}}(v)}{p_{\text{bkgd}}(v)} \right)^{z_{i(v)}} = \prod_{i=1}^{16} p_C(\mathbf{v} \mid x_i, z_i) \tag{4.6}$$

Assuming they are set consistently with the hand configuration x, the hidden occlusion variables z ensure that the likelihood of each pixel in $\Omega(x)$ is counted exactly once.

■ 4.2.2 Derivative Filter Histograms

As a hand is moved in front of a camera, it occludes the background scene, often producing intensity gradients along its boundary (see Fig. 4.3(b)). In an earlier version of our tracker [278], we used the Chamfer distance [271] to match candidate hand poses to the output of an edge detector. However, we have found it to be more efficient and effective to match hand boundaries using histograms of edge filter responses [173, 260].

Using boundaries labeled in training images, we estimated a histogram p_{on} of the response of a derivative of Gaussian filter steered to the edge's orientation [93, 260]. A similar histogram p_{off} was estimated for filter outputs at randomly chosen image locations. Let $\Pi(x)$ denote the oriented edges in the projection of 3D model configuration x. Then, again neglecting dependencies among pixels, image $\mathbf{v}$ has edge likelihood

$$p_E(\mathbf{v} \mid x, z) \propto \prod_{v \in \Pi(x)} \frac{p_{\text{on}}(v)}{p_{\text{off}}(v)} = \prod_{i=1}^{16} \prod_{v \in \Pi(x_i)} \left(\frac{p_{\text{on}}(v)}{p_{\text{off}}(v)} \right)^{z_{i(v)}} = \prod_{i=1}^{16} p_E(\mathbf{v} \mid x_i, z_i) \tag{4.7}$$

Slightly abusing notation, we do not explicitly denote the dependence of p_{on} and p_{off} on derivative filter responses, rather than raw pixel intensities. As with the color model of eq. (4.6), our use of occlusion masks z leads to a local likelihood decomposition.

■ 4.2.3 Occlusion Consistency Constraints

For the occlusion–sensitive color and edge likelihood decompositions of eqs. (4.6, 4.7) to be valid, the occlusion masks z must be chosen consistently with the 3D hand pose x. These consistency constraints can be expressed by the following potential function:

$$\eta(x_j, z_{i(v)}; x_i) = \begin{cases} 0 & \text{if } x_j \text{ occludes } x_i,\ v \in \Omega(x_j),\ \text{and } z_{i(v)} = 1 \\ 1 & \text{otherwise} \end{cases} \tag{4.8}$$

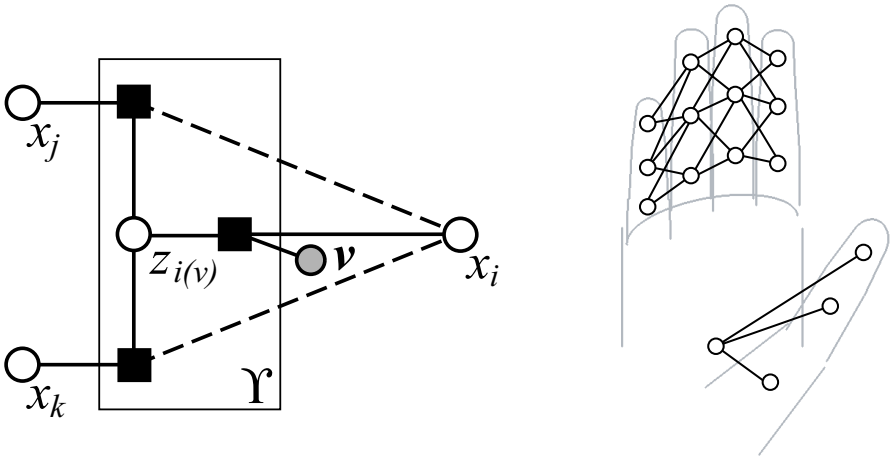


Figure 4.4. Constraints allowing distributed occlusion reasoning. *Left:* Factor graph showing the local appearance likelihoods $p_C(\mathbf{v} \mid x_i, z_i) \cdot p_E(\mathbf{v} \mid x_i, z_i)$ for rigid body i, and the occlusion constraints placed on x_i by two other hand components, x_j and x_k. Dashed lines denote weak dependencies. Variables and potentials within the plate are replicated once per pixel. *Right:* Pairs of hand components $\mathcal{E}_O$ for which occlusion relationships are explicitly considered.

Note that because the quadric surfaces defining our hand model are convex and non-intersecting, no two rigid bodies can ever take mutually occluding configurations. The constraint $\eta(x_j, z_{i(v)}; x_i)$ is zero precisely when pixel v in the projection of x_i should be occluded by x_j, but $z_{i(v)}$ is in the unoccluded state.

The following potential encodes all of the occlusion relationships between rigid bodies i and j:

$$\psi^O_{i,j}(x_i, z_i, x_j, z_j) = \prod_{v \in \Upsilon} \eta(x_j, z_{i(v)}; x_i)\, \eta(x_i, z_{j(v)}; x_j) \tag{4.9}$$

These occlusion constraints exist between all pairs of nodes. However, as with the structural prior, we enforce only those pairs $\mathcal{E}_O$ (see Fig. 4.4) most prone to occlusion:

$$p_O(x, z) \propto \prod_{(i,j) \in \mathcal{E}_O} \psi^O_{i,j}(x_i, z_i, x_j, z_j) \tag{4.10}$$

Figure 4.4 shows a factor graph for the occlusion relationships between x_i and its neighbors, as well as the observation potential $p_C(\mathbf{v} \mid x_i, z_i) \cdot p_E(\mathbf{v} \mid x_i, z_i)$. Note that the occlusion potential $\eta(x_j, z_{i(v)}; x_i)$ depends only on whether x_i is behind x_j relative to the camera, not on the precise 3D pose of x_i. We exploit this weak dependence in our algorithm for distributed occlusion reasoning.

■ 4.3 Graphical Models for Hand Tracking

In the previous sections, we have shown that a redundant, local representation of the geometric hand model's configuration x^t allows $p(x^t \mid \mathbf{v}^t)$, the posterior distribution of

the hand model at time t given observed image $\mathbf{v}^t$, to be expressed as

$$p\big(x^t \mid \mathbf{v}^t\big) \propto p_K\big(x^t\big)\, p_S\big(x^t\big)\, p_C\big(\mathbf{v}^t \mid x^t\big)\, p_E\big(\mathbf{v}^t \mid x^t\big) \tag{4.11}$$

where $p_K\big(x^t\big)$ and $p_S\big(x^t\big)$ are the kinematic and structural prior models. For simplicity, we begin by assuming that there is no self–occlusion among the fingers. In this case, as in eq. (4.5), the color and edge likelihoods factorize:

$$p\big(x^t \mid \mathbf{v}t\big) \propto p_K\big(x^t\big)\, p_S\big(x^t\big) \prod_{i=1}^{16} p_C\big(\mathbf{v}^t \mid x_i^t\big)\, p_E\big(\mathbf{v}^t \mid x_i^t\big) \tag{4.12}$$

When τ video frames are observed, the overall posterior distribution then equals

$$p(x \mid \mathbf{v}) \propto \prod_{t=1}^{\tau} p\big(x^t \mid \mathbf{v}^t\big)\, p_T\big(x^t \mid x^{t-1}\big) \tag{4.13}$$

Equation (4.13) is an example of a pairwise Markov random field, which as described in Sec. 2.2.2 takes the following general form:

$$p(x \mid \mathbf{v}) \propto \prod_{(i,j)\in\mathcal{E}} \psi_{i,j}(x_i, x_j) \prod_{i\in\mathcal{V}} \psi_i(x_i, \mathbf{v}) \tag{4.14}$$

Here, the nodes $\mathcal{V}$ correspond to the sixteen components of the hand model at each point in time, and the edges $\mathcal{E}$ arise from the union of the graphs encoding kinematic, structural, and temporal constraints. Visual hand tracking can thus be posed as inference in a graphical model.

As discussed in Sec. 2.3.2, the loopy belief propagation algorithm often provides accurate state estimates in graphs with cycles. However, for our hand tracking application, the 3D pose x_i of each rigid hand component is described by a six–dimensional continuous variable. Because accurate discretization of such spaces is intractable, and the BP message update integral has no closed form for the potentials composing our hand model, exact implementation of BP is infeasible. Instead, we employ particle–based approximations to these messages using the nonparametric belief propagation (NBP) algorithm developed in Chap. 3. The following sections adapt the NBP message and belief updates to the specific potentials arising in this visual tracking application.

■ 4.3.1 Nonparametric Estimation of Orientation

The hand tracking application is complicated by the fact that the orientation r_i of rigid body $x_i = (u_i, r_i)$ is an element of the three–dimensional rotation group $SO(3)$. Previous work on planar, affine transformations has shown that density estimates based on a manifold's true, non–Euclidean metric lead to improved accuracy and better generalization [209]. In this section, we develop kernel density estimates of orientation which respect the intrinsic geometry of $SO(3)$.

Three–Dimensional Orientation and Unit Quaternions

Unit–length quaternions provide a representation for 3D rotations and orientations with many attractive properties [259]. Like complex numbers, a quaternion $r = (a, b)$ has two components: a scalar, real part a, and a three–dimensional vector, imaginary part b. The quaternion representing a rotation θ about unit vector n is given by

$$r = (a, b) = (\cos(\theta/2),\ \sin(\theta/2)\, n) \tag{4.15}$$

Note that, because rotating by $-\theta$ about $-n$ produces the same result, the quaternions $-r$ and r represent the same 3D rotation. The unit sphere S^3 (embedded in four–dimensional Euclidean space) thus provides a double cover of the rotation group $SO(3)$, where antipodal points represent the same rotation.

Unlike Euler angles, quaternions do not suffer from the singularities which lead to the "gimbal lock" phenomenon [259]. More importantly, because S^3 and $SO(3)$ share the same metric structure, quaternions provide an appropriate space for interpolation and estimation of 3D orientations. The distance between two quaternions r_1 and r_2 is equal to the angle between them:

$$d(r_1, r_2) = \arccos(r_1 \cdot r_2) \tag{4.16}$$

In the following sections, we describe a computationally efficient framework for non-parametric estimation of orientation which respects this distance metric.

Density Estimation on the Circle

Because it is difficult to visualize S^3, we begin by assuming that two of the 3D object's rotational degrees of freedom are known. The remaining rotational direction is diffeomorphic to the unit circle, and can be represented by a single angle θ. Just as Gaussian, or normal, distributions play a central role in Euclidean space, the *folded* or *wrapped normal distribution* is natural for circular data:

$$\mathcal{N}_w\left(\theta; \mu, \sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \sum_{k=-\infty}^{\infty} \exp\left\{\frac{-(\theta - \mu + 2\pi k)^2}{2\sigma^2}\right\} \tag{4.17}$$

This density, which is intuitively constructed by wrapping a Euclidean normal distribution around the unit circle, arises from a version of the central limit theorem [194]. When $\sigma \leq \pi/3$, as is typical for kernel density estimation applications, all but one of the terms in the sum of eq. (4.17) are negligible. Using the unit vector representation of angles, it is straightforward to verify that the wrapped normal respects the distance metric of eq. (4.16).

In many applications involving circular data, it is more convenient to work with the *von Mises* distribution [194]:

$$\mathcal{M}(\theta; \mu, \kappa) = \frac{1}{2\pi I_0(\kappa)} \exp\left\{\kappa \cos(\theta - \mu)\right\} \tag{4.18}$$

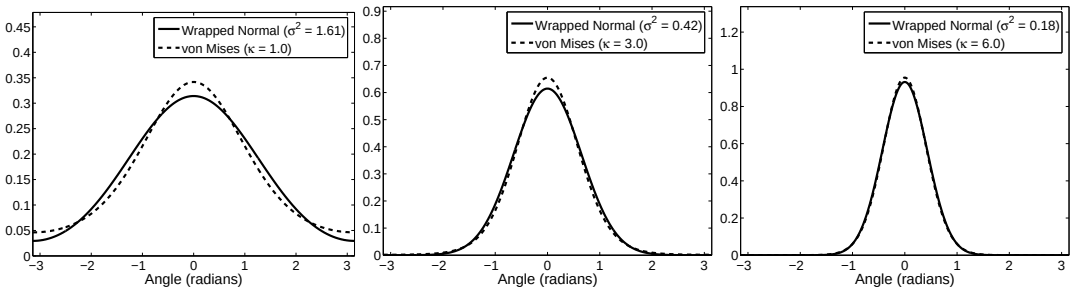


Figure 4.5. Three different wrapped normal distributions, and the corresponding von Mises distributions with moments matched according to eq. (4.19).

Here, $I_j(\kappa)$ denotes the modified Bessel function of the first kind and order j, while κ is known as the *concentration* parameter. Matching trigonometric moments, the von Mises and wrapped normal distributions are most closely aligned when the concentration parameter is set so that

$$\sigma^2 = -2 \log \frac{I_1(\kappa)}{I_0(\kappa)} \tag{4.19}$$

Figure 4.5 plots several wrapped normal distributions, and the best matching von Mises distributions. When σ^2 is small, as in applications of kernel density estimation, eq. (4.19) chooses $\kappa \approx \sigma^{-2}$, and the densities are nearly identical. Even for larger variances, the two densities are statistically indistinguishable from moderate sample sizes [44].

Interestingly, the von Mises distribution can also be derived from a bivariate Euclidean Gaussian distribution. In particular, consider a Gaussian random vector (x_1, x_2) with mean $(\cos\mu, \sin\mu)$ and covariance $\kappa^{-1} I_2$. If we transform to polar coordinates (r, θ), we find that the conditional distribution of θ given $||r|| = 1$ is von Mises [194], as in eq. (4.18).

Density Estimation on the Rotation Group

The von Mises distribution (eq. (4.18)) may be directly generalized to points r on the unit sphere S^3:

$$\mathcal{M}(r; \mu, \kappa) = \frac{\kappa}{2 I_1(\kappa)} \exp\{\kappa \mu \cdot r\} \tag{4.20}$$

Here, the mean direction μ is a unit quaternion, and κ is a scalar concentration parameter as before. This generalization, which is known as the *von Mises–Fisher* distribution [194], closely approximates a wrapped normal distribution based on the spherical metric of eq. (4.16). Furthermore, as with the circular von Mises distribution, we may obtain eq. (4.20) by conditioning an appropriate four–dimensional Euclidean Gaussian to have unit length. Note also that the von Mises–Fisher distribution defines a regular exponential family with canonical parameters $\kappa\mu$, and thus has a maximum entropy characterization as in Thm. 2.1.1.

Given these properties, the von Mises–Fisher distribution provides a natural kernel for nonparametric density estimation on the sphere:

$$\hat{p}(r) = \sum_{\ell=1}^{L} w^{(\ell)} \mathcal{M}(r; r^{(\ell)}, \kappa) \tag{4.21}$$

As in the Euclidean estimator of Sec. 2.4.2, $w^{(\ell)}$ is the weight associated with the von Mises–Fisher kernel centered on the ℓ^{th} sample $r^{(\ell)}$. Density estimates of this form have been shown to share the same strong asymptotic consistency guarantees as Euclidean kernel density estimators [13]. Furthermore, the asymptotic convergence rate attained by von Mises–Fisher kernels is the best among a wide class of candidate kernels [124], and Euclidean kernel size selection rules may be adapted to choose the concentration parameter κ.

Because we are interested in the rotation group $SO(3)$, we must account for the fact that antipodal points on S^3 represent the same rotation. One possibility would be to replace data points r_i by their negation $-r_i$ as necessary to cluster all points on a single hemisphere. However, this requires cumbersome logic, and may introduce distortions for distributions which are not tightly concentrated. Instead, as suggested by [273], we split the probability mass evenly between the two quaternion representations using a mixture of von Mises–Fisher kernels:

$$\begin{aligned} \hat{p}(r) &= \sum_{\ell=1}^{L} w^{(\ell)} \left[\frac{1}{2} \mathcal{M}(r; r^{(\ell)}, \kappa) + \frac{1}{2} \mathcal{M}(r; -r^{(\ell)}, \kappa) \right] \\ &\propto \sum_{\ell=1}^{L} w^{(\ell)} \cosh\left(\kappa r \cdot r^{(\ell)}\right) \end{aligned} \tag{4.22}$$

This construction ensures that different sources of orientation information are consistently combined, even when the underlying densities have high variance.

To compute NBP message updates, we must be able to sample from our orientation density estimates. To derive an efficient sampling rule, we represent each von Mises–Fisher kernel by a Gaussian in the ambient four–dimensional Euclidean space (see Sec. 4.3.1). It is then straightforward to sample from the *projection* of these Euclidean Gaussians onto the unit sphere: draw a sample in Euclidean space, and then divide by its length. We use importance weighting, as computed by numerical integration along the radial projection direction, to account for discrepancies between this projection and the von Mises–Fisher density obtained by *conditioning* samples to have unit length. When the concentration parameter κ is large, these weights are nearly uniform and may be neglected.

Comparison to Tangent Space Approximations

Riemannian manifolds are sometimes analyzed using the Euclidean space tangent to a specified linearization point. For S^3, the *exponential* mapping from vectors b in the

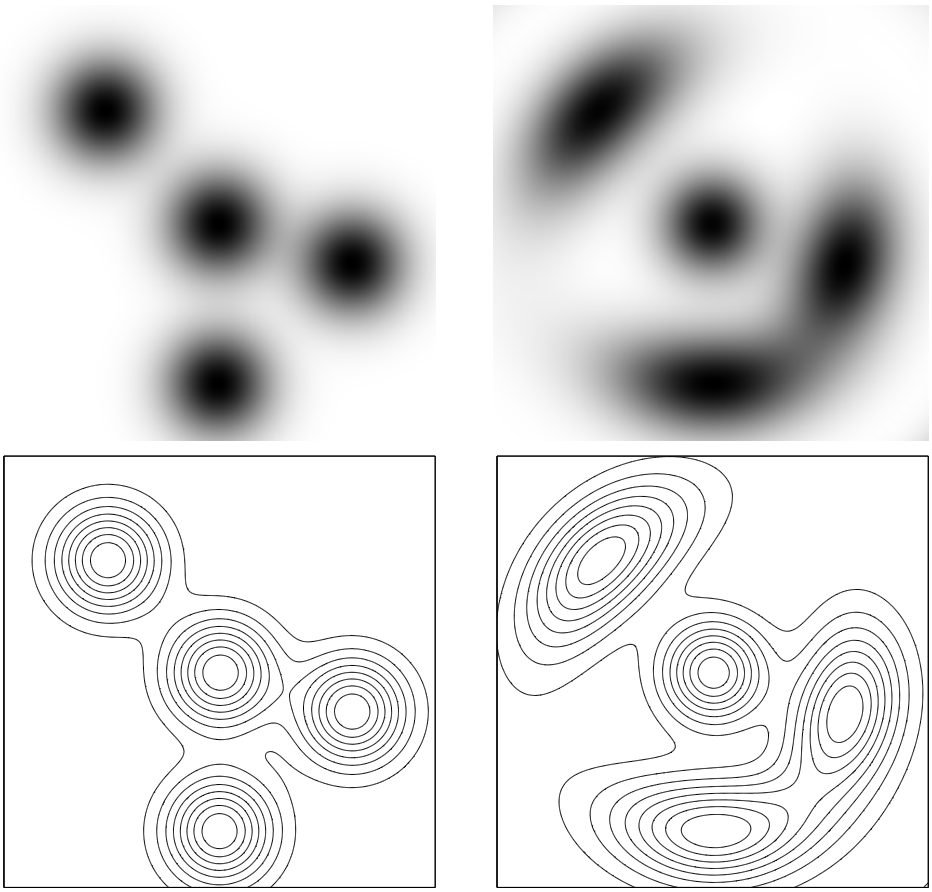


Figure 4.6. Visualization of two different kernel density estimates on S^2 using a tangent space: Gaussian kernels placed directly in the tangent space (left), and a tangent space mapping of von Mises–Fisher kernels defined on the sphere (right). The top row shows more likely points in darker gray, while the bottom row shows corresponding contour plots.

space tangent to the identity quaternion $(1, 0, 0, 0)$ to the sphere is

$$\exp(b) = \left(\cos(||b||),\ \frac{b}{||b||}\sin(||b||)\right) \tag{4.23}$$

The corresponding *logarithmic* mapping from the unit sphere, excluding the antipode $(-1, 0, 0, 0)$, to the tangent space equals

$$\log((a, b)) = \frac{b}{||b||}\arctan\left(\frac{||b||}{a}\right) \tag{4.24}$$

Approximations of this mapping have been used for automatic camera calibration [67], and to estimate limb orientation in a graphical model–based person tracker [261, 262].

While tangent space approximations are effective for some applications, they may also significantly distort the underlying Riemannian metric (eq. (4.16)). To visualize these distortions, we consider a plane tangent to S^2, the sphere obtained from S^3 by fixing one rotational degree of freedom. In Fig. 4.6, we show two kernel density estimates based on four data points. The first places Gaussian kernels directly in the space tangent to the central data point, while the second places von Mises–Fisher kernels on the sphere. Because it is difficult to plot densities on the sphere, we visualize these densities by projecting them to the same tangent plane. Although the central kernel is similar in both densities, the other von Mises–Fisher kernels are significantly distorted by the tangent mapping. Of course, on the true manifold S^2, the von Mises–Fisher kernels are symmetric, and the tangent space estimate is distorted. Among other effects, the tangent plane density estimate causes the two lower–right kernels to be improperly separated, potentially leading to generalization errors.

The distortion shown in this example is inevitable with tangent space approximations whenever the overall density is not tightly concentrated. In contrast, the proposed von Mises–Fisher kernel density estimates apply equally well to highly dispersed densities. Furthermore, we avoid having to choose a linearization point, and thus do not introduce boundary artifacts when fusing orientation estimates.

■ 4.3.2 Marginal Computation

As illustrated in Fig. 2.16, BP's belief estimate $q_i(x_i)$ of the marginal pose of rigid body i combines all incoming messages with the local observation potential. For our geometric hand model, the belief update equation is then

$$q_i(x_i) \propto p_C(\mathbf{v} \mid x_i)\, p_E(\mathbf{v} \mid x_i) \prod_{j \in \Gamma_T(i)} m_{ji}(x_i) \prod_{k \in \Gamma_K(i)} m_{ki}(x_i) \prod_{h \in \Gamma_S(i)} m_{hi}(x_i) \tag{4.25}$$

where the three products contain messages from temporal, kinematic, and structural neighbors, respectively. As we describe in Sec. 4.3.3, our NBP hand tracker employs Gaussian mixtures for some messages (along kinematic and temporal edges), and analytic functions for others (structural edges). To perform this belief update, we thus adapt the importance sampling procedure described in Sec. 3.3.2.

Alg. 4.1 summarizes the update procedure for the belief estimate $q_i(x_i)$. First, L samples $\{x_i^{(\ell)}\}_{\ell=1}^{L}$ are drawn directly from the product of the kinematic and temporal Gaussian mixture messages. As discussed in Sec. 3.4, there are a variety of computational approaches to this sampling step. Empirically, importance samplers are typically ineffective in the six–dimensional space of rigid body poses, and the epsilon–exact multiscale sampler of Alg. 3.6 requires too much computation for nodes with many neighbors. The simulations presented in this chapter thus use the sequential multiscale Gibbs sampler of Sec. 3.4.6, as it had the highest accuracy in the experiments of Sec. 3.4.8.

Using the Euclidean embedding described in Sec. 4.3.1, we may directly adapt our multiscale samplers to the von Mises–Fisher kernels used to estimate orientation. Following normalization of the rotational component, each sample $x_i^{(\ell)}$ is assigned a weight

Given input messages $m_{ji}(x_i)$ from kinematic neighbors $\Gamma_K(i)$, structural neighbors $\Gamma_S(i)$, and temporal neighbors $\Gamma_T(i)$:

1. Draw L independent samples from the product distribution
$$x_i^{(\ell)} \sim \prod_{j \in \Gamma_T(i)} m_{ji}(x_i) \prod_{k \in \Gamma_K(i)} m_{ki}(x_i)$$
using the multiscale sampling methods developed in Sec. 3.4.
2. For each $x_i^{(\ell)} = (u_i^{(\ell)}, r_i^{(\ell)})$, normalize the orientation $r_i^{(\ell)}$.
3. Compute an importance weight for each sample $x_i^{(\ell)}$:
$$w_i^{(\ell)} \propto p_C(\mathbf{v} \mid x_i^{(\ell)})\, p_E(\mathbf{v} \mid x_i^{(\ell)}) \prod_{j \in \Gamma_S(i)} m_{ji}(x_i^{(\ell)})$$
4. Use a bandwidth selection method (see [263]) to construct a kernel density estimate $q_i(x_i)$ from $\{x_i^{(\ell)}, w_i^{(\ell)}\}_{\ell=1}^L$.

Algorithm 4.1. Nonparametric BP update of the estimated 3D pose $q_i(x_i)$ for the rigid body corresponding to the i^{th} hand component.

$w_i^{(\ell)}$ equal to the product of the color and edge likelihoods with any messages along structural edges. Finally, the computationally efficient "rule of thumb" heuristic [263] is used to set the bandwidth of Gaussian smoothing kernels placed around each sample. The modes of $q_i(x_i)$ then provide good estimates of the 3D pose x_i, while samples may be used to gauge the uncertainty in these estimates.

The preceding algorithm assumes that at least one of the incoming messages is a Gaussian mixture. For the hand tracker, this is true except for the initial message updates on the first frame, when the only incoming message is the local analytic likelihood function. For the simulations presented in this chapter, we initialized the tracker by manually specifying a high variance Gaussian proposal distribution centered roughly around the true starting hand configuration. In the future, we hope to replace this manual initialization by automatic image–based feature detectors [261].

4.3.3 Message Propagation and Scheduling

To derive the message propagation rule, we consider the belief sampling form of NBP summarized by Alg. 3.2. In this method, outgoing messages from rigid body j are determined directly from the latest belief estimate at that node:

$$m_{ji}(x_i) \propto \int_{\mathcal{X}_j} \psi_{ij}(x_i, x_j) \frac{q_j(x_j)}{m_{ij}(x_j)}\, dx_j \tag{4.26}$$

This explicit use of the current marginal estimate $q_j(x_j)$ helps focus particles on the most important regions of the state space.

Consider first the case where $(i,j) \in \mathcal{E}_K$, so that ψ_{ij}^K corresponds to a kinematic constraint. The message propagation step makes direct use of the particles $\{x_j^{(\ell)}\}_{\ell=1}^L$

Given L weighted particles $\{x_j^{(\ell)}, w_j^{(\ell)}\}_{\ell=1}^L$ from $q_j(x_j)$, and the incoming message $m_{ij}(x_j)$ used to construct $q_j(x_j)$:

1. Reweight each particle as $\tilde{w}_j^{(\ell)} \propto w_j^{(\ell)} / m_{ij}(x_j^{(\ell)})$, thresholding any small weights to avoid producing an overly low effective sample size.

KINEMATIC EDGES:

2. Resample by drawing L particles $\{\tilde{x}_j^{(\ell)}\}_{\ell=1}^L$ with replacement according to $\Pr[\tilde{x}_j^{(\ell)}] \propto \tilde{w}_j^{(\ell)}$.
3. For each $\tilde{x}_j^{(\ell)}$, sample uniformly from the allowable angles for joint (i, j). Determine the corresponding $x_i^{(\ell)}$ via forward kinematics.
4. Use a bandwidth selection method to construct a kernel density estimate $m_{ji}(x_i)$ from the unweighted samples $\{x_i^{(\ell)}\}_{\ell=1}^L$.

TEMPORAL EDGES:

2. Construct a kernel density estimate $m_{ji}(x_i)$ with centers $\{x_j^{(\ell)}\}_{\ell=1}^L$, weights $\{\tilde{w}_j^{(\ell)}\}_{\ell=1}^L$, and uniform bandwidths Λ_i.

STRUCTURAL EDGES:

2. For any $x_i = (u_i, r_i)$, let $\mathcal{L} = \left\{ \ell \mid ||u_j^{(\ell)} - u_i|| > \varepsilon_{i,j} \right\}$.
3. Calculate $m_{ji}(x_i) = \sum_{\ell \in \mathcal{L}} \tilde{w}_j^{(\ell)}$.

Algorithm 4.2. Nonparametric BP update of the message $m_{ji}(x_i)$ sent from hand component j to component i as in eq. (4.26), for each of three pairwise potential types.

sampled during the last marginal belief update. To avoid overcounting information from neighboring node i, we reweight each particle $x_j^{(\ell)}$ as follows:

$$\tilde{w}_j^{(\ell)} \propto \frac{w_j^{(\ell)}}{m_{ij}\left(x_j^{(\ell)}\right)} \tag{4.27}$$

As observed in other applications [142], belief sampling message updates sometimes exhibit instability given small sample sizes L. In particular, our hand tracker's belief updates may produce samples $x_j^{(\ell)}$ which have very low probability with respect to some incoming message. When a new outgoing message is constructed, this particle is then assigned extremely *large* weight $\tilde{w}_j^{(\ell)}$ by eq. (4.27). To avoid instabilities from these outlier particles, our hand tracker heuristically thresholds all such weights to be at least 10% of the most probable particle's weight. More principled solutions to this issue are an area for future research.

Given these thresholded particle weights, we first resample to produce L unweighted particles $\{\tilde{x}_j^{(\ell)}\}_{\ell=1}^L$ (see Alg. 4.2). We must then sample candidate x_i configurations from the conditional distribution $\psi_{ij}^K(x_i, x_j = \tilde{x}_j^{(\ell)})$. Because ψ_{ij}^K is an indicator potential, this sampling has a particularly appealing form: first sample uniformly among allowable joint angles, and then use forward kinematics to find the $x_i^{(\ell)}$ corresponding to each $\tilde{x}_j^{(\ell)}$.

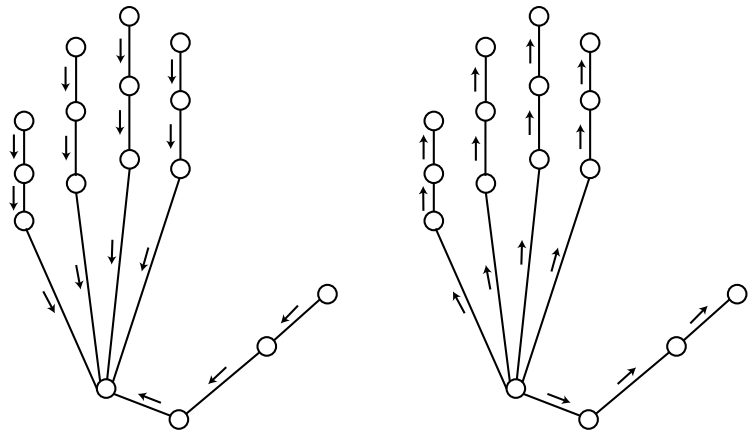


Figure 4.7. Scheduling of the kinematic constraint message updates for NBP: messages are first passed from fingertips to the palm (left), and then back to the fingertips (right). Structural constraint messages (not shown) are updated whenever the originating belief estimates change.

Finally, the "rule of thumb" bandwidth selection method [263] is used to construct the outgoing message.

Because the temporal constraint potentials are Gaussian, the sampling associated with kinematic message updates is unnecessary. Instead, as suggested by [145], we simply adjust the bandwidths of the current marginal estimate $q_j(x_j)$ to match the temporal covariance Λ_i (see Alg. 4.2). This update implicitly assumes that the bandwidth of $q_j(x_j)$ is much smaller than Λ_i, which will hold for sufficiently large L.

For structural constraint edges $\mathcal{E}_S$, a different approach is needed. In particular, from eq. (4.2) we see that the pairwise potential equals 1 for all state configurations outside some ball, and therefore the outgoing message will not be finitely integrable. For structural edges, messages must then take the form of analytic functions. In principle, at some point $x_i = (u_i, r_i)$ the message $m_{ji}(x_i)$ should equal the integral of $q_j(x_j)\,/m_{ij}(x_j)$ over all configurations outside some ball centered at u_i. We approximate this quantity by the sum of the weights of all kernels in $q_j(x_j)$ outside that ball (see Alg. 4.2).

For NBP, the message update order affects the outcome of each local Monte Carlo approximation, and may thus influence the quality of the final belief estimates. Given a single frame, we iterate the tree–based message schedule of Fig. 4.7, in which messages are passed from fingertips to the palm, and then back to the fingertips. Structural messages $m_{ji}(x_i)$, which for clarity are not shown, are also updated whenever the belief estimate $q_j(x_j)$ of the originating node changes. For video, we process the frames in sequence, updating temporal messages to the next frame following a fixed number of kinematic/structural message sweeps. This message schedule is similar to that used in the factored frontier algorithm for dynamic Bayesian networks [218]. We note, however, that the tracker could be easily extended to produce smoothed estimates, which incorporate information from future video frames, using reverse–time messages.

4.3.4 Related Work

The NBP algorithm has also recently been used to develop a multiple camera, 3D person tracker [261, 262]. However, this tracker uses a "loose–limbed" formulation of the kinematic constraints which represents the conditional distribution of each limb's location, given its neighbor, via a Gaussian mixture estimated from training data. The pair of conditional densities associated with each joint are learned independently, and do not necessarily match any global generative model. The 2D tracking results of [145, 332] are also based on explicit (and sometimes inconsistent) kinematic relaxations.

In contrast, we have shown that an NBP tracker may be built around the local structure of the true kinematic constraints. Conceptually, this has the advantage of providing a clearly specified, globally consistent, generative model whose properties can be analyzed. Practically, our formulation avoids the need to explicitly approximate the kinematic constraints, and allows us to build a functional tracker without the need for precise, labeled training data.

4.4 Distributed Occlusion Reasoning

The hand tracking algorithm developed in the previous section assumed the fingers being tracked did not occlude each other. However, most realistic hand motion sequences contain significant self–occlusion. To develop a tracker for these cases, we use the occlusion masks introduced in Sec. 4.2.1 to locally decompose the color (eq. (4.6)) and edge (eq. (4.7)) likelihoods. The posterior distribution over hand pose at time t is then

$$p\left(x^t \mid \mathbf{v}^t\right) \propto \sum_{z^t} p_K\left(x^t\right) p_S\left(x^t\right) p_O\left(x^t, z^t\right) \prod_{i=1}^{16} p_C\left(\mathbf{v}^t \mid x_i^t, z_i^t\right) p_E\left(\mathbf{v}^t \mid x_i^t, z_i^t\right) \tag{4.28}$$

where the summation marginalizes over occlusion masks z^t. Given a video sequence, the overall posterior distribution $p(x \mid \mathbf{v})$ is again given by eq. (4.13).

4.4.1 Marginal Computation

To derive BP updates for the occlusion masks z_i, we first group (x_i, z_i) for each hand component so that $p\left(x^t, z^t \mid \mathbf{v}^t\right)$ has a pairwise form (as in eq. (4.14)). In principle, NBP could manage occlusion constraints by sampling candidate occlusion masks z_i along with rigid body configurations x_i. However, due to the exponentially large number of possible occlusion masks, we employ a more efficient analytic approximation.

Consider the BP message sent from x_j to (z_i, x_i), calculated by applying eq. (4.26) to the occlusion potential $\prod_v \eta(x_j, z_{i(v)}; x_i)$. We assume that likely 3D poses in $q_j(x_j)$ are well separated from any candidate x_i, a situation typically ensured by the kinematic and structural constraints. The occlusion constraint's weak dependence on x_i (see Fig. 4.4) then separates the message computation into two cases. If x_i lies in front of typical

x_j configurations, the BP message $\bar{m}_{j,i(v)}(z_{i(v)})$ is uninformative. If x_i is occluded, the message approximately equals

$$\bar{m}_{j,i(v)}(z_{i(v)} = 0) = 1 \qquad \bar{m}_{j,i(v)}(z_{i(v)} = 1) = 1 - \Pr\left[v \in \Omega\left(x_j\right)\right] \tag{4.29}$$

where we have neglected correlations among pixel occlusion states, and where the probability is computed with respect to $q_j(x_j)$. The derivation of eq. (4.29) follows directly from the form of the occlusion constraints (eq. (4.8)).

By taking the product of these messages $\bar{m}_{k,i(v)}(z_{i(v)})$ from all potential occluders x_k and normalizing, we may determine an approximation to the marginal occlusion probability $\vartheta_{i(v)} \triangleq \Pr[z_{i(v)} = 0]$. Because the color likelihood $p_C(\mathbf{v} \mid x_i, z_i)$ factorizes across pixels $v \in \Upsilon$, the BP approximation to $p_C(\mathbf{v} \mid x_i)$ may be written in terms of these marginal occlusion probabilites:

$$p_C(\mathbf{v} \mid x_i) \propto \prod_{v \in \Omega(x_i)} \left[\vartheta_{i(v)} + (1 - \vartheta_{i(v)}) \left(\frac{p_{\text{skin}}(v)}{p_{\text{bkgd}}(v)}\right)\right] \tag{4.30}$$

Intuitively, this equation downweights the color evidence at pixel v as the probability of that pixel's occlusion increases. The edge likelihood $p_E(\mathbf{v} \mid x_i)$ averages over z_i similarly:

$$p_E(\mathbf{v} \mid x_i) \propto \prod_{v \in \Pi(x_i)} \left[\vartheta_{i(v)} + (1 - \vartheta_{i(v)}) \left(\frac{p_{\text{on}}(v)}{p_{\text{off}}(v)}\right)\right] \tag{4.31}$$

The NBP estimate of $q_i(x_i)$ is then determined by Alg. 4.1 as before, but now using the occlusion–sensitive likelihood functions of eqs. (4.30, 4.31) in step 3.

■ 4.4.2 Message Propagation

When occlusion masks are considered, the BP message update (eq. (4.26)) for non–occlusion edges takes the following form:

$$\begin{aligned} m_{ji}(x_i) &\propto \int_{\mathcal{X}_j} \sum_{z_j} \psi_{ij}(x_i, x_j) \frac{q_j(x_j, z_j)}{m_{ij}(x_j)} \, dx_j \\ &\propto \int_{\mathcal{X}_j} \psi_{ij}(x_i, x_j) \frac{q_j(x_j)}{m_{ij}(x_j)} \, dx_j \end{aligned} \tag{4.32}$$

Because kinematic, structural, and kinematic potentials do not directly depend on the occlusion mask z_j, messages sent along those edges depend only on the belief $q_j(x_j)$ following marginalization over all possible occlusion masks. Thus, no modifications to the updates of Sec. 4.3.3, as summarized by Alg. 4.2, are necessary for occlusion–sensitive tracking.

4.4.3 Relation to Layered Representations

For some video sequences, the observed images are well described by a set of depth–ordered, smoothly deforming textured regions, or *layers* [156, 316]. Each layer is described by an intensity image, a motion field, and an alpha map indicating the pixel region occupied by that layer. Combined with a depth ordering of the layers, the alpha maps specify occlusion relationships, and thus determine which layer defines the observed intensity at each pixel.

Although our occlusion masks z share some heuristic similarities with layered representations, there are important differences. Alpha maps determine which pixels each layer occludes, while occlusion masks determine which pixels are occluded by one or more other hand components. Furthermore, layered representations are typically learned by aggregating consistent image motions in a bottom–up fashion, while our tracker uses the hand model's known 3D geometry to infer proper occlusion masks. Most importantly, to determine a pixel's observed intensity in a layered representation, the globally ordered alpha maps must be jointly considered. In contrast, occlusion masks allow each hand component to independently determine the pixels in its silhouette, and thereby efficiently compute image likelihoods in a distributed fashion.

4.5 Simulations

We now examine the empirical performance of the NBP hand tracker. All results are based on 720×480 images (or video sequences) recorded by a calibrated camera. The physical dimensions of the quadrics composing the hand model were measured offline. All messages were represented by $L = 200$ particles, and updated via the procedures summarized in Algs. 4.1 and 4.2.

NBP estimates the marginal distribution $q_i(x_i)$ of each hand component's 3D position and orientation x_i. To visualize these belief estimates, we first use gradient ascent to find the five most significant modes of the Gaussian mixture defining $q_i(x_i)$. We then project the edges of these 3D hand configurations, with intensity proportional to the corresponding mode's posterior probability. As illustrated in Fig. 4.8, the variability of these projections indicates the current uncertainty in the hand's 3D pose.

4.5.1 Refinement of Coarse Initializations

Given a single image, NBP may be used to progressively refine a coarse, user–supplied initialization into an accurate estimate of the hand's 3D pose. See Fig. 4.8 for two examples of such a refinement. In the second example, note that the initial finger positions are not only misaligned, but the user has supplied no information about the grasping configuration of the hand. By the fourth NBP iteration, however, the system has aligned all of the joints properly. In both images, a poorly aligned palm is eventually attracted to the proper location by well–fit fingers. For these examples, each NBP iteration (a complete update of all messages in the graph) requires about 1 minute on a Pentium IV workstation.

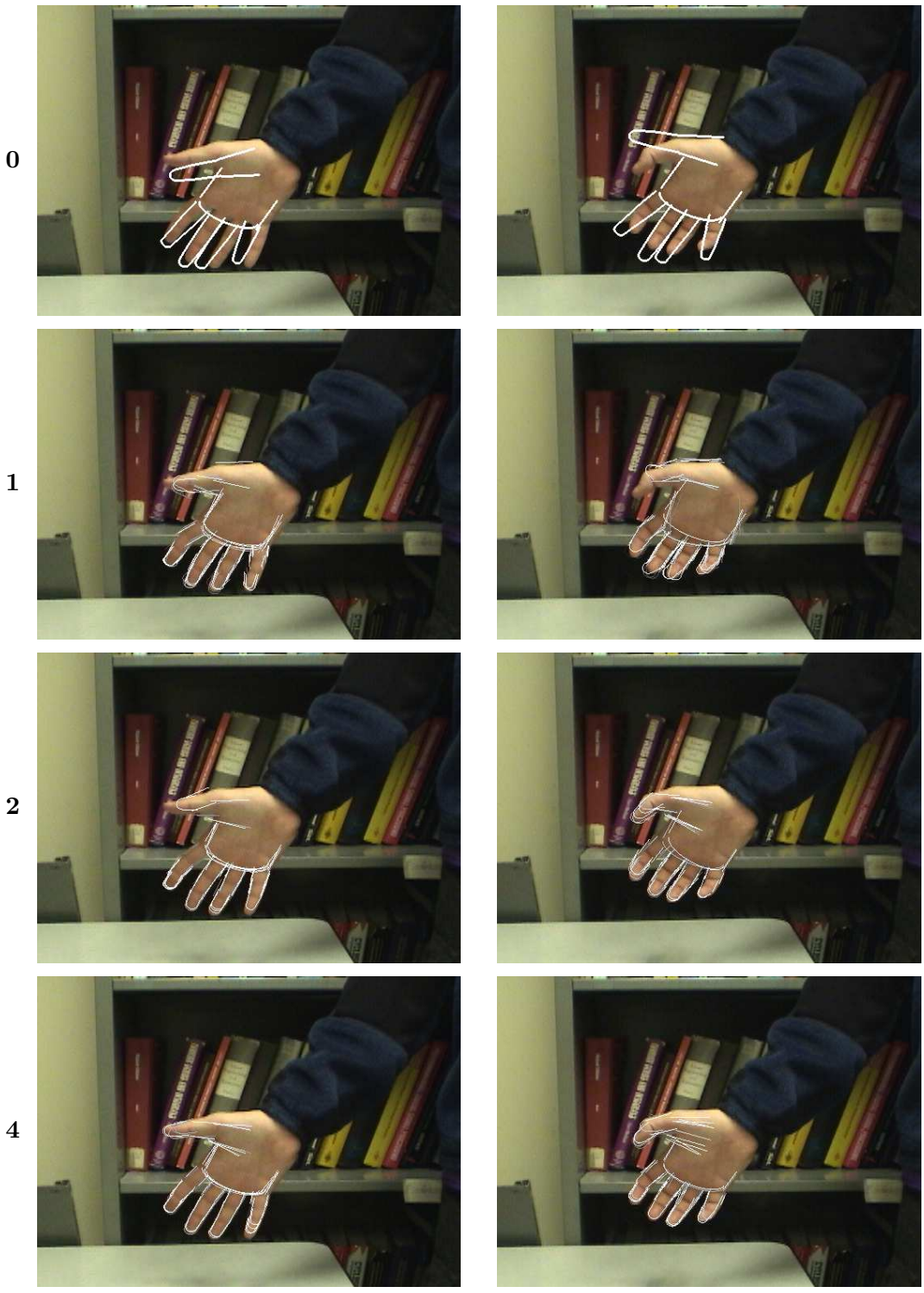


Figure 4.8. Two examples (columns) in which NBP iteratively refines coarse initial hand pose estimates. We show projections of the estimated 3D hand pose following 1, 2, and 4 iterations of the sequential message schedule illustrated in Fig. 4.7.

Occlusions Neglected | Distributed Occlusion Reasoning

Figure 4.9. Refinement of a coarse hand pose estimate via NBP assuming independent likelihoods for each finger (left), and using distributed occlusion reasoning (right). We display the middle finger's estimated pose in yellow, and the fourth (ring) finger's pose in magenta. When occlusions are neglected, NBP produces multimodal beliefs in which both fingers explain the same image data. When occlusions are considered, however, these fingers are properly separated.

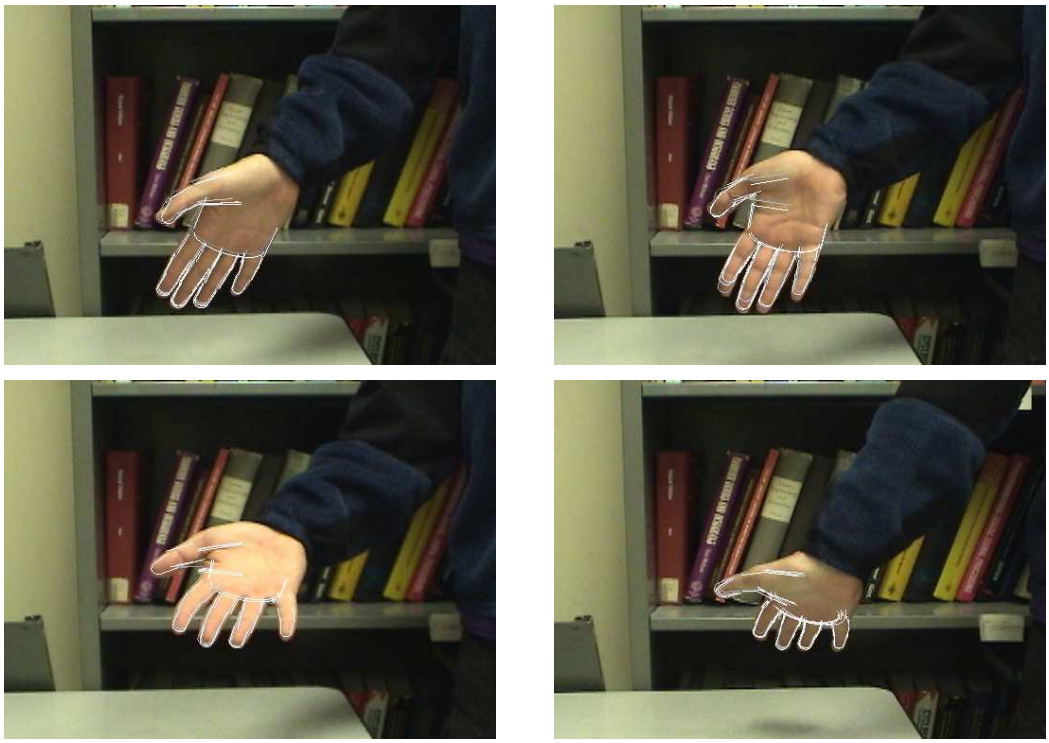


Figure 4.10. Four frames from a video sequence showing extrema of the hand's rigid motion, and projections of the NBP tracker's 3D pose estimates.

In Fig. 4.9, we show an initialization example with significant self–occlusion. When occlusion constraints are neglected, NBP improperly explains two fingers with the same image features, producing bimodal belief estimates. Via the distributed occlusion reasoning procedure of Sec. 4.4, however, NBP properly disambiguates these fingers, and produces a globally consistent hand pose estimate.

4.5.2 Temporal Tracking

Several video sequences demonstrating the NBP hand tracker are available online from the author's homepage.[1] For all sequences, the initial frame's hand pose was manually initialized. Total computation time for each video sequence, including all likelihood calculations, is approximately 4 minutes per frame. The first shows the hand rigidly moving in 3D space. End points of the hand's motion are shown in Fig. 4.10. NBP's pose estimates closely track the hand throughout the sequence, but are noisiest when the fingers point towards the camera because the sharp projection angle reduces the amount of image evidence. Note, however, that the estimates quickly lock back onto the true hand configuration when the hand rotates away from the camera.

The second video sequence exercises the hand model's joints, containing both individual finger motions and coordinated grasping motions (see Fig. 4.11). Our model supports all of these degrees of freedom, and maintains accurate estimates even when the ring finger is partially occluded by the middle finger (third row of Fig. 4.11). For this sequence, rough tracking is possible without occlusion reasoning [278], since all fingers are the same color and the background is unambiguous. However, we find that stability improves when occlusion reasoning is used to properly discount obscured edges and silhouettes.

4.6 Discussion

We have described the graphical structure underlying a 3D kinematic model of the hand, and used this model to build a tracking algorithm based on nonparametric belief propagation (NBP). Rather than estimating joint angles, our tracker directly infers the 3D pose of each hand component. This modular state representation allows edge and color cues to be independently evaluated for each finger, and then combined via the model's structural, kinematic, and temporal constraints. By introducing occlusion masks, NBP may consistently evaluate local image evidence even when the hand takes on self–occluding poses. Chap. 7 discusses potential generalizations of these methods, and implications for other vision applications, in more detail.

[1] As of May 2006, hand tracking videos are available at http://ssg.mit.edu/nbp/.

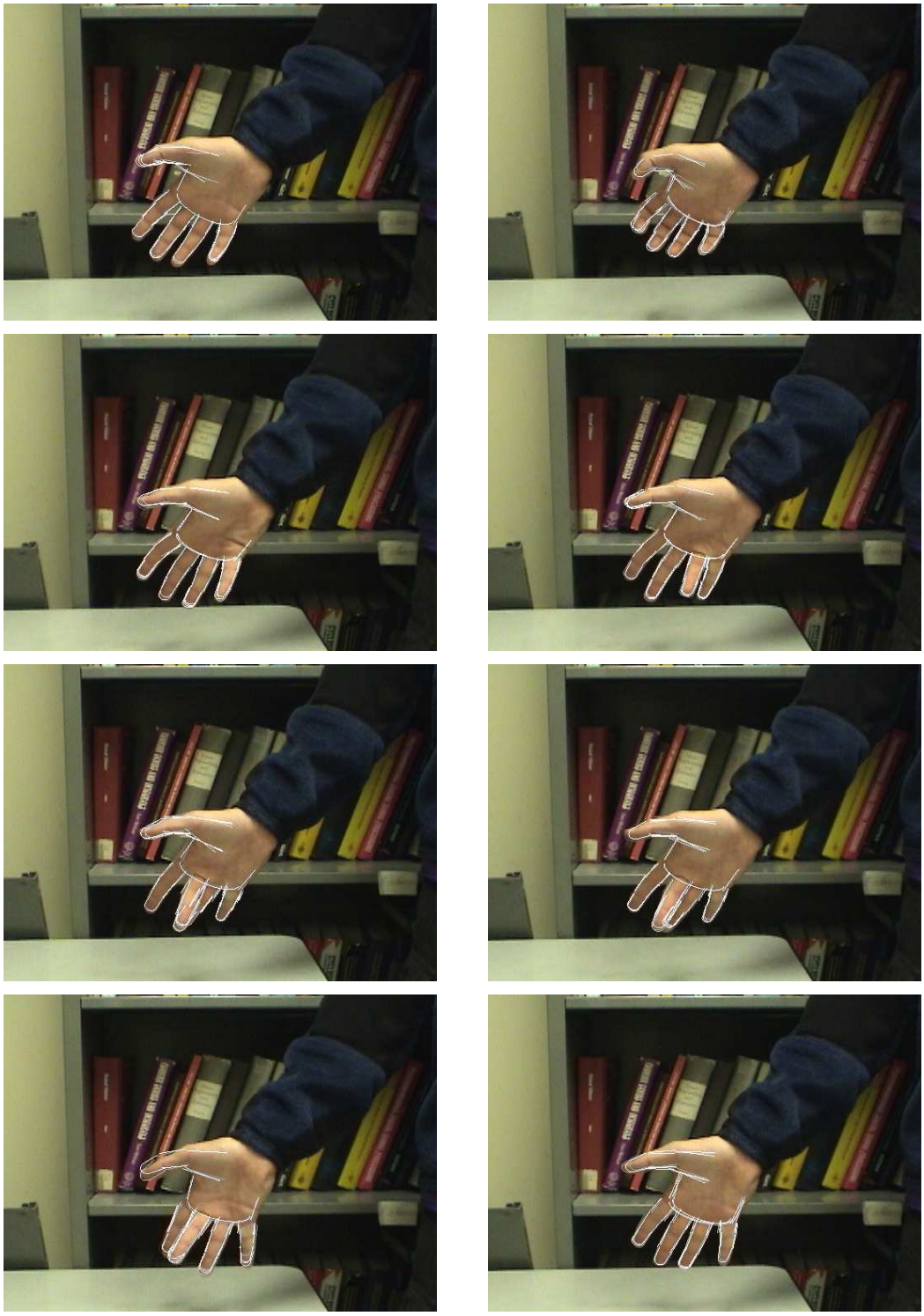


Figure 4.11. Eight frames from a video sequence in which the hand makes grasping motions and individual finger movements, and projections of the NBP tracker's 3D pose estimates. Note that the ring finger is accurately tracked through a partial occlusion by the middle finger (third row).

Chapter 5

Object Categorization using Shared Parts

VISUAL object recognition is complicated by wide variations in object appearance, and the paucity of large, representative datasets. This chapter develops hierarchical models which consider relationships among object categories during training. At the lowest level, this approach leads to significant computational savings by sharing a common set of features. More importantly, we show that integrated representations of multiple objects produce more robust predictions given few training images.

We begin in Sec. 5.1 by describing the sparse, affinely adapted image features underlying our recognition system. Several previous methods for object categorization discard the spatial locations of such interest points, treating the image as an unstructured "bag of features" [54, 266]. In Sec. 5.2, we describe a family of spatial transformations which allow flexible, consistent models for feature positions. Later results (Sec. 5.4 and 5.6) confirm that the spatial structure of image features is highly informative, and can significantly improve recognition performance.

Our hierarchical object appearance models are adapted from topic models originally used to analyze text documents [31, 289]. By incorporating spatial transformations, we construct robust part–based models for the visual appearance of object categories. In Sec. 5.3, we describe a parametric, fixed–order model which describes multiple object categories via a common set of shared parts. Sec. 5.5 then adapts the hierarchical Dirichlet process [289] to this task, and thus allows the *number* of latent parts to be learned automatically. We test both models on a dataset containing sixteen visual object categories (Sec. 5.4 and 5.6). Our results demonstrate the benefits of sharing parts, and show that nonparametric, Dirichlet process priors elegantly avoid potentially difficult model selection issues. Preliminary versions of these models, which were developed in collaboration with Dr. Antonio Torralba, were presented at the 2005 IEEE International Conference on Computer Vision [280].

■ 5.1 From Images to Invariant Features

Images can be stored, manipulated, and analyzed via a wide range of low–level formats. Rectangular grids of pixels provide one of the simplest, and most common, representa-

tions. While they are well matched to the sensors employed by modern digital cameras, raw pixels are often cumbersome for image analysis. In particular, the intensities of individual pixels are strongly dependent on nearby image regions, and unstable with respect to variations in three–dimensional (3D) viewpoint or lighting [91]. We thus consider alternative representations in which individual features are more informative.

Multiscale image descriptions, including wavelets [193] and steerable pyramids [93, 264], provide a popular framework for low–level image processing. Because wavelet coefficients derived from natural images have predictable statistical properties, they lead to effective denoising algorithms [269]. Steerable pyramids also provide some invariance to image–based translations, rotations, and scalings [264]. However, they do not directly compensate for more complex transformations induced by 3D pose changes. In addition, there are typically strong non–local dependencies among wavelet coefficients [269], due to the spatial structure of visual scenes.

In this chapter, we instead employ sparser, feature–based image representations. Interest points are first detected via criteria which are robust to changes in scale and lighting, and affinely adapted to correct for 3D pose variations [205, 207]. Unlike wavelet representations, the number and location of these features is adapted to the particular details of each individual scene. This approach reduces the dimensionality of the image representation, and also leads to features which are less directly inter–dependent. Coupled with a family of gradient–based appearance descriptors [188, 206], these features simplify the modeling of object categories by focusing on the most salient, repeatable image structures.

We emphasize that our object appearance models could be adapted to any image representation involving a finite set of interest points. In particular, while the features we employ are known to perform well in geometric correspondence tasks [206, 207], it is likely that other approaches will prove more suitable for object categorization.

■ 5.1.1 Feature Extraction

For each grayscale training or test image, we begin by detecting a set of affinely adapted interest regions. The shape of each region is described by a 2D ellipse, as illustrated in Fig. 5.1. We consider three different criteria for region detection, which emphasize complementary aspects of visual scenes. In particular, *Harris–affine* invariant regions [205] detect corner–like image structure by finding points whose surrounding intensities have significant second derivatives. A characteristic scale for each corner is then determined via a Laplacian of Gaussian operator [188, 205]. Finally, region shape is adapted according to the second moments of neighboring image pixels.

We also consider a set of *maximally stable extremal* regions [199], which are derived from a watershed segmentation algorithm. For each of several intensity thresholds, the connected components of the corresponding binary image are determined. Those components which are stable across a wide range of threshold values are then selected. As illustrated in Fig. 5.1, this approach favors larger, more homogeneous image regions. Software for the detection of maximally stable and Harris–affine regions is available from

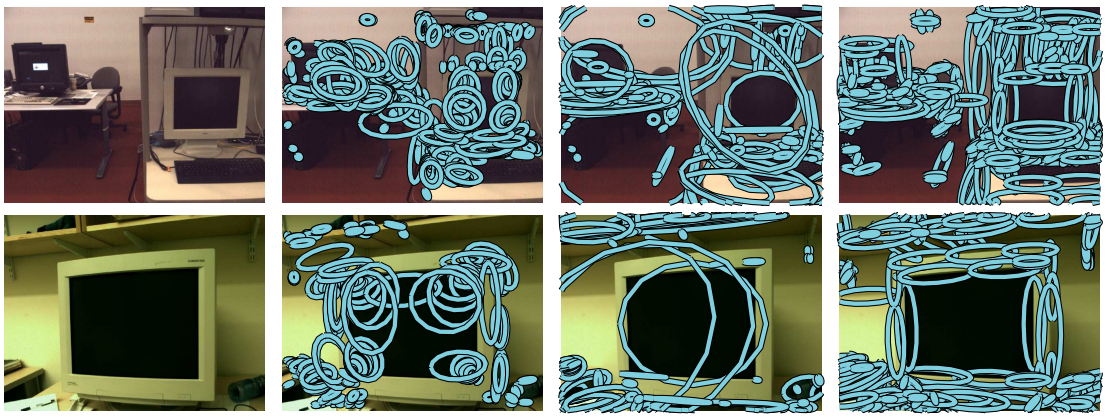


Figure 5.1. Three types of interest operators applied to two office scenes: Harris–affine corners (left), maximally stable extremal regions (center), and clustered Canny edges (right).

the Oxford University Visual Geometry Group [343].

The preceding features were originally designed to determine accurate geometric correspondences across multiple 3D views. For object recognition tasks, however, simpler edge–based features are often highly informative [18, 20]. To exploit this, we first find candidate edges via a standard Canny detector [38, 91]. Edges at adjacent pixels are then linked into segments, and lines determined by dividing these segments at points of high curvature [174]. These lines then form the major axes of elliptical edge shape descriptors, whose minor axes are taken to be 10% of that length. Given the density at which interest regions are detected (see Fig. 5.1), these features essentially provide a multiscale over–segmentation of the image [241].

Note that greedy, low–level interest operators are inherently noisy: even state–of–the–art detectors sometimes miss salient regions, and select features which do not align with real 3D scene structure (see Fig. 5.1 for examples). We handle this issue by extracting large feature sets, so that many regions are likely to be salient. It is then important to design recognition algorithms which exploit this redundancy, rather than relying on a small set of key features.

■ 5.1.2 Feature Description

Following several recent papers [54, 79, 81, 181, 266], we use SIFT descriptors [188] to characterize the appearance of each elliptical interest region. This approach provides some invariance to lighting and pose changes, and was more effective than raw pixel patches [302] in our experiments.

For each interest region, a Gaussian window is first used to estimate an orientation histogram, the peak of which provides a reference orientation for that region. This orientation, and the corresponding feature's scale, are then used to define a four–by–four grid of image pixels. Within each block in this grid, a histogram of gradient magnitudes at each of eight orientations is constructed. The overall SIFT descriptor

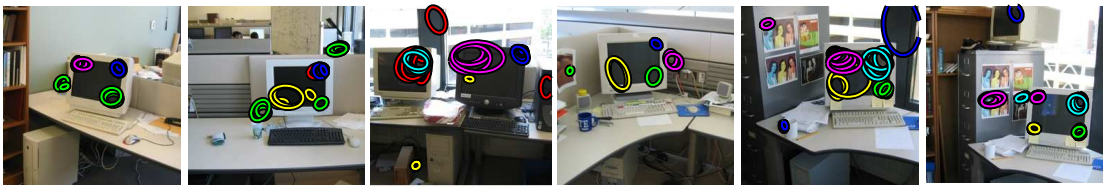


Figure 5.2. A subset of the affine covariant features (ellipses) detected in images of office scenes. In five different colors, we show the features corresponding to the five discrete vocabulary words which most frequently align with computer screens in the training images.

then equals these $4 \times 4 \times 8 = 128$ orientation energies, suitably normalized to correct for contrast and saturation effects [188]. We note that this approach is closely related to the *shape context* descriptor [18].

To simplify learning algorithms, we convert each raw, 128–dimensional SIFT descriptor to a vector quantized discrete value [79, 266]. In particular, for each training database, we use K–means clustering to identify a finite dictionary of W appearance patterns, where each of the three feature types is mapped to a disjoint set of visual "words". We set the total dictionary size via cross–validation; typically, $W \approx 1{,}000$ seems appropriate for object categorization tasks. In some experiments, we further improve discriminative power by dividing the affinely adapted regions according to their shape. In particular, we separate edges by orientation (horizontal versus vertical), while Harris–affine and maximally stable regions are divided into three groups (roughly circular, versus horizontally or vertically elongated). An expanded dictionary then jointly encodes the appearance and coarse shape of each feature.

Using this visual dictionary, the i^{th} interest region in image j is described by its detected image position v_{ji}, and the discrete appearance word w_{ji} with minimal Euclidean distance [188]. Let $\mathbf{w}_j$ and $\mathbf{v}_j$ denote the appearance and two–dimensional position, respectively, of the N_j features in image j. Fig. 5.2 illustrates some of the appearance words extracted from a database of office scenes.

■ 5.1.3 Object Recognition with Bags of Features

Using image features similar to those described in Sec. 5.1.1 and 5.1.2, discriminative machine learning methods have been used for visual object recognition [54]. In addition, the latent Dirichlet allocation model [31] has been adapted to discover object categories from images of single objects [266], categorize natural scenes [79], and (with a slight extension) parse presegmented captioned images [14]. However, following an initial stage of low–level feature extraction [54, 79, 266] or segmentation [14], these methods ignore spatial information, discarding feature positions $\mathbf{v}_j$ and treating the image as an unstructured bag of words $\mathbf{w}_j$. In this chapter, we develop richer models which consistently incorporate spatial relationships.

Figure 5.3. Twelve office scenes in which computer screens have been highlighted (yellow). Note the large variability in the image locations and scales at which objects are observed.

■ 5.2 Capturing Spatial Structure with Transformations

Figure 5.3 shows several images which illustrate the challenges in developing visual scene models incorporating feature positions. Due to variability in three–dimensional object location and pose, the absolute position at which features are observed provides little information about their corresponding category. Nevertheless, there is clearly significant, predictable spatial structure in natural scenes. For example, many object categories have regular internal geometries and textures. More generally, contextual relationships among objects, like the desks supporting the monitors depicted in Fig. 5.3, lend global structure to visual scenes.

This chapter adapts hierarchical topic models, including latent Dirichlet allocation (LDA) [31] and the hierarchical Dirichlet process (HDP) [289] (see Secs. 2.2.4 and 2.5.4), to model visual object categories. Recall that the LDA and HDP models describe different groups of data by reusing *identical* cluster parameters θ_k in varying proportions. Thus, applied directly to features incorporating both position and appearance, these topic models would need a separate global cluster for every possible location of each object category. This approach does not sensibly describe the spatial structure underlying scenes like those in Fig. 5.3, and would not adequately generalize to images captured in new environments.

A more effective model of visual scenes would allow the same global cluster to describe objects at many different locations. To accomplish this, we augment topic models

with *transformation* variables, thereby shifting global clusters from a "canonical" coordinate frame to the object positions underlying a particular image. Let $\tau(\theta;\rho)$ denote a family of transformations of the parameter vector θ, indexed by ρ. For computational reasons, we assume that parameter transformations are invertible, and have a complementary *data transformation* $\tilde{\tau}(v;\rho)$ defined so that

$$f(v \mid \tau(\theta;\rho)) = \frac{1}{Z(\rho)} f(\tilde{\tau}(v;\rho) \mid \theta) \tag{5.1}$$

The normalization constant $Z(\rho)$, which is determined by the transformation's Jacobian [229], is assumed independent of the underlying parameters θ. Using eq. (5.1), model transformations $\tau(\theta;\rho)$ are equivalently expressed by a change $\tilde{\tau}(v;\rho)$ of the observations' coordinate system. The following sections provide specific examples of transformations satisfying these conditions.

■ 5.2.1 Translations of Gaussian Distributions

Consider the mean and covariance parameters $\theta = (\mu, \Lambda)$ of a d–dimensional Gaussian distribution. In the simplest case, we consider transformations ρ which translate that Gaussian's mean:

$$\tau(\mu, \Lambda; \rho) = (\mu + \rho, \Lambda) \tag{5.2}$$

From the general form of Gaussian densities (see eq. (2.47)), we then have

$$\begin{aligned} \mathcal{N}(v; \mu+\rho, \Lambda) &= \frac{1}{(2\pi)^{d/2}|\Lambda|^{1/2}} \exp\left\{-\frac{1}{2}(v-(\mu+\rho))^T\Lambda^{-1}(v-(\mu+\rho))\right\} \\ &= \frac{1}{(2\pi)^{d/2}|\Lambda|^{1/2}} \exp\left\{-\frac{1}{2}((v-\rho)-\mu)^T\Lambda^{-1}((v-\rho)-\mu)\right\} \\ &= \mathcal{N}(v-\rho; \mu, \Lambda) \end{aligned} \tag{5.3}$$

Translations are thus equivalent to an opposing shift of the observations, so that the data transformation corresponding to eq. (5.2) equals

$$\tilde{\tau}(v;\rho) = v - \rho \tag{5.4}$$

In later sections, we use this relationship to tractably combine information from objects observed at different absolute image positions.

■ 5.2.2 Affine Transformations of Gaussian Distributions

Generalizing the translation of eq. (5.2), we consider arbitrary *affine* transformations $v = Bu + \rho$, $B \in \mathbb{R}^{d\times d}$, of a latent Gaussian random variable $u \sim \mathcal{N}(\mu, \Lambda)$. For any such transformation, the corresponding mean and covariance parameters equal

$$\tau(\mu, \Lambda; \rho, B) = (B\mu + \rho, B\Lambda B^T) \tag{5.5}$$

This parameterization arises naturally in computer vision applications [91], and ensures that the transformed covariance matrix remains positive semidefinite.

Suppose that the affine transformation is specified by an invertible matrix B. Expanding the Gaussian distribution corresponding to the transformed parameters of eq. (5.5), we then have

$$\begin{aligned}
&\mathcal{N}\big(v; B\mu + \rho, B\Lambda B^T\big) \\
&\qquad \propto \frac{1}{|B\Lambda B^T|^{1/2}} \exp\left\{-\frac{1}{2}\big(v - (B\mu+\rho)\big)^T \big(B\Lambda B^T\big)^{-1}\big(v - (B\mu+\rho)\big)\right\} \\
&\qquad \propto \frac{1}{|B| \cdot |\Lambda|^{1/2}} \exp\left\{-\frac{1}{2}\big((v-\rho) - B\mu\big)^T B^{-T}\Lambda^{-1}B^{-1}\big((v-\rho) - B\mu\big)\right\} \\
&\qquad \propto \frac{1}{|B|}\mathcal{N}\big(B^{-1}(v-\rho); \mu, \Lambda\big)
\end{aligned} \tag{5.6}$$

The data transform induced by the affine transformation of eq. (5.5) thus equals

$$\tilde{\tau}(v; \rho, B) = B^{-1}(v - \rho) \tag{5.7}$$

In this case, the normalization constant of eq. (5.6) could also be derived from the determinant $|B|$ of the Jacobian corresponding to the affine map $v = Bu + \rho$ (see [229]). More generally, however, one can consider parameter transformations $\tau(\theta; \rho)$ which do not directly correspond to a simple mapping of an underlying random variable.

In many applications, specializations of the general affine transformation of eq. (5.5) are of interest. For example, rigid body motion is described by orthogonal rotation matrices satisfying $B^{-1} = B^T$. Alternatively, homogeneous scalings $B = \beta I$ often arise in image–based approaches to object recognition [82]. Such restricted transformation families may also lead to simpler forms for the prior distributions over transformations employed in later sections.

■ 5.2.3 Related Work

Transformations have been previously used to learn mixture models which encode the objects in a video sequence using a fixed number of layers [97, 156]. In contrast, the hierarchical models developed in this thesis allow transformed mixture components to be shared among different object and scene categories, improving generalization performance. Nonparametric density estimates of transformations [209], coupled with template–based appearance models, have also been used to transfer knowledge between related object (in particular, digit) recognition tasks [210]. By embedding transformations in a nonparametric hierarchical model, we consider more complex tasks in which the number of objects in a visual scene, or the number of shared parts underlying an object's appearance, is uncertain.

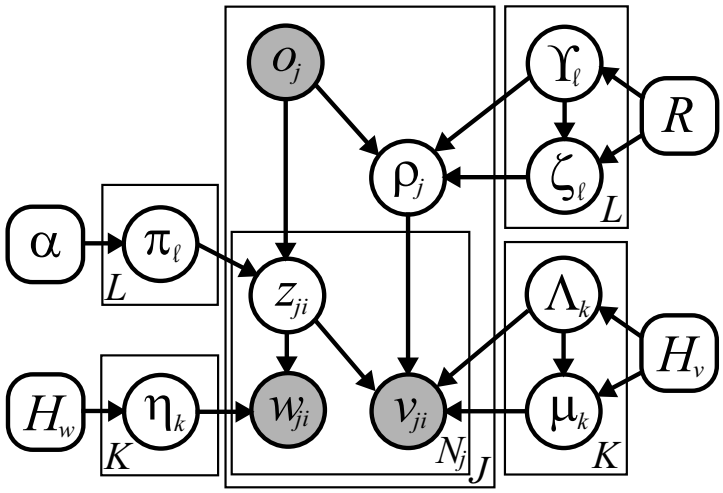


Figure 5.4. A parametric, fixed–order model which describes the visual appearance of L object categories via a common set of K shared parts. The j^{th} image depicts an instance of object category o_j, whose position is determined by the reference transformation ρ_j. The appearance w_{ji} and position v_{ji}, relative to ρ_j, of visual features are determined by assignments $z_{ji} \sim \boldsymbol{\pi}_{o_j}$ to latent parts.

■ 5.3 Learning Parts Shared by Multiple Objects

We begin by developing a parametric, hierarchical model for images dominated by a single object [280]. The representation of objects as a collection of spatially constrained parts has a long history in vision [89]. In the directed graphical model of Fig. 5.4, parts are formalized as groups of features that are spatially clustered, and have predictable appearances. Each of the L object categories is in turn characterized by a probability distribution $\boldsymbol{\pi}_\ell$ over a common set of K shared parts. For this *fixed–order* object appearance model, K is set to some known, constant value.

Consider the generative process for image j, which depicts object o_j and contains N_j features $(\mathbf{w}_j, \mathbf{v}_j)$. As described in Sec. 5.1, the i^{th} feature in image j is represented by a two–dimensional image position v_{ji}, and an appearance w_{ji} drawn from one of W discrete categories. We model feature positions relative to an image–specific *reference transformation*, or coordinate frame, ρ_j. Lacking other information distinguishing the images of category o_j, De Finetti's Theorem (see Sec. 2.2.4) implies that these transformations are independently sampled from some common prior distribution. The form of this distribution must be adapted to the chosen family of transformations $\tau(\cdot\,; \rho_j)$ (see Sec. 5.2). In the datasets considered by this chapter, objects are roughly scale–normalized and centered, so we assign the image positions of each category unimodal, Gaussian transformation densities:

$$\rho_j \sim \mathcal{N}\big(\zeta_{o_j}, \Upsilon_{o_j}\big) \qquad\qquad j = 1, \ldots, J \tag{5.8}$$

As shown in Fig. 5.4, these transformation distributions are regularized by conjugate, normal–inverse–Wishart priors $(\zeta_\ell, \Upsilon_\ell) \sim R$.

To capture the internal structure of objects, we define K distinct parts which generate features with different typical appearance w_{ji} and position v_{ji}, relative to ρ_j. The particular parts $\mathbf{z}_j = (z_{j1}, \ldots, z_{jN_j})$ associated with each feature are independently sampled from a category–specific multinomial distribution:

$$z_{ji} \sim \boldsymbol{\pi}_{o_j} \qquad i = 1, \ldots, N_j \tag{5.9}$$

When learning object appearance models from training data, we assign conjugate Dirichlet priors $\boldsymbol{\pi}_\ell \sim \mathrm{Dir}(\alpha)$ to these part association probabilities. Each part is then defined by a multinomial distribution $\eta_k \in \Pi_{W-1}$ on the discrete set of appearance descriptors, and a Gaussian distribution $\mathcal{N}(\mu_k, \Lambda_k)$ on feature positions:

$$w_{ji} \sim \eta_{z_{ji}} \tag{5.10}$$

$$v_{ji} \sim \mathcal{N}(\tau(\mu_{z_{ji}}, \Lambda_{z_{ji}}; \rho_j)) \tag{5.11}$$

As discussed in Sec. 5.2, the transformation of eq. (5.11) models feature positions relative to the object's pose in each particular image. For datasets which have been normalized to account for orientation and scale variations, transformations are defined as in eq. (5.2) to shift the part's mean:

$$v_{ji} \sim \mathcal{N}\big(\mu_{z_{ji}} + \rho_j, \Lambda_{z_{ji}}\big) \tag{5.12}$$

However, in principle the model could be easily generalized to capture more complex object pose variations.

Marginalizing the unobserved assignments z_{ji} of features to parts, we find that the graph of Fig. 5.4 defines object appearance via a finite mixture model:

$$p(w_{ji}, v_{ji} \mid \rho_j, o_j = \ell) = \sum_{k=1}^{K} \pi_{\ell k} \eta_k(w_{ji}) \mathcal{N}(v_{ji}; \mu_k + \rho_j, \Lambda_k) \tag{5.13}$$

In this construction, parts are latent variables which capture predictable dependencies between the location and appearance of observed features. Reference transformations then allow a common set of parts to model datasets which are not spatially aligned. Like latent Dirichlet allocation (LDA) [31], which shares topics among related groups, these parts are shared among multiple object categories.

As with other mixture models (see Sec. 2.2.4), it is sometimes useful to express this object model via a set of discrete distributions on the space of part parameters. Letting $\theta_k = (\eta_k, \mu_k, \Lambda_k)$ denote the appearance and position parameters of the k^{th} part, object ℓ then has the following representation:

$$G_\ell(\theta) = \sum_{k=1}^{K} \pi_{\ell k} \delta(\theta, \theta_k) \tag{5.14}$$

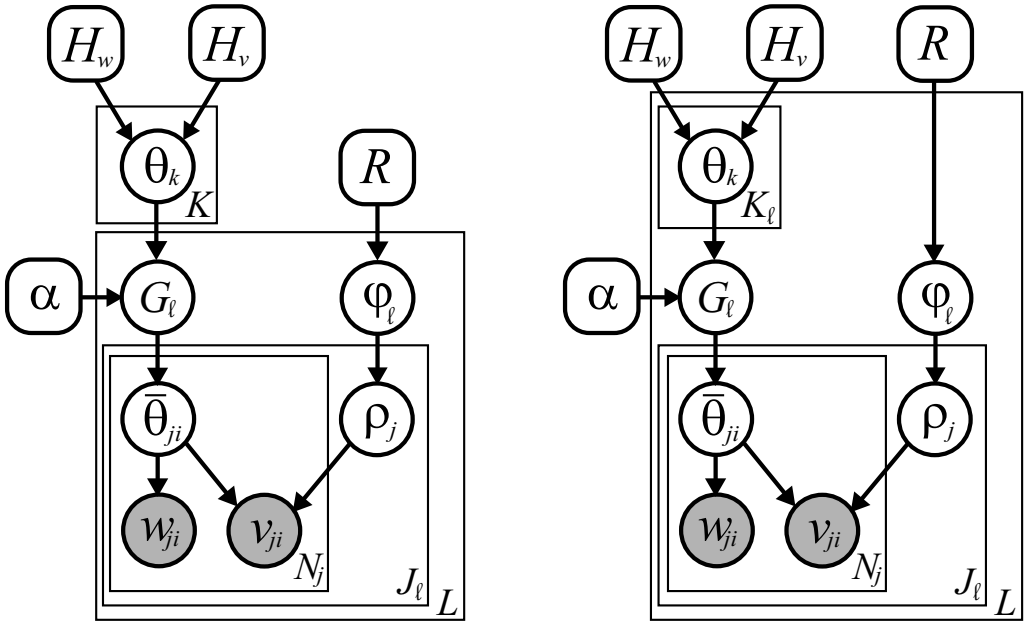


Figure 5.5. Alternative, distributional form of the fixed–order object model of Fig. 5.4. *Left:* An integrated model of L object categories, in which G_ℓ is a discrete distribution on the common set of K shared parts $\theta_k = (\eta_k, \mu_k, \Lambda_k)$. $\bar{\theta}_{ji} \sim G_\ell$ are the part parameters used to generate feature (w_{ji}, v_{ji}). Reference transformations ρ_j have object–specific priors parameterized by $\varphi_\ell = (\zeta_\ell, \Upsilon_\ell)$. *Right:* A related model in which each of the L object categories is described by an independent set of K_ℓ parts.

Images of this object are generated by choosing a reference transformation ρ_j according to eq. (5.8), and then sampling each feature (w_{ji}, v_{ji}) via part parameters $\bar{\theta}_{ji} = (\bar{\eta}_{ji}, \bar{\mu}_{ji}, \bar{\Lambda}_{ji})$ independently drawn from G_ℓ:

$$\begin{aligned} (\bar{\eta}_{ji}, \bar{\mu}_{ji}, \bar{\Lambda}_{ji}) &\sim G_\ell \\ w_{ji} &\sim \bar{\eta}_{ji} \\ v_{ji} &\sim \mathcal{N}\big(\bar{\mu}_{ji} + \rho_j, \bar{\Lambda}_{ji}\big) \end{aligned} \tag{5.15}$$

Fig. 5.5 provides a directed graphical model summarizing this equivalent, alternative form, and contrasts it with a similar model in which parts are not shared among object categories. Note that these graphs represent object categories implicitly by grouping the J_ℓ training images of each category, rather than via an explicit indicator variable o_j as in Fig. 5.4. In later sections, we provide experimental results which evaluate the benefits of shared parts when learning from small datasets.

■ 5.3.1 Related Work: Topic and Constellation Models

If reference transformations are removed from the graphical models of Figs. 5.4 and 5.5, we recover a variant of the *author–topic model* [247], where objects correspond to authors, features to words, and parts to the latent topics underlying a given text corpus.

The LDA model [31] (see Sec. 2.2.4) is in turn a special case in which each document has its own topic distribution, and authors (objects) are not explicitly modeled. This hierarchical structure shares information in two distinct ways: parts combine the same features in different spatial configurations, and objects reuse the same parts in different proportions. The learning algorithms developed in the following sections are free to give each object category its own parts, or "borrow" parts from other objects, depending on which better explains the training images. As we demonstrate in Sec. 5.4, this sharing can significantly improve detection performance.

When modeling a single object category, our model also shares many features with constellation models [82, 89, 318], particularly recent extensions which use Bayesian priors when learning from few examples [77, 78]. The principal difference is that constellation models assume that each part generates at most one observed feature, creating a combinatorial data association problem for which greedy approximations are needed to learn more than a few parts [129]. In contrast, our model associates parts with expected *proportions* of the observed features. This allows several different features to provide evidence for a given part, and thus seems better matched to the dense, overlapping feature detectors described in Sec. 5.1.1. Furthermore, by not placing hard constraints on the number of features assigned to each part, we are able to develop simple learning algorithms which scale linearly, rather than exponentially, with the number of parts.

5.3.2 Monte Carlo Feature Clustering

We now develop a Monte Carlo method which learns the parameters of our fixed–order object appearance model. We assume that all hyperparameters (rounded boxes in Fig. 5.4) have fixed, known values, and that training images are labeled by the single object o_j they depict. To simplify the learning process, we also employ prior distributions for the part and transformation densities that have a conjugate form [21, 107] (see Sec. 2.1.2). In particular, the multinomial appearance distributions are assigned symmetric Dirichlet priors $H_w = \mathrm{Dir}(\lambda)$, while the Gaussian part and reference transformation densities have normal–inverse–Wishart priors H_v and R.

We begin by assuming that all objects occur at roughly the same position in each image, so that the reference transformations ρ_j may be neglected. Certain standard object recognition datasets [78], as well as systems which use motion or saliency cues to focus attention [315], satisfy this assumption. In this case, the model is similar to the author–topic model [247], except that an additional observation (the position v_{ji}) is associated with each part indicator z_{ji}. Following [123, 247], we learn this model's parameters by iteratively sampling part assignments $\mathbf{z}$ using likelihoods which analytically marginalize part probabilities $\boldsymbol{\pi}$ and parameters $\boldsymbol{\theta} = \{\eta_k, \mu_k, \Lambda_k\}_{k=1}^K$. This Rao–Blackwellized Gibbs sampler, which generalizes the mixture model Gibbs sampler of Alg. 2.2, provides approximate samples from the posterior distribution $p(\mathbf{z} \mid \mathbf{w}, \mathbf{v}, \mathbf{o})$ which may then be used to estimate the underlying parameters.

Let $\mathbf{z}_{\setminus ji}$ denote all part assignments excluding z_{ji}, and define $\mathbf{w}_{\setminus ji}$ and $\mathbf{v}_{\setminus ji}$ similarly. The Gibbs sampler iteratively fixes the part assignments $\mathbf{z}_{\setminus ji}$ for all but one feature,

and then samples a part z_{ji} for the remaining feature from the induced conditional distribution. From the object model's Markov properties (see Fig. 5.4), the posterior distribution over part assignments factors as follows:

$$p(z_{ji} \mid \mathbf{z}_{\setminus ji}, \mathbf{w}, \mathbf{v}, \mathbf{o}) \;\propto\; p(z_{ji} \mid \mathbf{z}_{\setminus ji}, o_j)\, p(w_{ji} \mid \mathbf{z}, \mathbf{w}_{\setminus ji})\, p(v_{ji} \mid \mathbf{z}, \mathbf{v}_{\setminus ji}) \tag{5.16}$$

Let C_{kw}^{-i} denote the number of times appearance descriptor w is assigned to part k by $\mathbf{z}_{\setminus ji}$, and $N_{\ell k}^{-i}$ the number of features in images of object ℓ assigned to part k. Using standard expressions for the predictive likelihoods induced by Dirichlet priors (see Sec. 2.1.3), the first two terms of eq. (5.16) can be written as

$$p(z_{ji} = k \mid \mathbf{z}_{\setminus ji}, o_j = \ell) = \frac{N_{\ell k}^{-i} + \alpha/K}{\sum_{k'} N_{\ell k'}^{-i} + \alpha} \tag{5.17}$$

$$p(w_{ji} = w \mid z_{ji} = k, \mathbf{z}_{\setminus ji}, \mathbf{w}_{\setminus ji}) = \frac{C_{kw}^{-i} + \lambda/W}{\sum_{w'} C_{kw'}^{-i} + \lambda} \tag{5.18}$$

Note that these probabilities are simply the raw proportions defined by the part assignments $\mathbf{z}_{\setminus ji}$, regularized by the pseudo–counts contributed by the Dirichlet priors. Similarly, the position likelihood depends on the set of features which $\mathbf{z}_{\setminus ji}$ currently assigns to the same part:

$$p(v_{ji} \mid z_{ji} = k, \mathbf{z}_{\setminus ji}, \mathbf{v}_{\setminus ji}) = p\big(v_{ji} \mid \{v_{j'i'} \mid z_{j'i'} = k, (j', i') \neq (j, i)\}\big) \tag{5.19}$$

$$\approx \mathcal{N}(v_{ji}; \hat{\mu}_k, \hat{\Lambda}_k) \tag{5.20}$$

As discussed in Sec. 2.1.4, normal–inverse–Wishart prior distributions induce multivariate Student-t predictive likelihoods. The Gaussian likelihood of eq. (5.20) provides an accurate approximation when the mean $\hat{\mu}_k$ and covariance $\hat{\Lambda}_k$ are determined by regularized moment–matching of that part's assigned features, as in eq. (2.64).

Combining eqs. (5.17, 5.18, 5.20), we may evaluate eq. (5.16) for each of the K candidate assignments z_{ji}, and sample a new part for that feature. Alg. 5.1 summarizes one possible Rao–Blackwellized Gibbs sampler based on these predictive distributions. As in eq. (5.20), we use $(\hat{\mu}_k, \hat{\Lambda}_k)$ to denote the Gaussian likelihood determined via eq. (2.64) from the features currently assigned to part k. For compactness, we define $(\hat{\mu}_k, \hat{\Lambda}_k) \oplus v_{ji}$ to be an operator which updates a normal–inverse–Wishart posterior based on a new feature v_{ji} (see eqs. (2.62, 2.63)). Similarly, $(\hat{\mu}_k, \hat{\Lambda}_k) \ominus v_{ji}$ is defined to remove v_{ji} from the posterior statistics of part k. To efficiently determine these updates, our implementation caches the sum of the positions of the features assigned to each part, as well as the Cholesky decomposition of their outer products (see Sec. 2.1.4).

Given a training set with J images, each containing N features, a Gibbs sampling update of every feature assignment requires $\mathcal{O}(KJN)$ operations. These assignments also implicitly define posterior distributions for the part parameters, which can be easily inferred if desired (see Alg. 5.1, step 3). Convergence is improved by resampling features in a different, randomly chosen order at each iteration [246].

Given previous part assignments $\mathbf{z}_j^{(t-1)}$ for the N_j features in image j, which depicts object category $o_j = \ell$, sequentially sample new assignments as follows:

1. Sample a random permutation $\tau(\cdot)$ of the integers $\{1, \dots, N_j\}$.
2. Set $\mathbf{z}_j = \mathbf{z}_j^{(t-1)}$. For each $i \in \{\tau(1), \dots, \tau(N_j)\}$, sequentially resample z_{ji} as follows:
   (a) Remove feature (w_{ji}, v_{ji}) from the cached statistics for its current part $k = z_{ji}$:
   $$\begin{aligned} N_{\ell k} &\leftarrow N_{\ell k} - 1 \\ C_{kw} &\leftarrow C_{kw} - 1 \qquad\qquad w = w_{ji} \\ (\hat{\mu}_k, \hat{\Lambda}_k) &\leftarrow (\hat{\mu}_k, \hat{\Lambda}_k) \ominus v_{ji} \end{aligned}$$
   (b) For each of the K parts, determine the predictive likelihood
   $$f_k(w_{ji} = w, v_{ji}) = \left(\frac{C_{kw} + \lambda/W}{\sum_{w'} C_{kw'} + \lambda} \right) \cdot \mathcal{N}(v_{ji}; \hat{\mu}_k, \hat{\Lambda}_k)$$
   (c) Sample a new part assignment z_{ji} from the following multinomial distribution:
   $$z_{ji} \sim \frac{1}{Z_i} \sum_{k=1}^{K} (N_{\ell k} + \alpha/K) f_k(w_{ji}, v_{ji}) \delta(z_{ji}, k) \qquad Z_i = \sum_{k=1}^{K} (N_{\ell k} + \alpha/K) f_k(w_{ji}, v_{ji})$$
   (d) Add feature (w_{ji}, v_{ji}) to the cached statistics for its new part $k = z_{ji}$:
   $$\begin{aligned} N_{\ell k} &\leftarrow N_{\ell k} + 1 \\ C_{kw} &\leftarrow C_{kw} + 1 \qquad\qquad w = w_{ji} \\ (\hat{\mu}_k, \hat{\Lambda}_k) &\leftarrow (\hat{\mu}_k, \hat{\Lambda}_k) \oplus v_{ji} \end{aligned}$$
3. Set $\mathbf{z}_j^{(t)} = \mathbf{z}_j$. Optionally, part weights and parameters may be sampled as follows:
   $$\begin{aligned} \boldsymbol{\pi}_\ell^{(t)} &\sim \mathrm{Dir}(N_{\ell 1} + \alpha/K, \dots, N_{\ell K} + \alpha/K) \\ \eta_k^{(t)} &\sim \mathrm{Dir}(C_{k1} + \lambda/W, \dots, C_{kW} + \lambda/W) \qquad\qquad k = 1, \dots, K \end{aligned}$$
   Position parameters $\{\mu_k^{(t)}, \Lambda_k^{(t)}\}_{k=1}^K$ follow normal–inverse–Wishart distributions, with parameters determined from currently assigned features via eqs. (2.62, 2.63).

Algorithm 5.1. Rao–Blackwellized Gibbs sampler for the K part, fixed–order object model of Fig. 5.4, excluding reference transformations. We illustrate the sequential resampling of all feature assignments $\mathbf{z}_j$ in the j^{th} training image. A full iteration of the Gibbs sampler applies these updates to all images in a randomly chosen order. For efficiency, we cache and recursively update counts $N_{\ell k}$ of the features assigned to each part, as well as statistics $\{C_{kw}, \hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K$ of those features' appearance and position.

■ 5.3.3 Learning Part–Based Models of Facial Appearance

To provide intuition for the part–based object appearance model of Fig. 5.4, we consider a set of 64 manually cropped and aligned images of faces from the Caltech database [318]. Fig. 5.6 shows two sample training images. We examine a more complex dataset containing multiple object categories in Sec. 5.4. For each training image, we first extracted interest regions as described in Sec. 5.1.1. SIFT descriptors [188] of these interest points were then mapped to one of $W = 600$ appearance words. Qualitatively, results for this dataset seem relatively insensitive to the part–based model's hyperparameters (see

Fig. 5.4), which were set as described in Sec. 5.4.1.

We compare models with $K = 10$ and $K = 25$ parts, as learned via the Gibbs sampler of Alg. 5.1. In Fig. 5.6, we illustrate parts corresponding to the 300^{th} sampling iteration, based on subsets of the training images of varying size. For the 10–part model, 4 training images are sufficient to learn a coarse description of facial appearance, while additional images lead to slightly refined segmentations. Given 64 training images, the 25–part model produces a more detailed segmentation which intuitively aligns with natural facial features. However, with only 4 training images the 25–part model is very irregular, and provides segmentations which seem worse than the corresponding 10–part model. In Sec. 5.4.3, we show that object recognition performance also suffers when there is too little training data to reliably estimate the chosen number of parts.

■ 5.3.4 Gibbs Sampling with Reference Transformations

Returning to the full, fixed–order object model of Fig. 5.4, we now develop learning algorithms which account for reference transformations. In this section, we describe a generalized Gibbs sampler which alternates between sampling reference transformations and assignments of features to parts. Sec. 5.3.5 then develops an alternative, variational approximation based on incremental EM updates.

Part Assignment Resampling

Given a fixed set of reference transformations $\boldsymbol{\rho} = \{\rho_j\}_{j=1}^J$, the posterior distribution needed to resample part assignment z_{ji} factors as follows:

$$p(z_{ji} \mid \mathbf{z}_{\setminus ji}, \mathbf{w}, \mathbf{v}, \mathbf{o}, \boldsymbol{\rho}) \;\propto\; p(z_{ji} \mid \mathbf{z}_{\setminus ji}, o_j)\, p(w_{ji} \mid \mathbf{z}, \mathbf{w}_{\setminus ji})\, p(v_{ji} \mid \mathbf{z}, \mathbf{v}_{\setminus ji}, \boldsymbol{\rho}) \tag{5.21}$$

While the first two terms are unchanged from eqs. (5.17, 5.18), the predictive position likelihood now depends on the current transformations $\boldsymbol{\rho}$ for all training images. Expanding this term, which marginalizes latent position parameters (μ_k, Λ_k), we have

$$\begin{aligned} p(v_{ji} \mid z_{ji} = k, \mathbf{z}_{\setminus ji}, \mathbf{v}_{\setminus ji}, \boldsymbol{\rho}) &= \iint H_v(\mu_k, \Lambda_k) \prod_{j'i' \mid z_{j'i'} = k} \mathcal{N}\big(v_{j'i'}; \mu_k + \rho_{j'}, \Lambda_k\big)\, d\mu_k\, d\Lambda_k \\ &= \iint H_v(\mu_k, \Lambda_k) \prod_{j'i' \mid z_{j'i'} = k} \mathcal{N}\big(v_{j'i'} - \rho_{j'}; \mu_k, \Lambda_k\big)\, d\mu_k\, d\Lambda_k \end{aligned} \tag{5.22}$$

The equivalence of eq. (5.22) follows from eq. (5.3). Because $H_v(\mu_k, \Lambda_k)$ is normal–inverse–Wishart, Sec. 2.1.4 shows that this likelihood is approximately Gaussian, with mean $\hat{\mu}_k$ and covariance $\hat{\Lambda}_k$ computed from eq. (2.64) based on data *transformed* by the current reference positions:

$$\begin{aligned} p(v_{ji} \mid z_{ji} = k, \mathbf{z}_{\setminus ji}, \mathbf{v}_{\setminus ji}, \boldsymbol{\rho}) &= p\big(v_{ji} - \rho_j \mid \big\{(v_{j'i'} - \rho_{j'}) \mid z_{j'i'} = k, (j', i') \neq (j, i)\big\}\big) \\ &\approx \mathcal{N}(v_{ji} - \rho_j; \hat{\mu}_k, \hat{\Lambda}_k) \end{aligned} \tag{5.23}$$

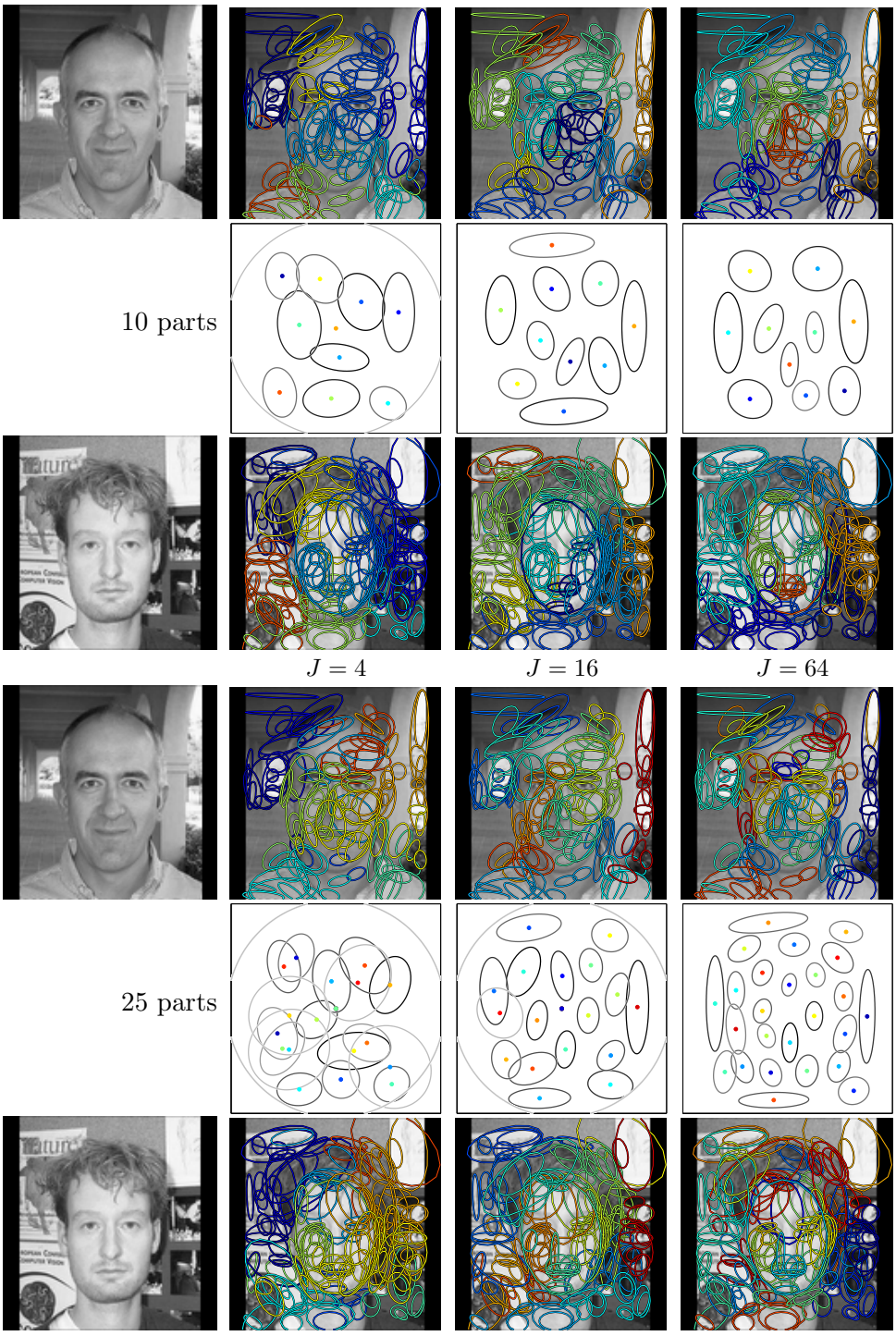


Figure 5.6. Visualization of single category, fixed–order facial appearance models (see Fig. 5.4), with $K = 10$ (top) and $K = 25$ (bottom) parts. We plot the Gaussian position distributions $\mathcal{N}(\mu_k, \Lambda_k)$ for each part (center rows), with intensity proportional to their posterior probability π_k. For each model, we also show two color–coded segmentations of image features according to their most likely corresponding part. Columns show models learned from $J = 4$, 16, or 64 training images. Note that additional training images allow more robust learning of richer models with additional parts.

Thus, if each part caches feature position sums and outer products relative to the current reference transformations $\boldsymbol{\rho}$, sampling updates of assignment z_{ji} still require only $\mathcal{O}(K)$ operations. Note that a similar approach could be applied to any family of reference transformations $\tau(\theta_k; \rho_j)$ for which a complementary data transformation $\tilde{\tau}(v_{ji}; \rho_j)$, as in eq. (5.1), is available.

Reference Transformation Resampling

As demonstrated for standard mixture models in Sec. 2.4.4, Gibbs samplers which integrate over latent parameters more rapidly mix to the true posterior distribution. While we have shown that such Rao–Blackwellization leads to tractable part assignment updates, it complicates transformation resampling. To handle this issue, we employ an *auxiliary variable* method [9, 222], as described in Sec. 2.4.3. Conditioned on all current assignments $\mathbf{z}$, we draw a single sample from the posterior distribution of each part's position parameters:

$$(\hat{\mu}_k, \hat{\Lambda}_k) \sim p(\mu_k, \Lambda_k \mid \{(v_{ji} - \rho_j) \mid z_{ji} = k\}) \qquad k = 1, \dots, K \tag{5.24}$$

Sampling from these normal–inverse–Wishart distributions (see Sec. 2.1.4) is straightforward [107]. For datasets with many training images, these densities are typically tightly concentrated, and the auxiliary samples of eq. (5.24) are closely approximated by the corresponding posterior modes.

Conditioned on part parameters sampled as in eq. (5.24), the graph of Fig. 5.4 shows that the posterior reference transformation distribution factors as follows:

$$p\big(\rho_j \mid \boldsymbol{\rho}_{\setminus j}, \mathbf{o}, \mathbf{z}, \mathbf{v}, \{\hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K\big) \propto p\big(\rho_j \mid \boldsymbol{\rho}_{\setminus j}, \mathbf{o}\big) \prod_{k=1}^K \prod_{i \mid z_{ji}=k} \mathcal{N}(v_{ji} - \rho_j; \hat{\mu}_k, \hat{\Lambda}_k) \tag{5.25}$$

$$p\big(\rho_j \mid \boldsymbol{\rho}_{\setminus j}, \mathbf{o}\big) = p\big(\rho_j \mid \{\rho_{j'} \mid o_{j'} = o_j\}\big) \approx \mathcal{N}(\rho_j; \hat{\zeta}_{o_j}, \hat{\Upsilon}_{o_j}) \tag{5.26}$$

The Gaussian distribution of eq. (5.26) approximates the Student-t predictive likelihood implied by the current transforms for other images of object o_j (see eq. (2.64) of Sec. 2.1.4). Examining eq. (5.25), we see that the reference transformations for other images act as a Gaussian prior for ρ_j, while the feature assignments in image j effectively provide Gaussian noise–corrupted observations. The posterior transformation distribution is thus also Gaussian, with mean and covariance given by the following information form (see eqs. (2.56, 2.57)):

$$p\big(\rho_j \mid \boldsymbol{\rho}_{\setminus j}, \mathbf{o}, \mathbf{z}, \mathbf{v}, \{\hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K\big) \approx \mathcal{N}(\rho_j; \chi_j, \Xi_j) \tag{5.27}$$

$$\Xi_j^{-1} = \hat{\Upsilon}_{o_j}^{-1} + \sum_{k=1}^K \sum_{i \mid z_{ji}=k} \hat{\Lambda}_k^{-1} \qquad \Xi_j^{-1}\chi_j = \hat{\Upsilon}_{o_j}^{-1}\hat{\zeta}_{o_j} + \sum_{k=1}^K \sum_{i \mid z_{ji}=k} \hat{\Lambda}_k^{-1}(v_{ji} - \hat{\mu}_k)$$

Note that all features assigned to part k induce the same observation covariance $\hat{\Lambda}_k$, so that Ξ_j^{-1} adds one multiple of $\hat{\Lambda}_k^{-1}$ for each such feature. The only approximation

underlying eq. (5.27) is the replacement of a Student-t prior by a moment–matched Gaussian (eq. (5.26)); given this, the posterior transformation distribution is available in closed form. After resampling ρ_j according to eq. (5.27), the part parameter auxiliary variables $\{\hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K$ are discarded. This auxiliary variable approach allows us to consistently update reference transformations, while retaining the advantages of Rao–Blackwellization when sampling assignments of features to parts.

In Alg. 5.2, we summarize a Rao–Blackwellized Gibbs sampler based on the preceding analysis. In our experiments, we have found the auxiliary variables sampled in eq. (5.24) to be well approximated by the mode of the corresponding normal–inverse–Wishart posterior (see Alg. 5.2, step 4(b)). Recall that parts model feature positions relative to reference transformations. Any cached statistics based on those features must then be updated whenever a reference transformation is resampled. Step 5 of Alg. 5.2 provides a simple form for such updates which is easily generalized to other families of transformations. In practice, the computation time of this sampler is dominated by the resampling of part assignments (Alg. 5.2, step 2), and accounting for reference transformations takes little time beyond that already required by Alg. 5.1.

■ 5.3.5 Inferring Likely Reference Transformations

In this section, we describe an alternative method for learning shared parts from unaligned datasets. Rather than sampling reference transformations as in Alg. 5.2, we use a variational approximation to integrate over these transformations, and only explicitly sample the assignments $\mathbf{z}$ of features to parts. An incremental form of the EM algorithm [225] then efficiently updates variational parameters as features are reassigned.

As before, consider the graphical model of Fig. 5.4. The posterior distribution of z_{ji}, given the observed image features and other part assignments $\mathbf{z}_{\setminus ji}$, factors as follows:

$$p(z_{ji} \mid \mathbf{z}_{\setminus ji}, \mathbf{w}, \mathbf{v}, \mathbf{o}) \;\propto\; p(z_{ji} \mid \mathbf{z}_{\setminus ji}, o_j)\, p(w_{ji} \mid \mathbf{z}, \mathbf{w}_{\setminus ji})\, p(v_{ji} \mid \mathbf{z}, \mathbf{v}_{\setminus ji}, \mathbf{o}) \tag{5.28}$$

The first two terms are unchanged from eqs. (5.17, 5.18), but uncertainty in the position parameters $\{\mu_k, \Lambda_k\}_{k=1}^K$ causes the predictive position likelihood to depend on the latent reference positions, and hence object labels, of all training images. In Sec. 5.3.4, we simplified this term by conditioning on the reference transformation ρ_j. The likelihood of eq. (5.28) instead marginalizes over transformations. Letting $\boldsymbol{\theta} = \{\mu_k, \Lambda_k\}_{k=1}^K$ denote part position parameters and $\boldsymbol{\varphi} = \{\zeta_\ell, \Upsilon_\ell\}_{\ell=1}^L$ transformation parameters, we have

$$p(v_{ji} \mid \mathbf{z}, \mathbf{v}_{\setminus ji}, \mathbf{o}) = \iint \left[\int p(v_{ji} \mid z_{ji}, \rho_j, \boldsymbol{\theta})\, p(\rho_j \mid \mathbf{z}_{j\setminus i}, \mathbf{v}_{j\setminus i}, o_j, \boldsymbol{\varphi})\; d\rho_j \right] \cdots \times p(\boldsymbol{\theta}, \boldsymbol{\varphi} \mid \mathbf{z}_{\setminus ji}, \mathbf{v}_{\setminus ji}, \mathbf{o})\; d\boldsymbol{\theta}\, d\boldsymbol{\varphi} \tag{5.29}$$

Here, $\mathbf{v}_{j\setminus i}$ denotes the set $\mathbf{v}_j$ of position features in image j, excluding v_{ji}. If this marginalized likelihood could be evaluated, the Rao–Blackwell Theorem (see Sec. 2.4.4) guarantees that the resulting sampler would have lower variance than that of Alg. 5.2.

Given a previous reference transformation $\rho_j^{(t-1)}$, and part assignments $\mathbf{z}_j^{(t-1)}$ for the N_j features in an image depicting object category $o_j = \ell$, resample these variables as follows:

1. Sample a random permutation $\tau(\cdot)$ of the integers $\{1, \ldots, N_j\}$.
2. Set $\mathbf{z}_j = \mathbf{z}_j^{(t-1)}$. For each $i \in \{\tau(1), \ldots, \tau(N_j)\}$, sequentially resample z_{ji} as follows:
   (a) Remove feature (w_{ji}, v_{ji}) from the cached statistics for its current part $k = z_{ji}$:
   $$\begin{aligned} N_{\ell k} &\leftarrow N_{\ell k} - 1 \\ C_{kw} &\leftarrow C_{kw} - 1 \qquad\qquad w = w_{ji} \\ (\hat{\mu}_k, \hat{\Lambda}_k) &\leftarrow (\hat{\mu}_k, \hat{\Lambda}_k) \ominus (v_{ji} - \rho_j^{(t-1)}) \end{aligned}$$
   (b) For each of the K parts, determine the predictive likelihood
   $$f_k(w_{ji} = w, v_{ji}) = \left(\frac{C_{kw} + \lambda/W}{\sum_{w'} C_{kw'} + \lambda} \right) \cdot \mathcal{N}(v_{ji} - \rho_j^{(t-1)}; \hat{\mu}_k, \hat{\Lambda}_k)$$
   (c) Sample a new part assignment z_{ji} from the following multinomial distribution:
   $$z_{ji} \sim \frac{1}{Z_i} \sum_{k=1}^{K} (N_{\ell k} + \alpha/K) f_k(w_{ji}, v_{ji}) \delta(z_{ji}, k) \qquad Z_i = \sum_{k=1}^{K} (N_{\ell k} + \alpha/K) f_k(w_{ji}, v_{ji})$$
   (d) Add feature (w_{ji}, v_{ji}) to the cached statistics for its new part $k = z_{ji}$:
   $$\begin{aligned} N_{\ell k} &\leftarrow N_{\ell k} + 1 \\ C_{kw} &\leftarrow C_{kw} + 1 \qquad\qquad w = w_{ji} \\ (\hat{\mu}_k, \hat{\Lambda}_k) &\leftarrow (\hat{\mu}_k, \hat{\Lambda}_k) \oplus (v_{ji} - \rho_j^{(t-1)}) \end{aligned}$$
3. Set $\mathbf{z}_j^{(t)} = \mathbf{z}_j$. Optionally, part weights $\boldsymbol{\pi}_\ell^{(t)}$ and parameters $\{\eta_k^{(t)}, \mu_k^{(t)}, \Lambda_k^{(t)}\}_{k=1}^K$ may be sampled as in step 3 of Alg. 5.1.
4. Sample a new reference transformation $\rho_j^{(t)}$ as follows:
   (a) Remove $\rho_j^{(t-1)}$ from cached transformation statistics for object ℓ:
   $$(\hat{\zeta}_\ell, \hat{\Upsilon}_\ell) \leftarrow (\hat{\zeta}_\ell, \hat{\Upsilon}_\ell) \ominus \rho_j^{(t-1)}$$
   (b) Sample $\rho_j^{(t)} \sim \mathcal{N}(\chi_j, \Xi_j)$, a posterior distribution determined via eq. (5.27) from the prior $\mathcal{N}(\rho_j; \hat{\zeta}_\ell, \hat{\Upsilon}_\ell)$, cached part statistics $\{\hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K$, and feature positions $\mathbf{v}_j$.
   (c) Add $\rho_j^{(t)}$ to cached transformation statistics for object ℓ:
   $$(\hat{\zeta}_\ell, \hat{\Upsilon}_\ell) \leftarrow (\hat{\zeta}_\ell, \hat{\Upsilon}_\ell) \oplus \rho_j^{(t)}$$
5. For each $i \in \{1, \ldots, N_j\}$, update cached statistics for part $k = z_{ji}$ as follows:
   $$\begin{aligned} (\hat{\mu}_k, \hat{\Lambda}_k) &\leftarrow (\hat{\mu}_k, \hat{\Lambda}_k) \ominus (v_{ji} - \rho_j^{(t-1)}) \\ (\hat{\mu}_k, \hat{\Lambda}_k) &\leftarrow (\hat{\mu}_k, \hat{\Lambda}_k) \oplus (v_{ji} - \rho_j^{(t)}) \end{aligned}$$

Algorithm 5.2. Rao–Blackwellized Gibbs sampler for the K part, fixed–order object model of Fig. 5.4, including reference transformations. We illustrate the sequential resampling of all feature assignments $\mathbf{z}_j$ in the j^{th} training image, as well as that image's coordinate frame ρ_j. A full iteration of the Gibbs sampler applies these updates to all images in random order. For efficiency, we cache and recursively update statistics $\{\hat{\zeta}_\ell, \hat{\Upsilon}_\ell)\}_{\ell=1}^L$ of each object's reference transformations, counts $N_{\ell k}$ of the features assigned to each part, and statistics $\{C_{kw}, \hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K$ of those features' appearance and position. The final step ensures consistency of these part statistics following reference transformation updates.

Unfortunately, however, due to posterior dependency between the part parameters $\boldsymbol{\theta}$ and transformation parameters $\boldsymbol{\varphi}$, eq. (5.29) lacks a closed analytic form.

Note that the likelihood of eq. (5.29) depends on the posterior distribution of the parameters given *fixed* assignments $\mathbf{z}_{\setminus ji}$ of all other features to parts. This distribution combines information from many training images, and is often tightly concentrated. We therefore propose to approximate this likelihood via the parameters' posterior mode:

$$(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\varphi}}) = \arg\max_{\boldsymbol{\theta},\boldsymbol{\varphi}} p(\boldsymbol{\theta}, \boldsymbol{\varphi} \mid \mathbf{z}_{\setminus ji}, \mathbf{v}_{\setminus ji}, \mathbf{o}) \tag{5.30}$$

Given these parameters, the predictive likelihood of eq. (5.29) reduces to an easily evaluated Gaussian integral. Direct optimization of eq. (5.30) is complicated by the latent, unobserved reference transformations. We therefore use a variant of the EM algorithm [107, 161] (see Sec. 2.3.3), in which the E–step determines Gaussian posteriors for reference transformations, and the M–step provides corresponding parameter estimates. Using incremental updates [225], we may efficiently track transformation statistics as the Gibbs sampler reassigns features to new parts.

Expectation Step

In the E–step, we assume fixed values for the transformation parameters $\widehat{\boldsymbol{\varphi}} = \{\hat{\zeta}_\ell, \hat{\Upsilon}_\ell\}_{\ell=1}^L$ and part position parameters $\widehat{\boldsymbol{\theta}} = \{\hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K$, and determine posterior distributions for the reference transformations $\boldsymbol{\rho} = \{\rho_j\}_{j=1}^J$. From the graph of Fig. 5.4, these distributions take the following form:

$$p(\rho_j \mid o_j = \ell, \mathbf{z}_j, \mathbf{v}_j, \widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\varphi}}) \propto \mathcal{N}(\rho_j; \hat{\zeta}_\ell, \hat{\Upsilon}_\ell) \prod_{k=1}^K \prod_{i|z_{ji}=k} \mathcal{N}(v_{ji} - \rho_j; \hat{\mu}_k, \hat{\Lambda}_k) \tag{5.31}$$

This expression is identical to that arising in the auxiliary variable Gibbs sampler of Sec. 5.3.4 (see eq. (5.25)). Reference transformation ρ_j thus has a Gaussian posterior distribution $\mathcal{N}(\chi_j, \Xi_j)$, with mean and covariance as in eq. (5.27). While Alg. 5.2 uses this posterior to sample a new transformation, the variational approach of this section instead estimates parameters analytically in the subsequent M–step.

Maximization Step

In the M–step, we use Gaussian reference transformation distributions $\rho_j \sim \mathcal{N}(\chi_j, \Xi_j)$ from the previous E–step to lower bound the posterior distribution of eq. (5.30). Let $\mathcal{NW}(\kappa_o, \vartheta_o, \nu_o, \Delta_o)$ denote the hyperparameters of the normal–inverse–Wishart prior R on transformation parameters, as defined in Sec. 2.1.4. Constructing a likelihood bound as in eq. (2.134) and taking derivatives, we find that the maximizing transformation

parameters equal

$$\hat{\zeta}_\ell = \frac{1}{\kappa_o + J_\ell}\left(\kappa_o \vartheta_o + \sum_{j|o_j=\ell} \chi_j\right) \tag{5.32}$$

$$\hat{\Upsilon}_\ell = \frac{(\kappa_o + J_\ell + 1)}{(\kappa_o + J_\ell)(\nu_o + J_\ell + d + 1)}\left(\nu_o \Delta_o + \sum_{j|o_j=\ell} \Xi_j + (\chi_j - \hat{\zeta}_\ell)(\chi_j - \hat{\zeta}_\ell)^T\right) \tag{5.33}$$

Here, J_ℓ is the number of training images of object ℓ, and $d = 2$ is the dimension of the reference transformation ρ_j. Intuitively, eq. (5.32) sets the transformation mean $\hat{\zeta}_\ell$ to a regularized average of the current transformations χ_j in images of object ℓ. Similarly, eq. (5.33) combines outer products of transformations $(\chi_j - \hat{\zeta}_\ell)$ with their uncertainties Ξ_j to determine $\hat{\Upsilon}_\ell$. The factor of $(d+1)$ in the denominator of eq. (5.33) arises from the skew inherent in inverse–Wishart distributions (see eq. (2.60)).

Let $\mathcal{NW}(\kappa_v, \vartheta_v, \nu_v, \Delta_v)$ denote the hyperparameters of the normal–inverse–Wishart prior H_v on part position parameters. Via a similar derivation, the likelihood bound is maximized when these parameters are set as follows:

$$\hat{\mu}_k = \frac{1}{\kappa_v + N_k}\left(\kappa_v \vartheta_v + \sum_{j=1}^{J} \sum_{i|z_{ji}=k} (v_{ji} - \chi_j)\right) \tag{5.34}$$

$$\hat{\Lambda}_k = \frac{(\kappa_v + N_k + 1)}{(\kappa_v + N_k)(\nu_v + N_k + d + 1)} \cdots \times \left(\nu_v \Delta_v + \sum_{j=1}^{J} \sum_{i|z_{ji}=k} \Xi_j + (v_{ji} - \chi_j - \hat{\mu}_k)(v_{ji} - \chi_j - \hat{\mu}_k)^T\right) \tag{5.35}$$

In this expression, N_k is the total number of features which $\mathbf{z}_{\setminus ji}$ currently assigns to part k. As in the Gibbs sampler of Alg. 5.2, part statistics depend on the relative displacements $(v_{ji} - \chi_j)$ of image features from current reference transformation estimates.

Likelihood Evaluation and Incremental EM Updates

Given fixed assignments $\mathbf{z}_{\setminus ji}$ of features to parts, the preceding EM updates converge to a local maximum of the posterior distribution of eq. (5.30). Conditioned on the parameters $\widehat{\boldsymbol{\varphi}} = \{\hat{\zeta}_\ell, \hat{\Upsilon}_\ell\}_{\ell=1}^L$ and $\widehat{\boldsymbol{\theta}} = \{\hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K$ computed in the final M–step, the reference transformation follows the Gaussian posterior $\rho_j \sim \mathcal{N}(\chi_j, \Xi_j)$ determined in the E–step via eq. (5.27). Integrating over ρ_j, the feature position likelihood of eq. (5.29) then has the following closed–form approximation:

$$p(v_{ji} \mid z_{ji} = k, \mathbf{z}_{\setminus ji}, \mathbf{v}_{\setminus ji}, \mathbf{o}) \approx \mathcal{N}(v_{ji}; \chi_j + \hat{\mu}_k, \Xi_j + \hat{\Lambda}_k) \tag{5.36}$$

This approximation will be accurate when the posterior distribution of eq. (5.30) is concentrated around a single mode. Empirically, this is usually true given "consistent" feature assignments $\mathbf{z}_{\setminus ji}$ which have high joint posterior probability.

Given a previous transformation posterior $\mathcal{N}(\rho_j; \chi_j, \Xi_j)$, and part assignments $\mathbf{z}_j^{(t-1)}$ for the N_j features in an image of object category $o_j = \ell$, sequentially resample assignments as follows:

1. Sample a random permutation $\tau(\cdot)$ of the integers $\{1, \dots, N_j\}$.
2. Set $\mathbf{z}_j = \mathbf{z}_j^{(t-1)}$. For each $i \in \{\tau(1), \dots, \tau(N_j)\}$, sequentially resample z_{ji} as follows:
   (a) Remove feature (w_{ji}, v_{ji}) from the cached statistics for its current part $k = z_{ji}$:
   $$N_{\ell k} \leftarrow N_{\ell k} - 1$$
   $$C_{kw} \leftarrow C_{kw} - 1 \qquad w = w_{ji}$$
   Update $(\hat{\mu}_k, \hat{\Lambda}_k)$ by subtracting $(v_{ji} - \chi_j)$ from mean statistics (eq. (5.34)), and $\Xi_j + (v_{ji} - \chi_j - \hat{\mu}_k)(v_{ji} - \chi_j - \hat{\mu}_k)^T$ from covariance statistics (eq. (5.35)).
   (b) *E–Step:* Update the reference transformation's Gaussian posterior distribution $\mathcal{N}(\rho_j; \chi_j, \Xi_j)$ using eq. (5.27), excluding the currently unassigned feature v_{ji}.
   (c) *M–Step:* Compute new transformation parameters $(\hat{\zeta}_\ell, \hat{\Upsilon}_\ell)$ using eqs. (5.32, 5.33), and new part position parameters $\{\hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K$ using eqs. (5.34, 5.35).
   (d) For each of the K parts, determine the predictive likelihood
   $$f_k(w_{ji} = w, v_{ji}) = \left(\frac{C_{kw} + \lambda/W}{\sum_{w'} C_{kw'} + \lambda} \right) \cdot \mathcal{N}(v_{ji}; \chi_j + \hat{\mu}_k, \Xi_j + \hat{\Lambda}_k)$$
   (e) Sample a new part assignment z_{ji} from the following multinomial distribution:
   $$z_{ji} \sim \frac{1}{Z_i} \sum_{k=1}^K (N_{\ell k} + \alpha/K) f_k(w_{ji}, v_{ji}) \delta(z_{ji}, k) \qquad Z_i = \sum_{k=1}^K (N_{\ell k} + \alpha/K) f_k(w_{ji}, v_{ji})$$
   (f) Add feature (w_{ji}, v_{ji}) to the cached statistics for its new part $k = z_{ji}$:
   $$N_{\ell k} \leftarrow N_{\ell k} + 1$$
   $$C_{kw} \leftarrow C_{kw} + 1 \qquad w = w_{ji}$$
   Update $(\hat{\mu}_k, \hat{\Lambda}_k)$ by adding $(v_{ji} - \chi_j)$ to mean statistics (eq. (5.34)), and $\Xi_j + (v_{ji} - \chi_j - \hat{\mu}_k)(v_{ji} - \chi_j - \hat{\mu}_k)^T$ to covariance statistics (eq. (5.35)).
3. Set $\mathbf{z}_j^{(t)} = \mathbf{z}_j$. Optionally, part weights $\boldsymbol{\pi}_\ell^{(t)}$ and parameters $\{\eta_k^{(t)}, \mu_k^{(t)}, \Lambda_k^{(t)}\}_{k=1}^K$ may be sampled as in step 3 of Alg. 5.1.

Algorithm 5.3. Rao–Blackwellized Gibbs sampler for the K part, fixed–order object model of Fig. 5.4, using a variational approximation to marginalize reference transformations. We illustrate the sequential resampling of all feature assignments $\mathbf{z}_j$ in the j^{th} training image, based on incremental EM updates of the model's position parameters. A full iteration of the Gibbs sampler applies these updates to all images in random order. For efficiency, we cache and recursively update statistics $\{\hat{\zeta}_\ell, \hat{\Upsilon}_\ell)\}_{\ell=1}^L$ of each object's reference transformations, counts $N_{\ell k}$ of the features assigned to each part, and statistics $\{C_{kw}, \hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K$ of those features' appearance and position.

To apply this analysis, we extend the Rao–Blackwellized Gibbs sampler of Alg. 5.1 to recompute the parameters' posterior mode following each feature reassignment. As summarized in Alg. 5.3, we initialize these EM updates with parameters induced by previous assignments. The newly updated position parameters then determine likelihoods of features v_{ji} as in eq. (5.36), allowing a new assignment z_{ji} to be sampled. Because

the posterior mode is not dramatically changed by the reassignment of one feature, a single EM iteration per sample is usually sufficient for accurate mode tracking.

Note that feature reassignments in one image alter the expected reference transformations in other images via shared part parameters. Direct implementation of the preceding EM updates thus requires $\mathcal{O}(JK)$ operations per iteration. Alg. 5.3 reduces the cost of each iteration to $\mathcal{O}(K)$ using incremental EM updates [225]. In particular, when sampling a feature assignment for image j, we fix the E–step's posterior transformation distributions (eq. (5.31)) for all other images. By caching statistics of other reference transformation estimates, the M–step may also be performed efficiently (see [225] for further discussion). Although we no longer find the exact posterior mode, the dependence between $\mathbf{z}_j$ and transformations $\boldsymbol{\rho}_{\setminus j}$ in other training images is very weak, so this approximation is extremely accurate. Empirically, incremental updates produce dramatic computational gains with comparable sampling accuracy.

■ 5.3.6 Likelihoods for Object Detection and Recognition

To use our fixed–order object appearance model for detection or recognition, we must compute the likelihood that a test image j, with N_j features $(\mathbf{w}_j, \mathbf{v}_j)$, is generated by each candidate object category o_j. Because each image's features are independently sampled from a common parameter set, we have

$$p(\mathbf{w}_j, \mathbf{v}_j \mid o_j, \mathcal{J}) = \int p(\mathbf{w}_j, \mathbf{v}_j \mid o_j, \boldsymbol{\pi}, \boldsymbol{\theta}, \boldsymbol{\varphi})\, p(\boldsymbol{\pi}, \boldsymbol{\theta}, \boldsymbol{\varphi} \mid \mathcal{J})\; d\boldsymbol{\pi} d\boldsymbol{\theta} d\boldsymbol{\varphi}$$

In this expression, $\mathcal{J}$ denotes the set of training images, $\boldsymbol{\theta} = \{\eta_k, \mu_k, \Lambda_k\}_{k=1}^K$ the part position and appearance parameters, and $\boldsymbol{\varphi} = \{\zeta_\ell, \Upsilon_\ell\}_{\ell=1}^L$ the reference transformation parameters. The sequence of part assignments produced by the Gibbs sampler provides samples $\mathbf{z}^{(a)}$ approximately distributed according to $p(\mathbf{z} \mid \mathcal{J})$. Given A such samples, we approximate the test image likelihood as

$$p(\mathbf{w}_j, \mathbf{v}_j \mid o_j, \mathcal{J}) \approx \frac{1}{A} \sum_{a=1}^{A} p(\mathbf{w}_j, \mathbf{v}_j \mid o_j, \boldsymbol{\pi}^{(a)}, \boldsymbol{\theta}^{(a)}, \boldsymbol{\varphi}^{(a)}) \tag{5.37}$$

In this expression, $(\boldsymbol{\pi}^{(a)}, \boldsymbol{\theta}^{(a)}, \boldsymbol{\varphi}^{(a)})$ denote parameters sampled from the posterior distribution induced by $\mathbf{z}^{(a)}$ (see Alg. 5.1, step 3).

When reference transformations are neglected, as in Sec. 5.3.2, image features are conditionally independent given model parameters. Test image likelihoods may then be efficiently computed as follows:

$$p(\mathbf{w}_j, \mathbf{v}_j \mid o_j = \ell, \boldsymbol{\pi}^{(a)}, \boldsymbol{\theta}^{(a)}) = \prod_{i=1}^{N_j} \sum_{k=1}^{K} \hat{\pi}_{\ell k} \hat{\eta}_k(w_{ji})\, \mathcal{N}(v_{ji}; \hat{\mu}_k, \hat{\Lambda}_k) \tag{5.38}$$

Here, $\boldsymbol{\pi}^{(a)} = \{\hat{\boldsymbol{\pi}}_\ell\}_{\ell=1}^L$ and $\boldsymbol{\theta}^{(a)} = \{\hat{\eta}_k, \hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K$ denote the parameters corresponding to $\mathbf{z}^{(a)}$. This expression calculates the likelihood of N_j features in $\mathcal{O}(N_j K)$ operations.

To account for reference transformations, we first run the Gibbs sampler of Alg. 5.2 on the test image, extracting several samples $\rho_j^{(b)} \sim p(\rho_j \mid o_j, \boldsymbol{\pi}^{(a)}, \boldsymbol{\theta}^{(a)}, \boldsymbol{\varphi}^{(a)})$. We then average the feature likelihoods implied by these B samples:

$$p(\mathbf{w}_j, \mathbf{v}_j \mid o_j = \ell, \boldsymbol{\pi}^{(a)}, \boldsymbol{\theta}^{(a)}, \boldsymbol{\varphi}^{(a)}) \approx \frac{1}{B} \sum_{b=1}^{B} \prod_{i=1}^{N_j} \sum_{k=1}^{K} \hat{\pi}_{\ell k} \hat{\eta}_k(w_{ji}) \mathcal{N}(v_{ji}; \hat{\mu}_k + \rho_j^{(b)}, \hat{\Lambda}_k) \quad (5.39)$$

By marginalizing assignments $\mathbf{z}_j$ of test features to parts, this Rao–Blackwellized estimator has lower variance than one which directly incorporates assignments $\mathbf{z}_j^{(b)}$ from the Gibbs sampler. In some cases, extracting these samples from multiple independent MCMC trials better avoids local optima and improves results [192]. Alternatively, the incremental EM updates of Alg. 5.3 can be combined with eq. (5.36) to form a different likelihood estimator for test images.

■ 5.4 Fixed–Order Models for Sixteen Object Categories

The hierarchical model developed in the preceding section describes the appearance of several object categories via a common set of shared parts. To explore the benefits of sharing parts, we consider a collection of 16 categories with noticeable visual similarities. Fig. 5.7 shows images from each category, which can be divided into three groups: seven animal faces, five animal profiles, and four objects with wheels. While training images are labeled with their corresponding category, we do *not* explicitly modify our part–based models to reflect these coarser groupings. As object recognition systems scale to applications involving hundreds or thousands of categories, the inter–category similarities exhibited by this dataset will become increasingly common.

■ 5.4.1 Visualization of Shared Parts

Given 30 training images from each of the 16 categories, we first extracted Harris–affine invariant regions [205], and maximally stable extremal regions [199], as described in Sec. 5.1.1. SIFT descriptors [188] for these interest points were then mapped to one of $W = 600$ appearance words, using a vocabulary determined as in Sec. 5.1.2. Given these features, we used the Gibbs sampler of Alg. 5.1 to fit a fixed–order object model (see Fig. 5.4) with 32 shared parts. Because our 16–category dataset contains approximately aligned images, the reference transformation updates of Algs. 5.2 or 5.3 were not needed. Chap. 6 considers generalizations of these algorithms in more detail.

For our Matlab implementation, each sampling iteration requires roughly 0.1 seconds per training image on a 3.0 GHz Intel Xeon processor. Empirically, the learning procedure is fairly robust to the hyperparameters H_w and H_v associated with part appearance and position. We chose H_v to provide a weak ($\nu = 6$ prior degrees of freedom) bias towards moderate covariances, and $H_w = \mathrm{Dir}(W/10)$ to favor sparse appearance distributions. The Dirichlet prior $\mathrm{Dir}(\alpha)$ for object–specific part distributions $\boldsymbol{\pi}_\ell$ was set via cross–validation as described in Sec. 5.4.2.

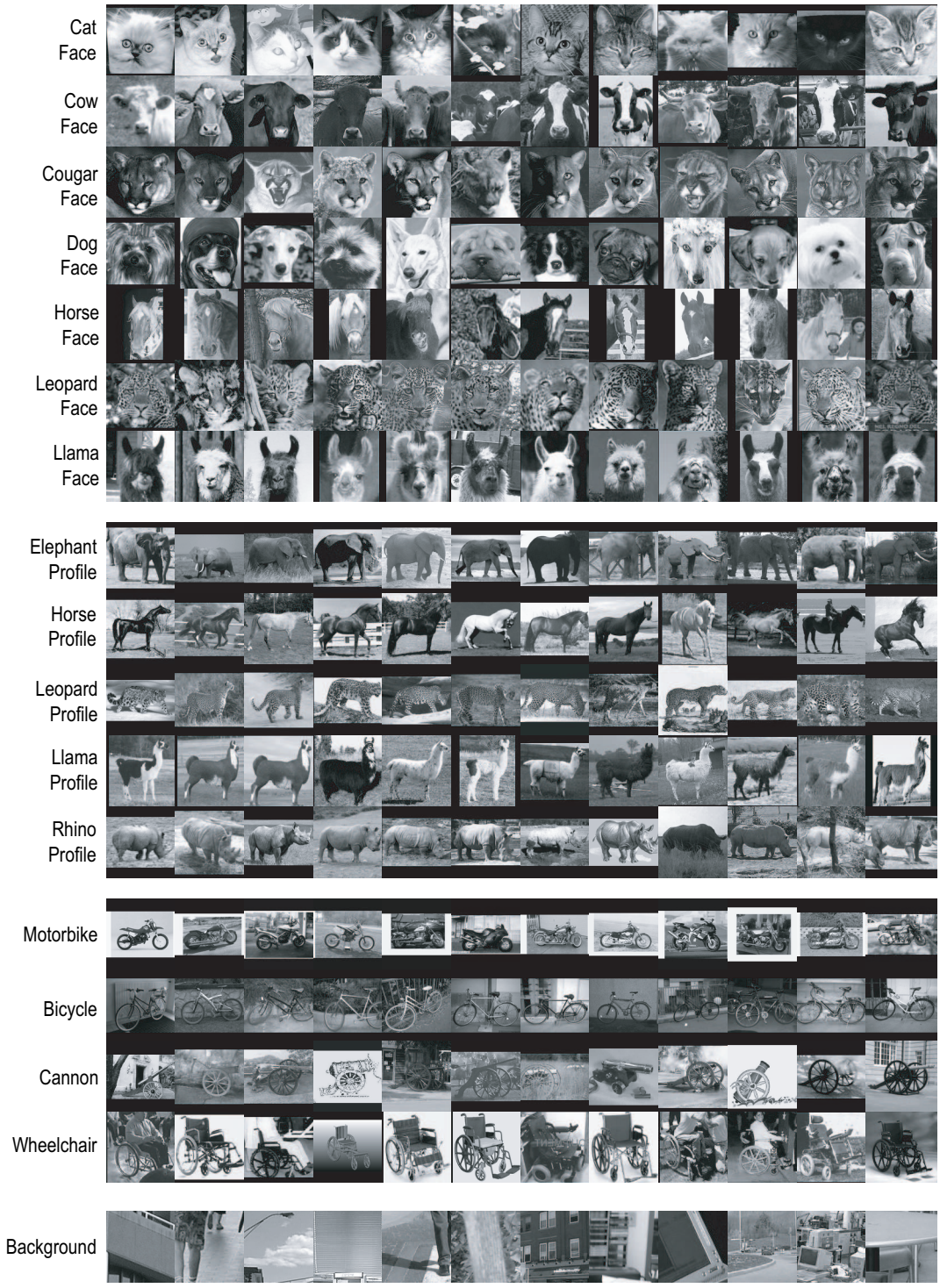


Figure 5.7. Example images from a dataset containing 16 object categories (rows). These categories combine images collected from web searches with the Caltech 101 [78], Weizmann Institute [33, 302], and MIT-CSAIL [299, 300] databases. Including a complementary background category (bottom), there are a total of 1,885 images used for training and testing, with at least 50 images for each category.

Following 500 iterations of the Gibbs sampler, we used the final assignments $\mathbf{z}$ to estimate each part's posterior distribution over feature appearance and position (Alg. 5.1, step 3). In Fig. 5.8, we visualize these distributions for seven parts. Several parts are shared among the different animal face categories, modeling common features of the mouth region, as well as corresponding vertical contours. Other parts describe the left and right wheels of the vehicle categories, along with textural features associated with animal legs. We also show one of several parts which model background clutter around image boundaries, and are widely shared among categories.

To further investigate these shared parts, we used the symmetrized KL divergence, as in [247], to compute a distance between all pairs of object–specific part distributions:

$$D(\boldsymbol{\pi}_\ell, \boldsymbol{\pi}_m) = \sum_{k=1}^{K} \pi_{\ell k} \log \frac{\pi_{\ell k}}{\pi_{mk}} + \pi_{mk} \log \frac{\pi_{mk}}{\pi_{\ell k}} \tag{5.40}$$

In Fig. 5.9, we show the two–dimensional embedding of these distances produced by metric multidimensional scaling (MDS), as well as a greedy, agglomerative clustering [257]. Interestingly, there is significant sharing of parts within each of the three coarse–level groups underlying this dataset. The animal faces are fairly well separated, while there is more overlap between the animal profiles and vehicles.

■ 5.4.2 Detection and Recognition Performance

To evaluate our fixed–order, part–based object appearance model, we consider two sets of experiments. In the detection task, we use 100 images of cluttered, natural scenes (see Fig. 5.8) to train a 30–part background appearance model. We then use probabilities computed as in Sec. 5.3.6 to classify test images as object or background. To facilitate comparisons, we also consider a recognition task in which test images are classified as either their true category, or one of the 15 other categories. For both tasks, we compare a *shared* model of all object categories to a set of 16 *unshared* models trained on individual categories (see Fig. 5.5). We also consider versions of both models which neglect the spatial location of features, as in the "bag of features" approaches [54, 266] discussed in Sec. 5.1.3. Performance curves are based on a randomly chosen training set of the specified size, and use all other images for testing.

In Fig. 5.10, we examine detection and recognition performance given training sets containing between 4 and 30 images per category. All models are allocated two parts per category (32 shared parts versus 16 unshared two–part models), and likelihoods are estimated from 10 samples extracted across 500 iterations of Alg. 5.1 (see Sec. 5.3.6). We see that shared parts lead to significant improvements in detection performance, particularly when few training examples are available. Even with 30 training images, the unshared models incorporating feature positions exhibit significant overfitting, performing worse than corresponding models based solely on feature appearance. As shown by the scatter plots of Fig. 5.10, all 16 categories benefit from the use of shared parts.

For the recognition task, the shared and unshared appearance–only models behave

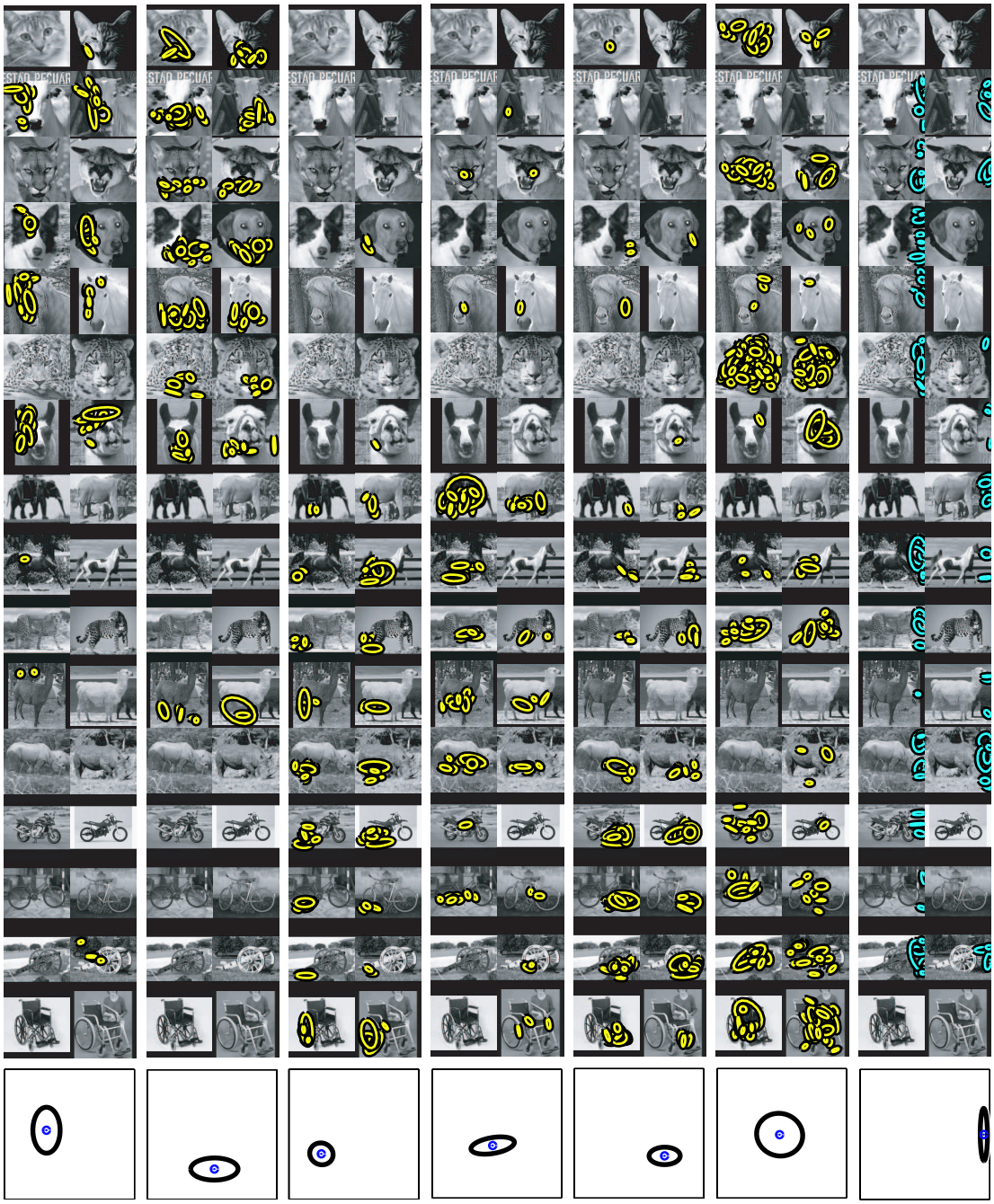


Figure 5.8. Seven of the 32 shared parts (columns) learned by a fixed–order, hierarchical model for 16 object categories (rows). Using two images from each category, we display those features with the highest posterior probability of being generated by each part. For comparison, we show six of the parts which are specialized to the fewest object categories (left, yellow), as well as one of several widely shared parts (right, cyan), which seem to model texture and background clutter. The bottom row plots the Gaussian position densities corresponding to each part. Interestingly, several parts have rough semantic interpretations, and are shared within the coarse–level object groupings underlying this dataset.

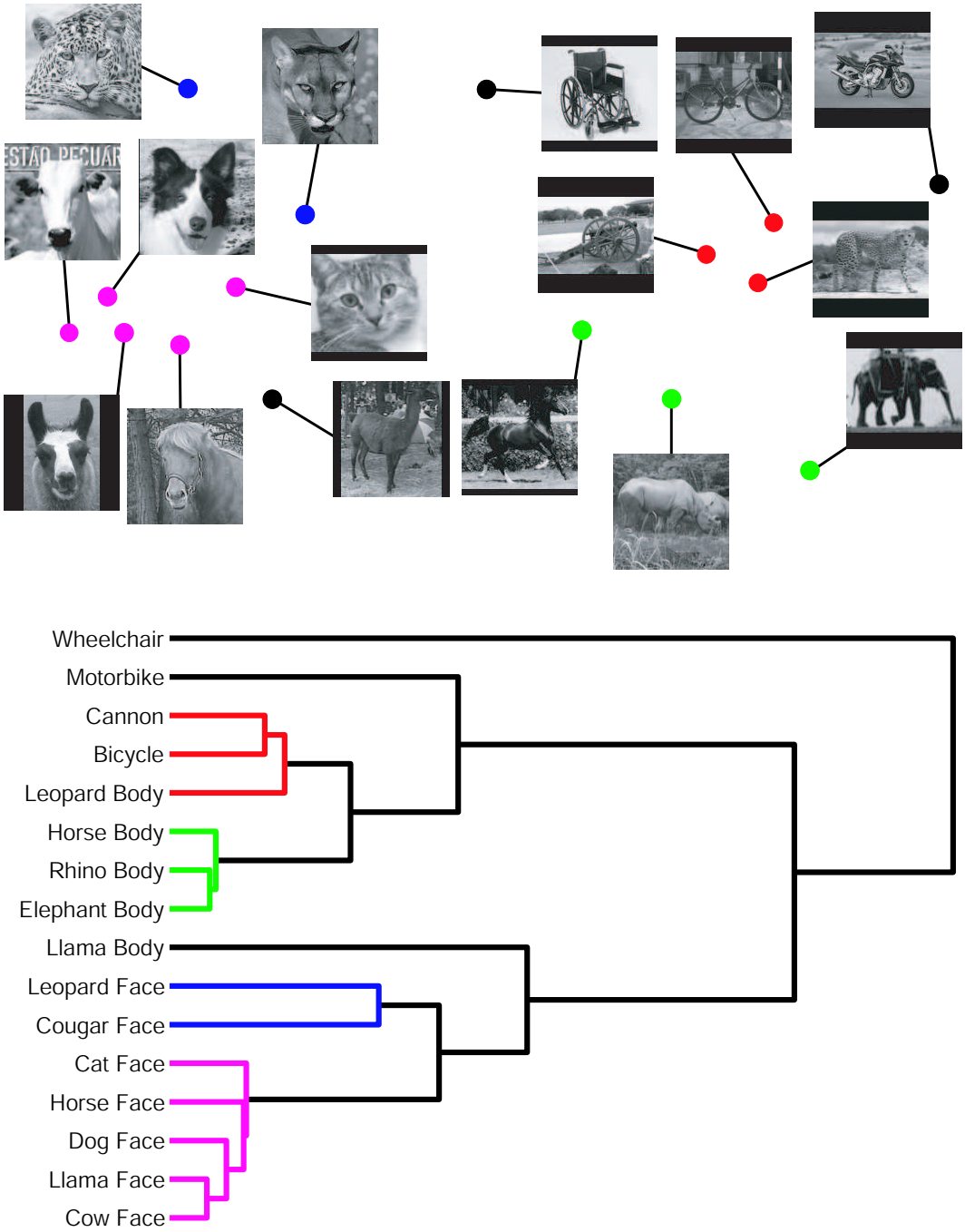


Figure 5.9. Two visualizations of learned part distributions $\boldsymbol{\pi}_\ell$ for the fixed–order, 32–part object appearance model depicted in Fig. 5.8. *Top:* Two–dimensional embedding computed by metric MDS, in which coordinates for each object category are chosen to approximate pairwise KL distances as in eq. (5.40). Animal faces are clustered on the left, vehicles in the upper right, and animal profiles in the lower right. *Bottom:* Dendrogram illustrating a greedy, hierarchical clustering, where branch lengths are proportional to inter–category distances. The four most significant clusters, which with the exception of the leopard body have intuitive interpretations, are highlighted in color.

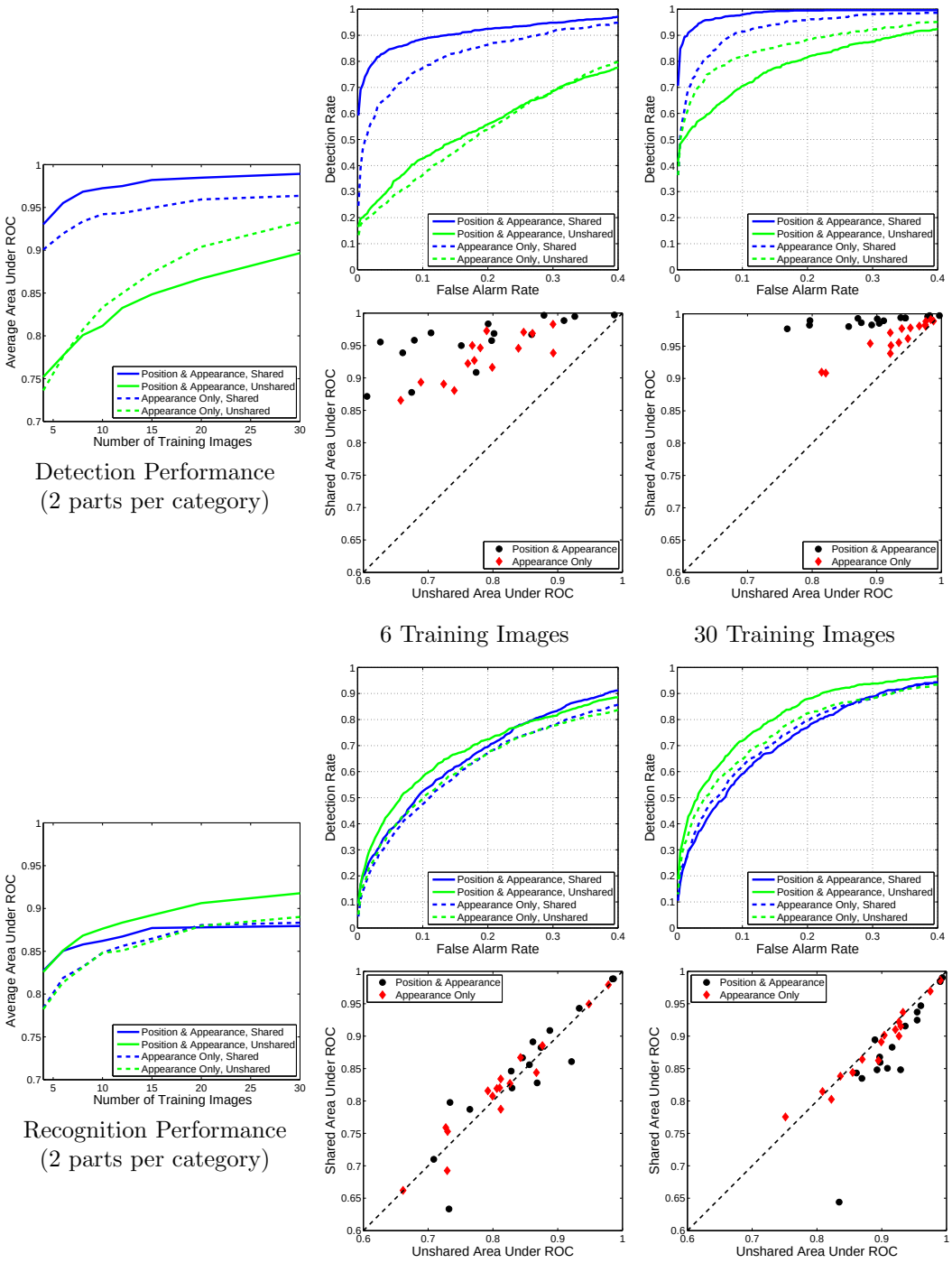


Figure 5.10. Performance of fixed–order object appearance models with two parts per category for the detection (top block) and recognition (bottom block) tasks. *Left:* Area under average ROC curves for different numbers of training images per category. *Top Right:* Average of ROC curves across all categories (6 versus 30 training images). *Bottom Right:* Scatter plot of areas under ROC curves for the shared and unshared models of individual categories (6 versus 30 training images).

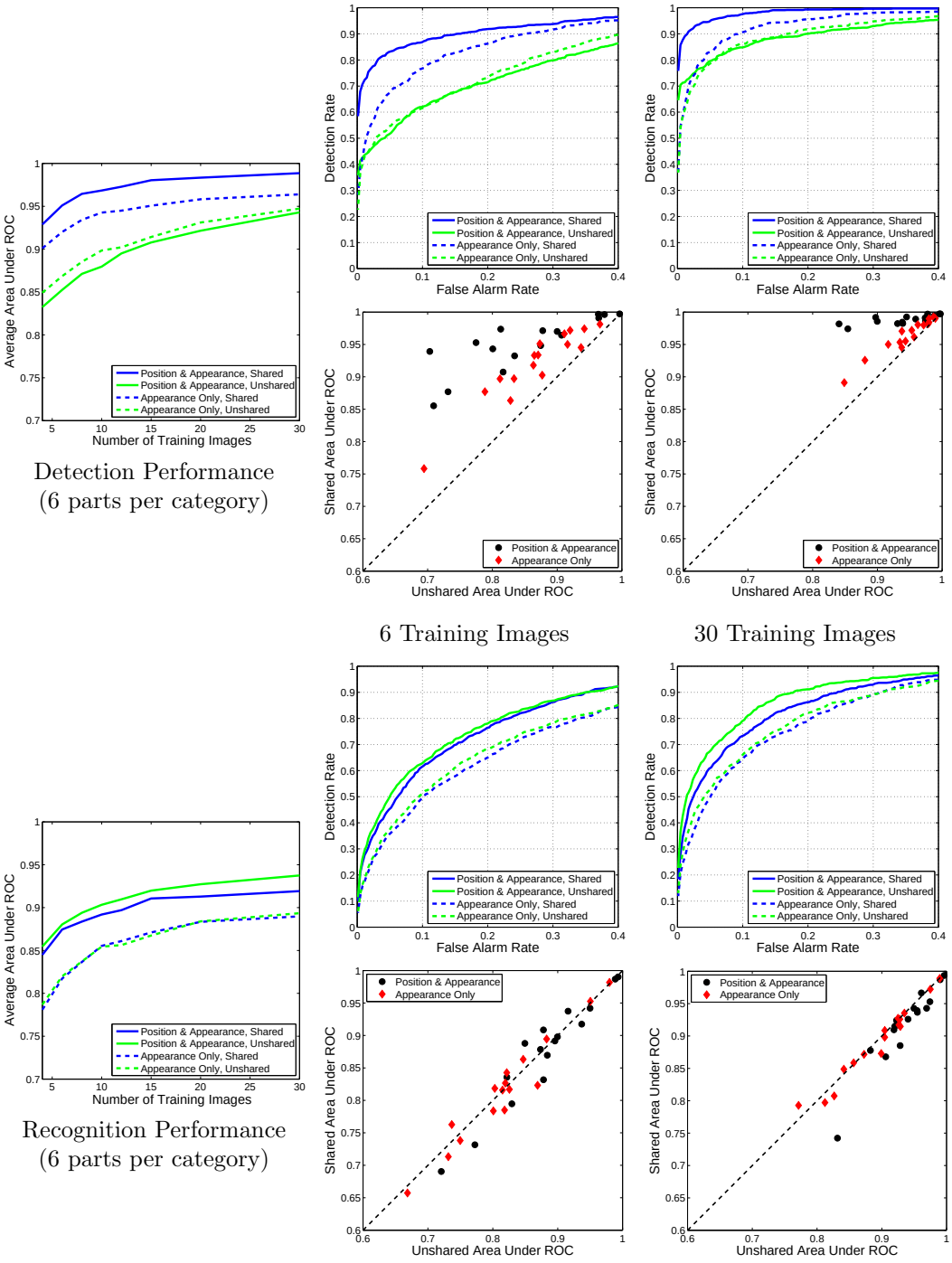


Figure 5.11. Performance of fixed–order object appearance models with six parts per category for the detection (top block) and recognition (bottom block) tasks. *Left:* Area under average ROC curves for different numbers of training images per category. *Top Right:* Average of ROC curves across all categories (6 versus 30 training images). *Bottom Right:* Scatter plot of areas under ROC curves for the shared and unshared models of individual categories (6 versus 30 training images).

similarly. However, the performance of the shared feature position model saturates given 15 or more training images, and is less effective than a comparable set of unshared models. Confusion matrices (not shown) confirm that this small performance degradation is due to errors involving pairs of object categories with similar part distributions (see Fig. 5.9). For both tasks, feature positions contain important information, and neglecting them reduces performance.

Given few training images, the Dirichlet part association prior $\boldsymbol{\pi}_\ell \sim \mathrm{Dir}(\alpha)$ affects the characteristics of the inferred object model. In particular, small α values reduce sharing and slightly increase recognition performance, while large α values increase sharing, leading to improved detection accuracy. The results in Fig. 5.10 use symmetric Dirichlet priors with precision $\alpha_0 = 10$, a value selected via cross–validation.

■ 5.4.3 Model Order Determination

The performance of the fixed–order object appearance model is appreciably affected by the chosen number of parts. In Fig. 5.11, we examine higher–order models with six parts per category (96 shared parts versus 16 unshared six–part models), trained as in Sec. 5.4.2. Using more parts significantly improves the unshared models' performance, although their detection accuracy is still inferior to the shared model. It also slightly improves the shared model's recognition accuracy. Note, however, that using six rather than two parts per category triples the computational cost of both training and testing (see discussion in Secs. 5.3.2 and 5.3.6).

To further explore these issues, we examined fixed–order object appearance models with between two and thirty parts per category (32–480 shared parts versus 16 unshared 2–30 part models). For each model order, we ran the Gibbs sampler of Alg. 5.1 for 200 iterations, and categorized test images via probabilities derived from six posterior samples. We first considered part association probabilities $\boldsymbol{\pi}_\ell$ learned using the following symmetric Dirichlet prior:

$$(\pi_{\ell 1}, \ldots, \pi_{\ell K}) \sim \mathrm{Dir}(\bar{\alpha}, \ldots, \bar{\alpha}) = \mathrm{Dir}(\bar{\alpha} K) \tag{5.41}$$

Our experiments set $\bar{\alpha} = 5$, inducing a small bias towards distributions which assign non–negligible weight to each of the K parts. Fig. 5.12 shows the average detection and recognition performance, as measured by the area under the ROC curve, for varying model orders. Even with 15 training images of each category, shared models with more than 4–6 parts per category (64–96 total parts) overfit and exhibit reduced accuracy. Similar issues arise when learning finite mixture models, where priors as in eq. (5.41) may produce inconsistent parameter estimates if K is not selected with care [149].

In some applications of the LDA model, the number of topics K is determined via cross–validation [31, 123]. This approach is also possible with the fixed–order object appearance model, but in practice requires extensive computational effort. Alternatively, model complexity can be regulated by the following modified part association prior:

$$(\pi_{\ell 1}, \ldots, \pi_{\ell K}) \sim \mathrm{Dir}\left(\frac{\alpha_0}{K}, \ldots, \frac{\alpha_0}{K}\right) = \mathrm{Dir}(\alpha_0) \tag{5.42}$$

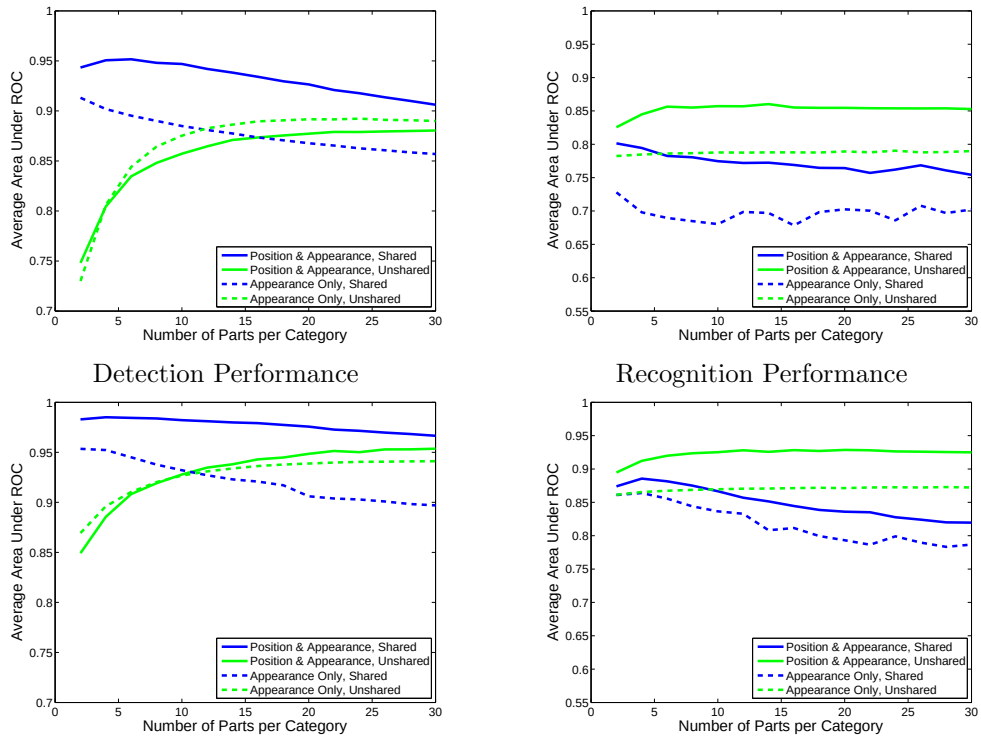


Figure 5.12. Performance of fixed–order object appearance models with varying numbers of parts K. Part association priors $\boldsymbol{\pi}_\ell \sim \mathrm{Dir}(\bar{\alpha}K)$ are biased towards uniform distributions (eq. (5.41)). We compare detection and recognition performance given 4 (top) or 15 (bottom) training images per category.

For a fixed precision α_0, this prior becomes biased towards sparse part association distributions $\boldsymbol{\pi}_\ell$ as K grows large (see Sec. 2.1.3). Fig. 5.13 illustrates the behavior of this prior when $\alpha_0 = 10$. In contrast with the previously observed overfitting, eq. (5.42) produces stable recognition results across a wider range of model orders K.

As described in Sec. 2.5.3 (see Thm. 2.5.5), predictions based on Dirichlet priors scaled as in eq. (5.42) approach a corresponding Dirichlet process as $K \to \infty$. However, if we apply this limit directly to the model of Fig. 5.4, objects asymptotically associate features with *disjoint* sets of parts, and the benefits of sharing are lost. We see the beginnings of this trend in Fig. 5.13, which shows a slow decline in detection performance as K increases. As discussed in Sec. 2.5.4, the hierarchical Dirichlet process (HDP) [289] provides an elegant solution to this problem, by using a global Dirichlet process to share clusters among related groups. The following section adapts the HDP to our object categorization task, and shows that it cleanly avoids these model selection issues.

■ 5.5 Sharing Parts with Dirichlet Processes

We now revisit the parametric, fixed–order object appearance model of Fig. 5.4. By adapting the hierarchical Dirichlet process (HDP) [289], we develop tractable, nonpara-

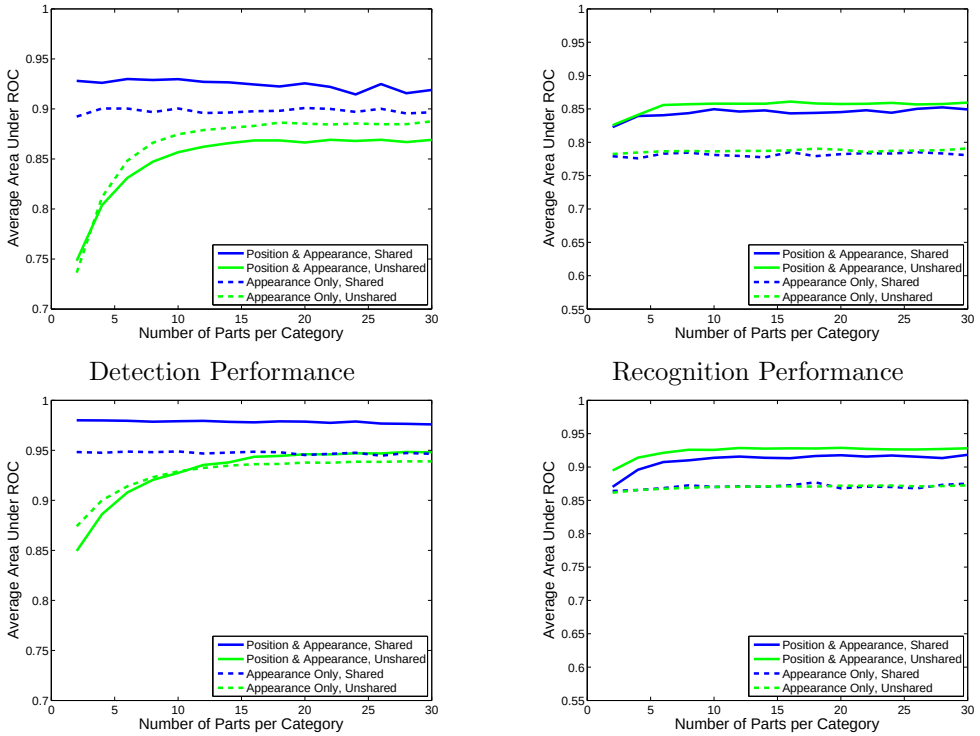


Figure 5.13. Performance of fixed–order object appearance models with varying numbers of parts K. Part association priors $\boldsymbol{\pi}_\ell \sim \mathrm{Dir}(\alpha_0)$ are biased towards sparse distributions (eq. (5.42)). We compare detection and recognition performance given 4 (top) or 15 (bottom) training images per category.

metric models which automatically determine an appropriate number of latent parts. As in the fixed–order object model, we augment the HDP with image–specific spatial transformations, and thus model datasets which are not spatially aligned.

Let H_w denote a Dirichlet prior on feature appearance distributions, H_v a normal–inverse–Wishart prior on feature position distributions, and $H_w \times H_v$ the corresponding product measure. As in the standard HDP model described in Sec. 2.5.4, a global probability measure $G_0 \sim \mathrm{DP}(\gamma, H_w \times H_v)$ constructs an infinite set of shared parts:

$$G_0(\theta) = \sum_{k=1}^{\infty} \beta_k \delta(\theta, \theta_k) \qquad \begin{aligned} \boldsymbol{\beta} &\sim \mathrm{GEM}(\gamma) \\ (\eta_k, \mu_k, \Lambda_k) = \theta_k &\sim H_w \times H_v \end{aligned} \tag{5.43}$$

Object categories are then defined by reweighted distributions $G_\ell \sim \mathrm{DP}(\alpha, G_0)$, which reuse these global parts in varying proportions:

$$G_\ell(\theta) = \sum_{t=1}^{\infty} \widetilde{\pi}_{\ell t} \delta(\theta, \widetilde{\theta}_{\ell t}) \qquad \begin{aligned} \widetilde{\boldsymbol{\pi}}_\ell &\sim \mathrm{GEM}(\alpha) \\ \widetilde{\theta}_{\ell t} &\sim G_0 \end{aligned} \tag{5.44}$$

$$= \sum_{k=1}^{\infty} \pi_{\ell k} \delta(\theta, \theta_k) \qquad \boldsymbol{\pi}_\ell \sim \mathrm{DP}(\alpha, \boldsymbol{\beta}) \tag{5.45}$$

As discussed in Sec. 2.5.4, each *local* part t (eq. (5.44)) has parameters $\widetilde{\theta}_{\ell t}$ copied from some *global* part $\theta_{k_{\ell t}}$, indicated by $k_{\ell t} \sim \boldsymbol{\beta}$. Aggregating the probabilities associated with these copies, eq. (5.45) then directly expresses each object's appearance via the common set of shared, global parts.

Consider the generative process for image j, which depicts object o_j and contains N_j features $(\mathbf{w}_j, \mathbf{v}_j)$. As before, each image has a corresponding reference transformation ρ_j, whose category–specific Gaussian prior equals

$$\rho_j \sim \mathcal{N}\big(\zeta_{o_j}, \Upsilon_{o_j}\big) \qquad\qquad j = 1, \dots, J \tag{5.46}$$

Each feature (w_{ji}, v_{ji}) is then independently sampled from some part of object o_j:

$$\begin{aligned} (\bar{\eta}_{ji}, \bar{\mu}_{ji}, \bar{\Lambda}_{ji}) &\sim G_{o_j} \\ w_{ji} &\sim \bar{\eta}_{ji} \\ v_{ji} &\sim \mathcal{N}\big(\bar{\mu}_{ji} + \rho_j, \bar{\Lambda}_{ji}\big) \end{aligned} \tag{5.47}$$

As before, we assume that the reference transformation ρ_j induces a simple translation of the observed features. Generalizations involving more complex transformations, like those discussed in Sec. 5.2.2, are also possible.

In Fig. 5.14, we provide a directed graphical representation of our HDP object appearance model. Marginalizing the unobserved assignments of features to latent parts, object appearance is defined by the following infinite mixture model:

$$p(w_{ji}, v_{ji} \mid \rho_j, o_j = \ell) = \sum_{k=1}^{\infty} \pi_{\ell k} \eta_k(w_{ji}) \mathcal{N}(v_{ji}; \mu_k + \rho_j, \Lambda_k) \tag{5.48}$$

This approach generalizes the parametric, fixed–order object model of Fig. 5.5 by defining an infinite set of potential global parts, and using the Dirichlet process' stick–breaking prior to automatically choose an appropriate model order. It also extends the HDP of Fig. 2.28 by associating a different reference transformation with each training image. Fig. 5.14 shows a related model in which each object's parts are defined by an independent Dirichlet process prior, and thus not shared among categories.

■ 5.5.1 Gibbs Sampling for Hierarchical Dirichlet Processes

To develop a learning algorithm for the nonparametric object appearance model of Fig. 5.14, we consider the "Chinese restaurant franchise" description of the HDP (see Sec. 2.5.4). This representation was previously used to develop a Rao–Blackwellized HDP Gibbs sampler [289]. In this section, we generalize this approach to also resample reference transformations, using methods similar to those underlying the fixed–order Gibbs sampler of Alg. 5.2.

As illustrated in Fig. 2.29, the Chinese restaurant franchise involves two sets of assignment variables. Object categories ℓ have infinitely many local parts (tables) t, which are assigned to global parts $k_{\ell t}$. Each observed feature, or customer, (w_{ji}, v_{ji}) is

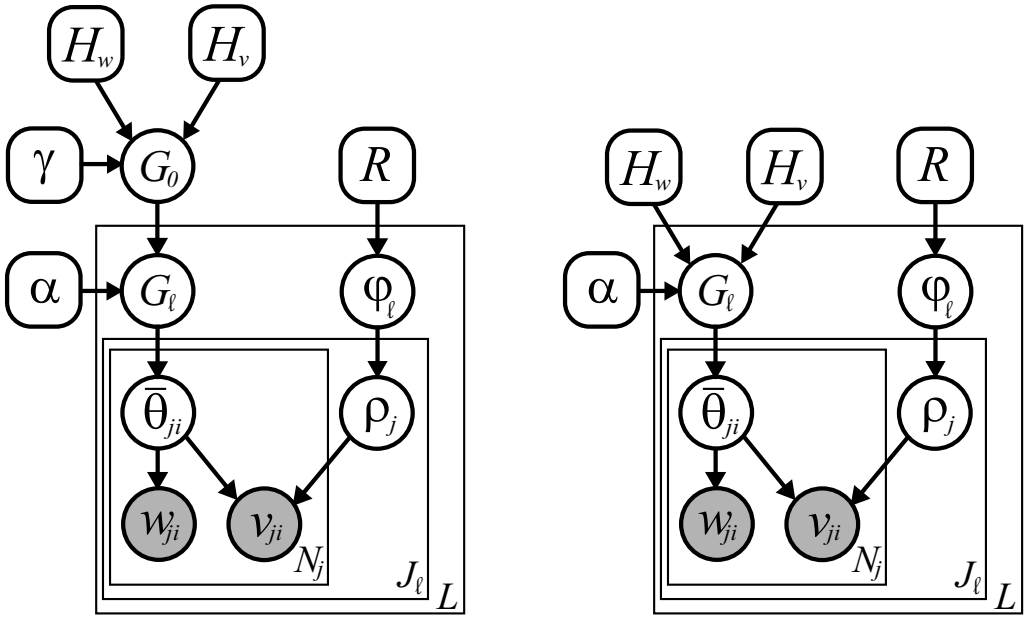


Figure 5.14. Nonparametric, Dirichlet process models for the visual appearance of L object categories. Each of the J_ℓ images of object ℓ has a reference transformation $\rho_j \sim \mathcal{N}(\zeta_\ell, \Upsilon_\ell)$, where $\varphi_\ell = (\zeta_\ell, \Upsilon_\ell)$. *Left:* Integrated, HDP model in which $G_0 \sim \mathrm{DP}(\gamma, H_w \times H_v)$ defines an infinite set of global parts, and objects reuse those parts via the reweighted distribution $G_\ell \sim \mathrm{DP}(\alpha, G_0)$. $\bar{\theta}_{ji} \sim G_\ell$ are then the part parameters used to generate feature (w_{ji}, v_{ji}). *Right:* A related model in which each category is described by an independent, infinite set of parts, with DP prior $G_\ell \sim \mathrm{DP}(\alpha, H_w \times H_v)$.

then assigned to some table t_{ji}. The proposed Gibbs sampler thus has three sets of state variables: assignments $\mathbf{t}$ of features to tables, assignments $\mathbf{k}$ of tables to global parts, and reference transformations $\boldsymbol{\rho}$ for each training image. Given these variables, the weights associated with the global (eq. (5.43)) and local (eq. (5.44)) part distributions can be analytically marginalized, as in the DP mixture Gibbs sampler of Alg. 2.3.

To avoid cumbersome notation, we let $z_{ji} = k_{o_j t_{ji}}$ denote the global part associated with feature (w_{ji}, v_{ji}). Note that z_{ji} is uniquely determined by that feature's table assignment $t_{ji} = t$, and the corresponding table's part assignment $k_{\ell t}$.

Table Assignment Resampling

We first consider the posterior distribution of the table assignment t_{ji} for feature (w_{ji}, v_{ji}), given all other state variables. Letting $\mathbf{t}_{\setminus ji}$ denote all table assignments excluding t_{ji}, the Markov properties of the HDP (see Figs. 2.29 and 5.14) imply that

$$p(t_{ji} \mid \mathbf{t}_{\setminus ji}, \mathbf{k}, \mathbf{w}, \mathbf{v}, \mathbf{o}, \boldsymbol{\rho}) \propto p(t_{ji} \mid \mathbf{t}_{\setminus ji}, o_j)\, p(w_{ji} \mid \mathbf{t}, \mathbf{k}, \mathbf{w}_{\setminus ji})\, p(v_{ji} \mid \mathbf{t}, \mathbf{k}, \mathbf{v}_{\setminus ji}, \boldsymbol{\rho}) \quad (5.49)$$

Let $N_{\ell t}^{-i}$ denote the number of features in images of object ℓ assigned to each of the T_ℓ tables currently instantiated by $\mathbf{t}_{\setminus ji}$. The clustering bias induced by the Chinese

restaurant process (see eq. (2.203)) then implies that

$$p(t_{ji} \mid \mathbf{t}_{\setminus ji}, o_j = \ell) \propto \sum_{t=1}^{T_\ell} N_{\ell t}^{-i} \delta(t_{ji}, t) + \alpha \delta(t_{ji}, \bar{t}) \tag{5.50}$$

where $\bar{t}$ denotes an assignment to a new, previously unoccupied table. For existing tables, the appearance likelihood depends on those features currently assigned to the same global, shared part:

$$p(w_{ji} = w \mid z_{ji} = k, \mathbf{t}_{\setminus ji}, \mathbf{k}, \mathbf{w}_{\setminus ji}) = \frac{C_{kw}^{-i} + \lambda/W}{\sum_{w'} C_{kw'}^{-i} + \lambda} \tag{5.51}$$

Here, C_{kw}^{-i} is the number of times appearance descriptor w is assigned to part k by $(\mathbf{t}_{\setminus ji}, \mathbf{k})$. The position likelihood similarly equals

$$\begin{aligned} p(v_{ji} \mid z_{ji} = k, \mathbf{t}_{\setminus ji}, \mathbf{k}, \mathbf{v}_{\setminus ji}, \boldsymbol{\rho}) &= p\big(v_{ji} - \rho_j \mid \{(v_{j'i'} - \rho_{j'}) \mid z_{j'i'} = k, (j', i') \neq (j, i)\}\big) \\ &\approx \mathcal{N}(v_{ji} - \rho_j; \hat{\mu}_k, \hat{\Lambda}_k) \end{aligned} \tag{5.52}$$

This predictive likelihood, whose derivation follows eqs. (5.22, 5.23), depends on the *transformed* positions of all other features currently assigned to the same global part.

For new tables $\bar{t}$, we improve sampling efficiency by integrating over potential assignments $k_{\ell\bar{t}}$ to global parts:

$$\begin{aligned} &p(w_{ji} = w, v_{ji} \mid t_{ji} = \bar{t}, \mathbf{t}_{\setminus ji}, \mathbf{k}, \mathbf{w}_{\setminus ji}, \mathbf{v}_{\setminus ji}, \boldsymbol{\rho}) \\ &\qquad \propto \sum_k p(k_{\ell\bar{t}} = k \mid \mathbf{k}) \cdot \left(\frac{C_{kw}^{-i} + \lambda/W}{\sum_{w'} C_{kw'}^{-i} + \lambda} \right) \cdot \mathcal{N}(v_{ji} - \rho_j; \hat{\mu}_k, \hat{\Lambda}_k) \end{aligned} \tag{5.53}$$

As in eq. (2.204), the probability of each of the K existing global parts depends on the number of other tables M_k which $\mathbf{k}$ assigns to that part:

$$p(k_{\ell\bar{t}} \mid \mathbf{k}) \propto \sum_{k=1}^{K} M_k \delta(k_{\ell\bar{t}}, k) + \gamma \delta(k_{\ell\bar{t}}, \bar{k}) \tag{5.54}$$

In this expression, $\bar{k}$ indicates a potential new global part, to which no tables or features are currently assigned. As in the DP mixture sampler of Alg. 2.3, our implementation maintains a dynamically resized list of those parts associated with at least one feature.

To sample according to eq. (5.49), we first evaluate the likelihoods of eqs. (5.51, 5.52) for each candidate table t and global part $k_{\ell\bar{t}}$. Combining these likelihoods with the Dirichlet process clustering biases of eqs. (5.50, 5.54), we sample a new table assignment t_{ji}, and if a new table $t_{ji} = \bar{t}$ is chosen, a corresponding part assignment $k_{\ell\bar{t}}$.

Global Part Assignment Resampling

We now consider the assignments $k_{\ell t}$ of tables to global parts, given fixed associations $\mathbf{t}$ between features and tables. Although each object category ℓ has infinitely many tables, we only explicitly resample assignments for the T_ℓ tables currently occupied by at least one feature ($N_{\ell t} > 0$).

Because $k_{\ell t}$ determines the part associated with all features assigned to table t, its posterior distribution depends on their joint likelihood. Let $\mathbf{w}_{\ell t} = \{w_{ji} \mid t_{ji} = t, o_j = \ell\}$ denote the appearance features assigned to table t, and $\mathbf{w}_{\setminus \ell t}$ all other appearance features. Defining $\mathbf{v}_{\ell t}$ and $\mathbf{v}_{\setminus \ell t}$ similarly, we then have

$$p(k_{\ell t} \mid \mathbf{k}_{\setminus \ell t}, \mathbf{t}, \mathbf{w}, \mathbf{v}, \boldsymbol{\rho}) \propto p(k_{\ell t} \mid \mathbf{k}_{\setminus \ell t})\, p(\mathbf{w}_{\ell t} \mid \mathbf{t}, \mathbf{k}, \mathbf{w}_{\setminus \ell t})\, p(\mathbf{v}_{\ell t} \mid \mathbf{t}, \mathbf{k}, \mathbf{v}_{\setminus \ell t}, \boldsymbol{\rho}) \tag{5.55}$$

Here, the prior clustering bias is as in eq. (5.54), except that $k_{\ell t}$ is excluded when counting the number of tables M_k^{-t} assigned to each global part. The joint likelihood of $\mathbf{w}_{\ell t}$ is then determined by those features assigned to the same part:

$$p(\mathbf{w}_{\ell t} \mid k_{\ell t} = k, \mathbf{t}, \mathbf{k}_{\setminus \ell t}, \mathbf{w}_{\setminus \ell t}) \propto \int p\big(\eta_k \mid \{w_{j'i'} \mid z_{j'i'} = k, t_{j'i'} \neq t\}\big) \prod_{j,i \mid t_{ji} = t} p(w_{ji} \mid \eta_k)\, d\eta_k \tag{5.56}$$

This expression equals the predictive likelihood of the $N_{\ell t}$ appearance features occupying table t, as given by Prop. 2.1.4. The likelihood of $\mathbf{v}_{\ell t}$ has a similar form, except that part statistics are determined by transformed feature positions as in eq. (5.52). Evaluating these likelihoods for each of the K currently instantiated parts, as well as a potential new global part $\bar{k}$, we may then sample a new part assignment via eq. (5.55).

Reference Transformation Resampling

Fixing the values of all assignment variables $(\mathbf{t}, \mathbf{k})$, each feature is associated with a unique global part. Given these associations, the posterior distribution of ρ_j is *identical* to that arising in a fixed–order object model (see Fig. 5.5) with matching assignments. Thus, as in Sec. 5.3.4, we may efficiently resample reference transformations via an auxiliary variable method.

Suppose that $(\mathbf{t}, \mathbf{k})$ assign at least one feature to each of K different global parts. For each instantiated part, we sample a single set of feature position parameters from their resulting posterior distribution:

$$(\hat{\mu}_k, \hat{\Lambda}_k) \sim p(\mu_k, \Lambda_k \mid \{(v_{ji} - \rho_j) \mid z_{ji} = k\}) \qquad k = 1, \ldots, K \tag{5.57}$$

As before, these auxiliary samples are often closely approximated by their corresponding posterior modes. The posterior distribution of reference transformation ρ_j then factors exactly as in eq. (5.25), and has the closed Gaussian form of eq. (5.27).

Concentration Parameter Resampling

The preceding sampling equations assumed fixed values for the concentration parameters γ and α defining the HDP's stick–breaking priors (see eqs. (5.43, 5.44)). In practice, these parameters noticeably impact the number of global and local parts learned by the Gibbs sampler. As with standard Dirichlet process mixtures (see Alg. 2.3), it is thus preferable to choose weakly informative gamma priors for these concentration parameters. Auxiliary variable methods may then be used to resample α and γ following each Gibbs iteration [76, 289]. Our incorporation of reference transformations does not change the form of these resampling steps, which are described in detail by [289].

■ 5.5.2 Learning Dirichlet Process Facial Appearance Models

To illustrate the use of Dirichlet processes in learning part–based object appearance models, we revisit the Caltech face database [318] considered in Sec. 5.3.3. We assign a gamma prior $\alpha \sim \text{Gamma}(0.1, 0.1)$ to the DP concentration parameter, and set the hyperparameters defining part distributions as before. Because this example involves a single object category, the global part distribution G_0 is unnecessary. We thus let $\gamma \to \infty$, so that $G_0 = H_w \times H_v$ as in the unshared model of Fig. 5.14. The resulting Gibbs sampler is similar to methods for standard DP mixtures (see Alg. 2.3).

In Fig. 5.15, we show the parts instantiated by the 300^{th} iteration of the Gibbs sampler of Sec. 5.5.1. Interestingly, the number of parts learned by the model grows with the number of available training images. Given 4 training images, roughly ten parts are used, avoiding the overfitting exhibited by the fixed–order, 25 part model of Fig. 5.6. When more training images become available, additional parts are created, thus allowing more detailed segmentations. As discussed in Sec. 2.5.2, the Dirichlet process prior encourages models whose complexity grows as data is observed. By resampling α, we allow a data–driven growth rate to be determined automatically.

■ 5.6 Nonparametric Models for Sixteen Object Categories

We now examine the sixteen category dataset of Fig. 5.7 using our HDP object appearance model. Part distribution hyperparameters H_w and H_v were set as in Sec. 5.4.1, and concentration parameters are assigned weakly informative priors $\gamma \sim \text{Gamma}(5, 0.1)$, $\alpha \sim \text{Gamma}(0.1, 0.1)$. We then learn an appropriate number of global parts, and their corresponding parameters, via the Gibbs sampler of Sec. 5.5.1.

■ 5.6.1 Visualization of Shared Parts

Given 30 training images from each of the 16 categories, we extracted features as in Sec. 5.4.1, and ran the Gibbs sampler of Sec. 5.5.1 for 1000 iterations. After an initial burn–in phase, there were typically between 120 and 140 global parts associated with at least one observation (see Fig. 5.16). As observed with standard Dirichlet process mixtures, however, some of these parts have very small posterior probability (see Sec. 2.5.3).

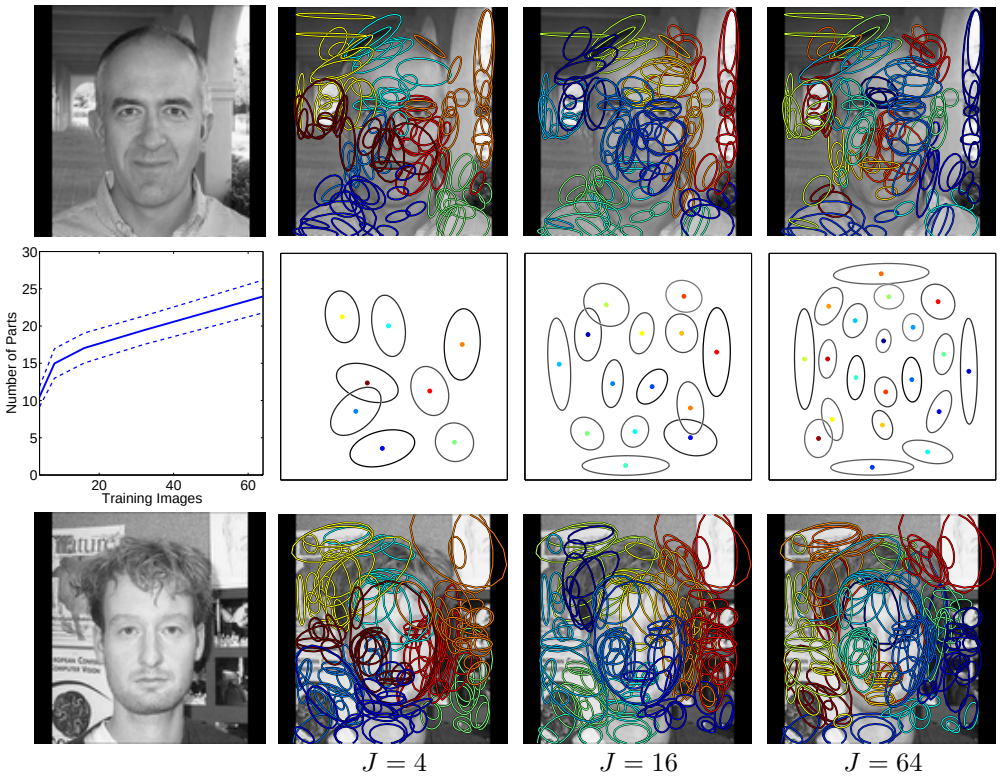


Figure 5.15. Visualization of single category, Dirichlet process facial appearance models (see Fig. 5.14). For different training set sizes J (columns), we plot the posterior mean and variance in the number of model parts (center row, left), and Gaussian position distributions $\mathcal{N}(\mu_k, \Lambda_k)$ for those parts accounting for 95% of the observed features (center row, right). For each model, we also show two color–coded segmentations of image features according to their most likely corresponding part. Dirichlet processes make robust predictions by estimating simple models from small training sets, and creating more parts as additional images are observed.

In Fig. 5.17, we visualize the feature distributions corresponding to seven of the more significant parts. A few seem specialized to distinctive features of individual categories, such as the spots appearing on the leopard's forehead. Many other parts are shared among several categories, modeling common aspects such as ears, mouths, and wheels. Compared to the fixed–order model illustrated in Fig. 5.8, the HDP's parts model fewer features, and cover smaller spatial areas. Note that the specific number of parts, and corresponding granularity of the appearance model, are determined automatically via the Dirichlet process prior.

As with the fixed–order object model, we also visualize pairwise distances between the object–specific part distributions $\boldsymbol{\pi}_\ell$ learned by the Gibbs sampler. We again measure distance via the symmetrized KL divergence of eq. (5.40), using only those parts associated with at least one feature. Fig. 5.18 shows the two–dimensional metric MDS embedding of these distances, and a corresponding agglomerative clustering [257]. Note

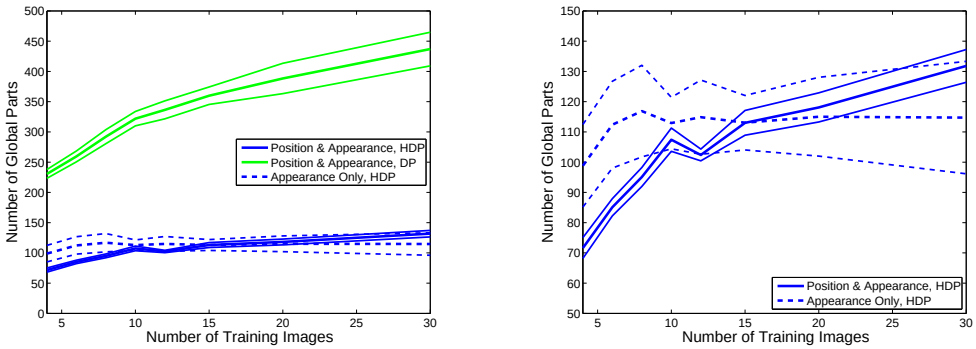


Figure 5.16. Mean (thick lines) and variance (thin lines) of the number of global parts created by the HDP Gibbs sampler (Sec. 5.5.1), given training sets of varying size. *Left:* Number of global parts used by HDP object models (blue), and the total number of parts instantiated by sixteen independent DP object models (green). *Right:* Expanded view of the parts instantiated by the HDP object models.

that the coarse–level object groups underlying this dataset (animal faces, animal profiles, vehicles) are more accurately identified than they were by the fixed–order model (see Fig. 5.9). In addition, the similarities among the three categories of cat faces, and among those animals with elongated faces, are reflected in the learned, shared parts.

■ 5.6.2 Detection and Recognition Performance

To evaluate our HDP object appearance model, we consider the same detection and recognition tasks examined in Sec. 5.4.2. We compare the HDP model to a set of 16 independent DP models (see Fig. 5.14), and also consider corresponding "bag of features" models based solely on feature appearance. In Fig. 5.16, we illustrate the number of global parts instantiated by the Gibbs samplers for the HDP models. The appearance–only HDP model learns a consistent number of parts given between 10 and 30 training images, while the HDP model of feature positions uses additional parts as more images are observed. We also show the considerably larger number of total parts (roughly 25 per category) employed by the independent DP models of feature positions. Because we use multinomial appearance distributions, estimation of the number of parts defining a single–category, appearance–only model is ill–posed, and sensitive to the concentration parameter α We thus do not show this model in Fig. 5.16.

Fig. 5.19 shows detection and recognition performance given training sets containing between 4 and 30 images per category. Likelihoods are estimated from 40 samples extracted across 1000 iterations of the Gibbs sampler. As with the fixed–order model, sharing significantly improves detection performance when few training images are available. In this case, however, the independent DP models of each category come closer to the HDP's accuracy by using more parts, and hence computation. As expected, these nonparametric models perform similarly to fixed–order object models employing equal numbers of parts (see Sec. 5.4). In some cases, however, the HDP Gibbs sampler discovers a more appropriate number of parts than those we examined for the fixed–order

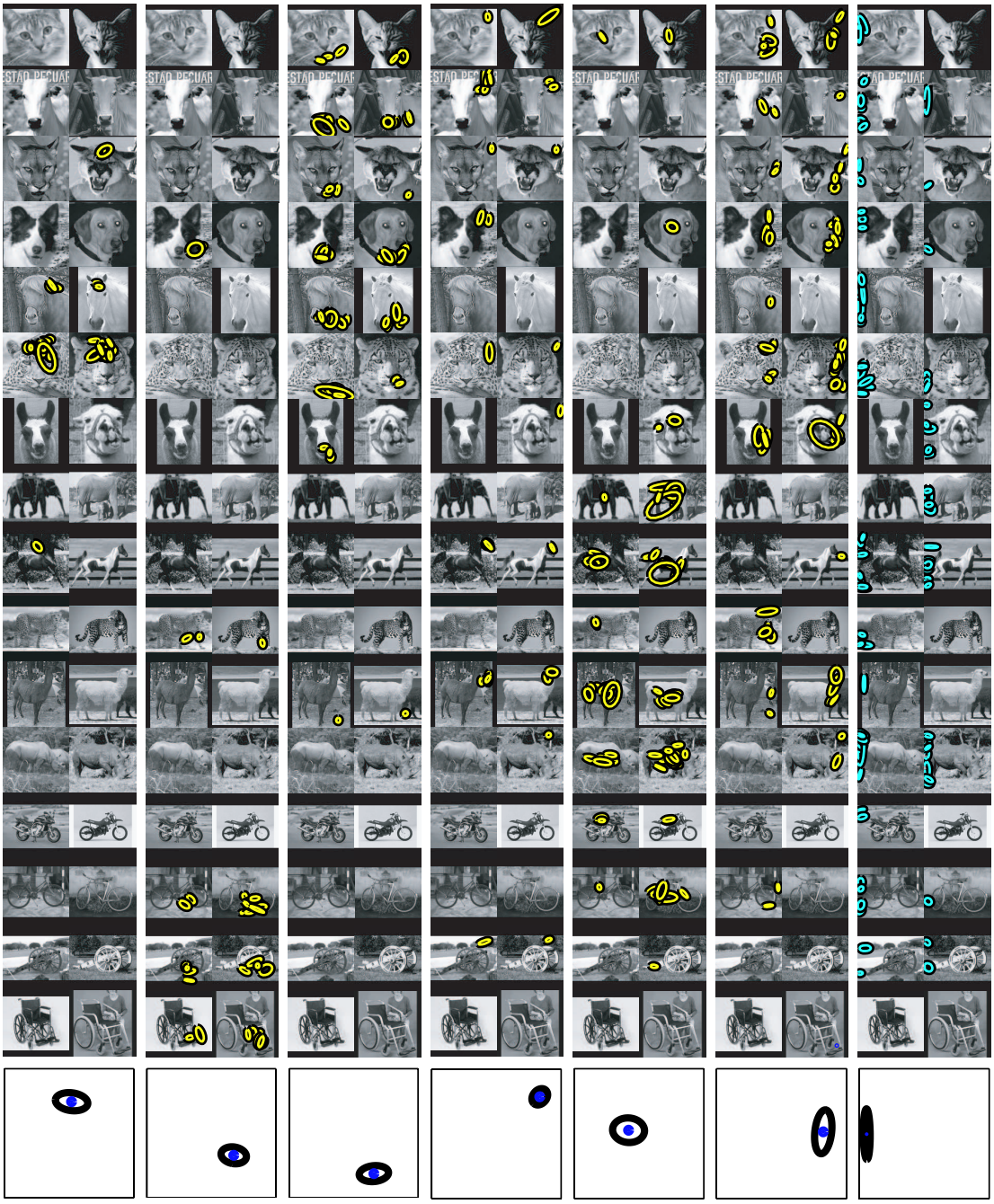


Figure 5.17. Seven of the 135 shared parts (columns) learned by an HDP model for 16 object categories (rows). Using two images from each category, we display those features with the highest posterior probability of being generated by each part. For comparison, we show six of the parts which are specialized to the fewest object categories (left, yellow), as well as one of several widely shared parts (right, cyan), which seem to model texture and background clutter. The bottom row plots the Gaussian position densities corresponding to each part. Compared to the parametric model of Fig. 5.8, the HDP uses additional parts to create more detailed appearance models. These parts also more closely align with the coarse–level object groupings underlying this dataset.

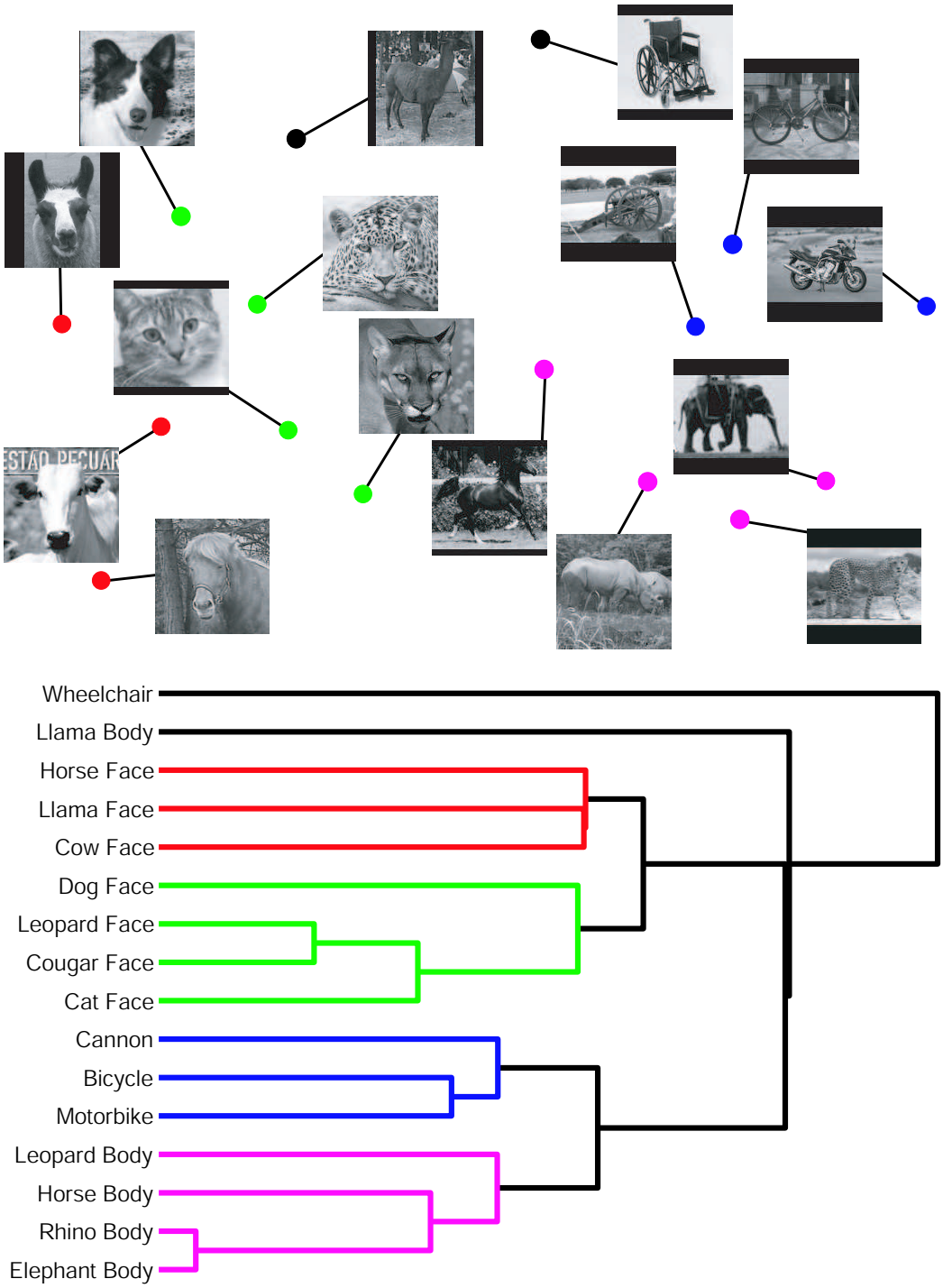


Figure 5.18. Two visualizations of learned part distributions $\boldsymbol{\pi}_\ell$ for the HDP object appearance model depicted in Fig. 5.17. *Top:* Two–dimensional embedding computed by metric MDS, in which coordinates for each object category are chosen to approximate pairwise KL distances as in eq. (5.40). *Bottom:* Dendrogram illustrating a greedy, hierarchical clustering, where branch lengths are proportional to inter–category distances. The four most significant clusters, which very intuitively align with semantic relationships among these categories, are highlighted in color.

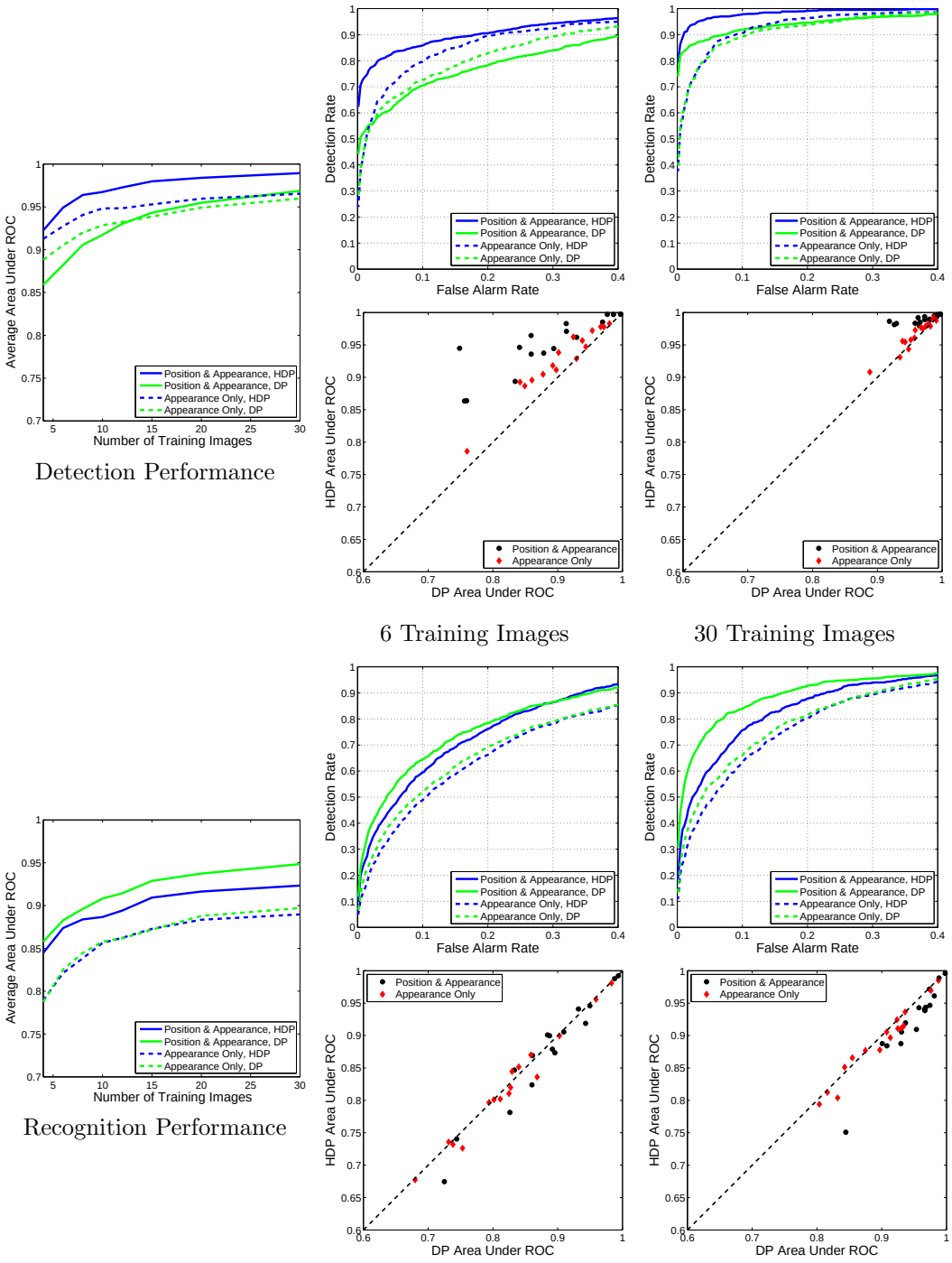


Figure 5.19. Performance of Dirichlet process object appearance models for the detection (top block) and recognition (bottom block) tasks. *Left:* Area under average ROC curves for different numbers of training images per category. *Top Right:* Average of ROC curves across all categories (6 versus 30 training images). *Bottom Right:* Scatter plot of areas under ROC curves for the shared and unshared models of individual categories (6 versus 30 training images).

model, and thus performs better.

For the recognition task, the HDP model of feature positions is slightly less effective than the independent DP models. However, it improves on our earlier fixed–order model by using more parts, and thus better differentiating the most similar pairs of object categories. This improvement is reflected by the more uniform object spacings in the MDS embedding of Fig. 5.18. For both detection and recognition, we again find that the spatial structure of visual features provides important information, and leads to more effective object appearance models.

5.7 Discussion

The preceding results demonstrate the potential benefits of transferring information among object categories when learning from few examples. Interestingly, while shared parts lead to substantial gains in distinguishing objects from background clutter, they may slightly reduce the discriminability of very similar categories; Chap. 7 discusses this tradeoff in more detail. Our results further show that nonparametric, Dirichlet process priors lead to learning algorithms which cleanly avoid model selection issues. Motivated by these advantages, Chap. 6 develops richer nonparametric models which analyze multiple object scenes.

Chapter 6

Scene Understanding via Transformed Dirichlet Processes

COMPUTER vision systems are most challenged by the interpretation of unconstrained, dynamic environments. In some cases, models of individual object appearance are directly adapted to scene understanding tasks. However, doing so neglects valuable contextual information which can disambiguate unexpected or partially occluded visual features. In this chapter, we instead design integrated, hierarchical models for multiple object scenes. Extending the nonparametric methods employed in previous chapters, we develop algorithms which robustly discover the number of parts composing each object, and objects depicted in each image.

As in Chap. 5, we describe images via a collection of affinely adapted interest regions. We begin in Sec. 6.1 by developing a parametric, fixed–order model which describes known sets of objects using a common set of shared parts. While this model desirably captures contextual relationships among objects, it restrictively assumes that the *number* of object instances depicted in each scene is known. To address this issue, Sec. 6.2 develops a nonparametric framework which couples Dirichlet processes with spatial transformations. The resulting *transformed Dirichlet process* (TDP) then provides a consistent, generative model for scenes in which the numbers of depicted object instances, and total object categories, are uncertain.

We provide two sample applications of the TDP to scene understanding tasks. Sec. 6.3 first develops a two–dimensional model which uses image–based translations to learn part–based object appearance models. Generalizing this approach, Sec. 6.4 then introduces a hierarchical description of the three–dimensional structure and appearance of visual scenes. We calibrate this model with binocular stereo training images, and thereby simultaneously recognize objects and reconstruct scene geometry. Both TDP models employ efficient Monte Carlo algorithms which analytically marginalize many parameters, and thus achieve robust learning with few manually specified parameters.

This chapter describes models developed in collaboration with Dr. Antonio Torralba. Some results were presented at the 2005 IEEE International Conference on Computer Vision [280], the 2005 Conference on Neural Information Processing Systems [282], and the 2006 IEEE Conference on Computer Vision and Pattern Recognition [281].

6.1 Contextual Models for Fixed Sets of Objects

We begin by revisiting the parametric, fixed–order object appearance model developed in Sec. 5.3. As summarized in Fig. 5.4, this model describes several object categories via a common set of shared parts. However, it restrictively assumes that each training or test image depicts a single object. In this section, we generalize this framework to describe *visual scenes* containing multiple objects. Retaining the fixed–order object model's parametric form, we assume that scenes contain fixed, *known* sets of objects. For example, a simple office scene might be defined by one computer screen, one keyboard, and one mouse. Later sections consider more flexible scene models, in which the number of object instances is also uncertain.

In Fig. 6.1, we summarize the proposed fixed–order model of visual scenes. The scene s_j associated with image j is defined by a fixed collection of L object categories. Conditioned on s_j, one of S possible scenes, the scene transformation ρ_j provides a reference frame for each object. For simplicity, we consider scale–normalized datasets, so that ρ_j is a $2L$–dimensional vector specifying each object's image coordinates. Each scene category is then given a different Gaussian transformation prior:

$$\rho_j \sim \mathcal{N}\big(\zeta_{s_j}, \Upsilon_{s_j}\big) \qquad\qquad j = 1, \dots, J \tag{6.1}$$

Because this Gaussian distribution has a full, $2L$–dimensional covariance matrix, we may learn contextual, scene–specific correlations in the locations at which objects are observed. As before, we regularize these transformation distributions with conjugate, normal–inverse–Wishart priors $(\zeta_s, \Upsilon_s) \sim R$.

Each visual scene is also associated with a discrete distribution $\boldsymbol{\beta}_s$, which specifies the proportion of observed features generated by each object. Each feature is generated by sampling an object category o_{ji}, and then a corresponding part z_{ji}:

$$\begin{aligned} o_{ji} &\sim \boldsymbol{\beta}_{s_j} \\ z_{ji} &\sim \boldsymbol{\pi}_{o_{ji}} \end{aligned} \qquad\qquad i = 1, \dots, N_j \tag{6.2}$$

Conditioned on these part assignments, the discrete appearance w_{ji} of each feature is independently sampled as in Sec. 5.3. Similarly, the feature position v_{ji} is determined by the chosen part, relative to the associated object's reference transformation:

$$\begin{aligned} w_{ji} &\sim \eta_{z_{ji}} \\ v_{ji} &\sim \mathcal{N}\big(\mu_{z_{ji}} + \rho_{j\ell}, \Lambda_{z_{ji}}\big) \qquad\qquad o_{ji} = \ell \end{aligned} \tag{6.3}$$

Here, $\rho_{j\ell}$ is the subvector of ρ_j corresponding to the reference transformation for object ℓ. While eq. (6.3) transforms objects via image–based translations, more complex pose variations could be modeled using richer transformation families (see Sec. 5.2).

Marginalizing the unobserved assignments z_{ji} of features to parts, we find that each object's appearance is defined by a different finite mixture model:

$$p(w_{ji}, v_{ji} \mid \rho_j, o_{ji} = \ell) = \sum_{k=1}^{K} \pi_{\ell k} \eta_k(w_{ji}) \mathcal{N}(v_{ji}; \mu_k + \rho_{j\ell}, \Lambda_k) \tag{6.4}$$

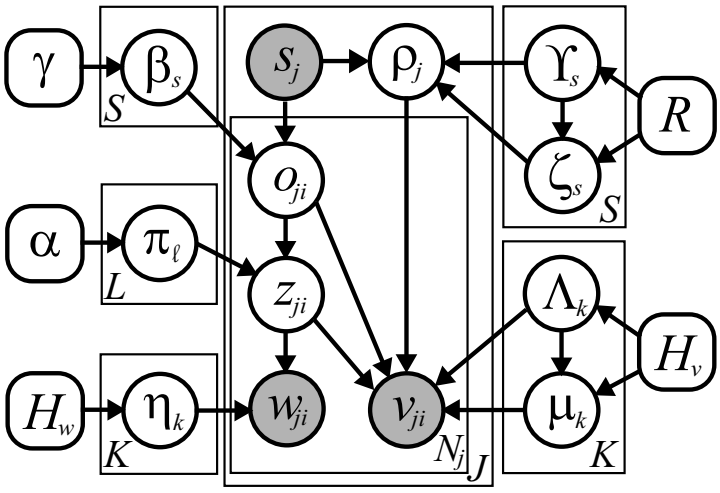


Figure 6.1. A parametric model for visual scenes containing fixed sets of objects. The j^{th} image depicts visual scene s_j, which combines L object categories at locations determined by the vector ρ_j of reference transformations. Each object category is in turn defined by a distribution $\boldsymbol{\pi}_\ell$ over a common set of K shared parts. The appearance w_{ji} and position v_{ji} of visual features, relative to the position of associated object o_{ji}, are then determined by assignments $z_{ji} \sim \boldsymbol{\pi}_{o_{ji}}$ to latent parts.

For scenes containing a single object, this model is equivalent to the fixed–order model of Sec. 5.3. More generally, however, eq. (6.4) faithfully describes images containing several objects, which differ in their observed locations and underlying part–based decompositions. The graph of Fig. 6.1 generalizes the author–topic model [247] by incorporating reference transformations, and by not constraining objects (authors) to generate equal proportions of image features (words).

■ 6.1.1 Gibbs Sampling for Multiple Object Scenes

Learning and inference in the scene–object–part hierarchy of Fig. 6.1 is possible via direct generalizations of the algorithms developed in Sec. 5.3. Extending Alg. 5.2, we first describe a Gibbs sampler which alternatively samples assignments (o_{ji}, z_{ji}) of features to objects and parts, and corresponding reference transformations ρ_j. This method generalizes a Rao–Blackwellized Gibbs sampler previously proposed for the author–topic model [247]. Sec. 6.1.2 then develops an alternative, variational approximation based on incremental EM updates.

Object and Part Assignment Resampling

To improve convergence, we consider a blocked Gibbs sampler which jointly resamples the object o_{ji} and part z_{ji} associated with each feature. Given a fixed set of reference transformations $\boldsymbol{\rho} = \{\rho_j\}_{j=1}^J$, the posterior distribution of these assignment variables

factors as follows:

$$p(o_{ji}, z_{ji} \mid \mathbf{o}_{\setminus ji}, \mathbf{z}_{\setminus ji}, \mathbf{w}, \mathbf{v}, \mathbf{s}, \boldsymbol{\rho}) \propto p(o_{ji} \mid \mathbf{o}_{\setminus ji}, s_j)\, p(z_{ji} \mid \mathbf{z}_{\setminus ji}, o_{ji})\, p(w_{ji} \mid \mathbf{z}, \mathbf{w}_{\setminus ji})\, p(v_{ji} \mid \mathbf{z}, \mathbf{v}_{\setminus ji}, \mathbf{o}, \boldsymbol{\rho}) \quad (6.5)$$

Let $M_{s\ell}^{-i}$ denote the number of times $\mathbf{o}_{\setminus ji}$ assigns features to object ℓ in images of scene s. Because $\boldsymbol{\beta}_s \sim \mathrm{Dir}(\gamma)$ is assigned a symmetric Dirichlet prior, we then have

$$p(o_{ji} = \ell \mid \mathbf{o}_{\setminus ji}, s_j = s) = \frac{M_{s\ell}^{-i} + \gamma/L}{\sum_{\ell'} M_{s\ell'}^{-i} + \gamma} \quad (6.6)$$

Similarly, the Dirichlet priors assigned to each object's part distribution $\boldsymbol{\pi}_\ell \sim \mathrm{Dir}(\alpha)$, and each part's appearance distribution $\eta_k \sim \mathrm{Dir}(\lambda)$, imply that

$$p(z_{ji} = k \mid \mathbf{z}_{\setminus ji}, o_{ji} = \ell) = \frac{N_{\ell k}^{-i} + \alpha/K}{\sum_{k'} N_{\ell k'}^{-i} + \alpha} \quad (6.7)$$

$$p(w_{ji} = w \mid z_{ji} = k, \mathbf{z}_{\setminus ji}, \mathbf{w}_{\setminus ji}) = \frac{C_{kw}^{-i} + \lambda/W}{\sum_{w'} C_{kw'}^{-i} + \lambda} \quad (6.8)$$

As in the fixed–order object model (see eqs. (5.17, 5.18)), C_{kw}^{-i} denotes the number of times appearance descriptor w is assigned to part k by $\mathbf{z}_{\setminus ji}$, and $N_{\ell k}^{-i}$ the number of features simultaneously assigned to object ℓ and part k. Note that features associated with different objects contribute to a common set of K shared parts.

The position likelihood of eq. (6.3) models features v_{ji} relative to the position of the currently associated object o_{ji}. Via an argument analogous to that used for the single–object model in Sec. 5.3.4 (see eq. (5.22)), the posterior distribution of (μ_k, Λ_k) is normal–inverse–Wishart, and depends on features *transformed* by the assigned objects' reference positions. The predictive likelihood of eq. (6.5) then equals

$$\begin{aligned} &p(v_{ji} \mid z_{ji} = k, o_{ji} = \ell, \mathbf{z}_{\setminus ji}, \mathbf{o}_{\setminus ji}, \mathbf{v}_{\setminus ji}, \boldsymbol{\rho}) \\ &\quad \propto p\big(v_{ji} - \rho_{j\ell} \mid \big\{(v_{j'i'} - \rho_{j'\ell'}) \mid z_{j'i'} = k, o_{j'i'} = \ell', (j', i') \neq (j, i)\big\}\big) \\ &\quad \approx \mathcal{N}(v_{ji} - \rho_{j\ell}; \hat{\mu}_k, \hat{\Lambda}_k) \end{aligned} \quad (6.9)$$

As discussed in Sec. 2.1.4, this expression approximates the Student–t predictive likelihood with a moment–matched Gaussian. Each part then caches feature position sums and outer products relative to the assigned objects' reference transformations. Combining these expressions, we may evaluate eq. (6.5) for all potential object and part assignments in $\mathcal{O}(LK)$ operations, and thus jointly resample (o_{ji}, z_{ji}).

Reference Transformation Resampling

As discussed in Sec. 5.3.4, while marginalization of the parameters defining each part improves performance, it also complicates transformation resampling. To address this

issue, we again employ an auxiliary variable method. Fixing all assignments $(\mathbf{o}, \mathbf{z})$ of features to objects and parts, we draw a single sample from the normal–inverse–Wishart posterior distribution of each part's position parameters:

$$(\hat{\mu}_k, \hat{\Lambda}_k) \sim p(\mu_k, \Lambda_k \mid \{(v_{ji} - \rho_{j\ell}) \mid z_{ji} = k, o_{ji} = \ell\}) \qquad k = 1, \ldots, K \tag{6.10}$$

Given these parameters, the posterior distribution of the reference transformation ρ_j for the visual scene in image j equals

$$p\big(\rho_j \mid \boldsymbol{\rho}_{\setminus j}, \mathbf{s}, \mathbf{o}, \mathbf{z}, \mathbf{v}, \{\hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K\big) \propto p\big(\rho_j \mid \boldsymbol{\rho}_{\setminus j}, \mathbf{s}\big) \prod_{k=1}^{K} \prod_{i \mid z_{ji} = k} \mathcal{N}\big(v_{ji} - \rho_{jo_{ji}}; \hat{\mu}_k, \hat{\Lambda}_k\big) \tag{6.11}$$

Recall that ρ_j is a $2L$–dimensional vector, which defines a reference position for each of the L objects in scene s_j. The first term of eq. (6.11) is an approximately Gaussian prior implied by the current transformations in other images of the same scene:

$$p\big(\rho_j \mid \boldsymbol{\rho}_{\setminus j}, \mathbf{s}\big) = p\big(\rho_j \mid \big\{\rho_{j'} \mid s_{j'} = s_j\big\}\big) \approx \mathcal{N}(\rho_j; \hat{\zeta}_{s_j}, \hat{\Upsilon}_{s_j}) \tag{6.12}$$

This expression again approximates a Student–t predictive distribution by an appropriately moment–matched Gaussian.

Examining eq. (6.11), we see that each feature v_{ji} effectively provides a Gaussian observation of the 2–dimensional *subvector* of ρ_j corresponding to the currently assigned object o_{ji}. The posterior transformation distribution is then also Gaussian, with mean and covariance given by the following information form:

$$p\big(\rho_j \mid \boldsymbol{\rho}_{\setminus j}, \mathbf{s}, \mathbf{o}, \mathbf{z}, \mathbf{v}, \{\hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K\big) \approx \mathcal{N}(\rho_j; \chi_j, \Xi_j) \tag{6.13}$$

$$\Xi_j^{-1} = \hat{\Upsilon}_{s_j}^{-1} + \text{blkdiag}\left\{ \sum_{k=1}^{K} \sum_{\substack{i \mid z_{ji} = k \\ o_{ji} = 1}} \hat{\Lambda}_k^{-1}, \ldots, \sum_{k=1}^{K} \sum_{\substack{i \mid z_{ji} = k \\ o_{ji} = L}} \hat{\Lambda}_k^{-1} \right\}$$

$$\Xi_j^{-1} \chi_j = \hat{\Upsilon}_{s_j}^{-1} \hat{\zeta}_{s_j} + \left[\sum_{k=1}^{K} \sum_{\substack{i \mid z_{ji} = k \\ o_{ji} = 1}} \hat{\Lambda}_k^{-1} (v_{ji} - \hat{\mu}_k), \ldots, \sum_{k=1}^{K} \sum_{\substack{i \mid z_{ji} = k \\ o_{ji} = L}} \hat{\Lambda}_k^{-1} (v_{ji} - \hat{\mu}_k) \right]^T$$

By caching the statistics required by this expression, we may then sample a new reference transformation in $\mathcal{O}(L^3)$ operations. Note that the influence of each feature on the posterior distribution of eq. (6.13) depends on the covariance of its associated part. After resampling ρ_j, the auxiliary part parameters $\{\hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K$ are discarded to allow Rao–Blackwellized assignment sampling.

Alg. 6.1 summarizes a Gibbs sampler based on the preceding analysis. As in Chap. 5, we define $(\hat{\mu}_k, \hat{\Lambda}_k) \oplus v_{ji}$ to be an operator which updates a normal–inverse–Wishart posterior based on a new feature v_{ji} (see eqs. (2.62, 2.63)). Similarly, $(\hat{\mu}_k, \hat{\Lambda}_k) \ominus v_{ji}$ removes v_{ji} from the posterior statistics of part k. Like the single–object Gibbs sampler

Given a previous reference transformation $\rho_j^{(t-1)}$, and object and part assignments $(\mathbf{o}_j^{(t-1)}, \mathbf{z}_j^{(t-1)})$ for the N_j features in an image depicting scene $s_j = s$, resample these variables as follows:

1. Sample a random permutation $\tau(\cdot)$ of the integers $\{1, \dots, N_j\}$.
2. Set $(\mathbf{o}_j, \mathbf{z}_j) = (\mathbf{o}_j^{(t-1)}, \mathbf{z}_j^{(t-1)})$. For $i \in \{\tau(1), \dots, \tau(N_j)\}$, sequentially resample (o_{ji}, z_{ji}):
   (a) Remove feature (w_{ji}, v_{ji}) from the cached statistics for its current part and object:
   $$\begin{aligned} M_{s\ell} &\leftarrow M_{s\ell} - 1 & \ell &= o_{ji} \\ N_{\ell k} &\leftarrow N_{\ell k} - 1 & k &= z_{ji} \\ C_{kw} &\leftarrow C_{kw} - 1 & w &= w_{ji} \\ (\hat{\mu}_k, \hat{\Lambda}_k) &\leftarrow (\hat{\mu}_k, \hat{\Lambda}_k) \ominus (v_{ji} - \rho_{j\ell}^{(t-1)}) \end{aligned}$$
   (b) For each of the $L \cdot K$ pairs of objects and parts, determine the predictive likelihood
   $$f_{\ell k}(w_{ji} = w, v_{ji}) = \left(\frac{C_{kw} + \lambda/W}{\sum_{w'} C_{kw'} + \lambda} \right) \cdot \mathcal{N}(v_{ji} - \rho_{j\ell}^{(t-1)}; \hat{\mu}_k, \hat{\Lambda}_k)$$
   (c) Sample new object and part assignments from the following multinomial distribution:
   $$(o_{ji}, z_{ji}) \sim \frac{1}{Z_i} \sum_{\ell=1}^{L} \sum_{k=1}^{K} (M_{s\ell} + \gamma/L) \left(\frac{N_{\ell k} + \alpha/K}{\sum_{k'} N_{\ell k'} + \alpha} \right) f_{\ell k}(w_{ji}, v_{ji}) \delta(o_{ji}, \ell) \delta(z_{ji}, k)$$
   (d) Add feature (w_{ji}, v_{ji}) to the cached statistics for its new object and part:
   $$\begin{aligned} M_{s\ell} &\leftarrow M_{s\ell} + 1 & \ell &= o_{ji} \\ N_{\ell k} &\leftarrow N_{\ell k} + 1 & k &= z_{ji} \\ C_{kw} &\leftarrow C_{kw} + 1 & w &= w_{ji} \\ (\hat{\mu}_k, \hat{\Lambda}_k) &\leftarrow (\hat{\mu}_k, \hat{\Lambda}_k) \oplus (v_{ji} - \rho_{j\ell}^{(t-1)}) \end{aligned}$$
3. Set $(\mathbf{o}_j^{(t)}, \mathbf{z}_j^{(t)}) = (\mathbf{o}_j, \mathbf{z}_j)$. Optionally, part parameters $\{\eta_k^{(t)}, \mu_k^{(t)}, \Lambda_k^{(t)}\}_{k=1}^K$ may be sampled as in step 3 of Alg. 5.1. Object and part probabilities follow Dirichlet distributions.
4. Sample a new reference transformation $\rho_j^{(t)}$ as follows:
   (a) Remove $\rho_j^{(t-1)}$ from cached transformation statistics for scene s:
   $$(\hat{\zeta}_s, \hat{\Upsilon}_s) \leftarrow (\hat{\zeta}_s, \hat{\Upsilon}_s) \ominus \rho_j^{(t-1)}$$
   (b) Sample $\rho_j^{(t)} \sim \mathcal{N}(\chi_j, \Xi_j)$, a posterior distribution determined via eq. (6.13) from the prior $\mathcal{N}(\rho_j; \hat{\zeta}_s, \hat{\Upsilon}_s)$, cached part statistics $\{\hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K$, and feature positions $\mathbf{v}_j$.
   (c) Add $\rho_j^{(t)}$ to cached transformation statistics for scene s:
   $$(\hat{\zeta}_s, \hat{\Upsilon}_s) \leftarrow (\hat{\zeta}_s, \hat{\Upsilon}_s) \oplus \rho_j^{(t)}$$
5. For each $i \in \{1, \dots, N_j\}$, update cached statistics for part $k = z_{ji}$ as follows:
   $$\begin{aligned} (\hat{\mu}_k, \hat{\Lambda}_k) &\leftarrow (\hat{\mu}_k, \hat{\Lambda}_k) \ominus (v_{ji} - \rho_{j\ell}^{(t-1)}) \\ (\hat{\mu}_k, \hat{\Lambda}_k) &\leftarrow (\hat{\mu}_k, \hat{\Lambda}_k) \oplus (v_{ji} - \rho_{j\ell}^{(t)}) \end{aligned} \qquad \ell = o_{ji}$$

Algorithm 6.1. Rao–Blackwellized Gibbs sampler for the fixed–order visual scene model of Fig. 6.1. We illustrate the sequential resampling of all object and part assignments $(\mathbf{o}_j, \mathbf{z}_j)$ in the j^{th} training image, as well as that image's coordinate frame ρ_j. A full iteration of the Gibbs sampler applies these updates to all images in random order. For efficiency, we cache and recursively update statistics $\{\hat{\zeta}_s, \hat{\Upsilon}_s)\}_{s=1}^S$ of each scene's reference transformations, counts $M_{s\ell}$, $N_{\ell k}$ of the features assigned to each object and part, and statistics $\{C_{kw}, \hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K$ of those features' appearance and position.

of Sec. 5.3.4, we update cached statistics whenever reference transformations are resampled to ensure consistency (Alg. 6.1, step 5). Given a training set with J images, each containing N features, a Gibbs sampling update of every object and part assignment requires $\mathcal{O}(LKJN)$ operations.

■ 6.1.2 Inferring Likely Reference Transformations

In this section, we describe an alternative method for learning hierarchical models of visual scenes. Extending methods developed in Sec. 5.3.5, we use a variational approximation to integrate over reference transformations, and only explicitly sample assignments $(\mathbf{o}, \mathbf{z})$ of features to objects and parts. An incremental form of the EM algorithm [225] then efficiently updates variational parameters as features are reassigned.

From the graphical model of Fig. 6.1, the posterior distribution of (o_{ji}, z_{ji}) given observed image features, and other assignments $(\mathbf{o}_{\setminus ji}, \mathbf{z}_{\setminus ji})$, factors as follows:

$$p(o_{ji}, z_{ji} \mid \mathbf{o}_{\setminus ji}, \mathbf{z}_{\setminus ji}, \mathbf{w}, \mathbf{v}, \mathbf{s}) \propto p(o_{ji} \mid \mathbf{o}_{\setminus ji}, s_j)\, p(z_{ji} \mid \mathbf{z}_{\setminus ji}, o_{ji})\, p(w_{ji} \mid \mathbf{z}, \mathbf{w}_{\setminus ji})\, p(v_{ji} \mid \mathbf{z}, \mathbf{v}_{\setminus ji}, \mathbf{o}, \mathbf{s}) \quad (6.14)$$

The first three terms are unchanged from eqs. (6.6, 6.7, 6.8), but uncertainty in the position parameters $\{\mu_k, \Lambda_k\}_{k=1}^K$ causes the predictive position likelihood to depend on the latent object positions, and hence scene labels, of all training images. Sec. 6.1.1 simplified this term by conditioning on the reference transformation ρ_j. The likelihood of eq. (6.14) instead marginalizes over transformations. Letting $\boldsymbol{\theta} = \{\mu_k, \Lambda_k\}_{k=1}^K$ denote part position parameters and $\boldsymbol{\varphi} = \{\zeta_s, \Upsilon_s\}_{s=1}^S$ transformation parameters, we have

$$p(v_{ji} \mid \mathbf{z}, \mathbf{v}_{\setminus ji}, \mathbf{o}, \mathbf{s}) = \iint \left[\int p(v_{ji} \mid z_{ji}, o_{ji}, \rho_j, \boldsymbol{\theta})\, p(\rho_j \mid \mathbf{z}_{j\setminus i}, \mathbf{v}_{j\setminus i}, \mathbf{o}_{j\setminus i}, s_j, \boldsymbol{\varphi})\, d\rho_j \right] \cdots \times p(\boldsymbol{\theta}, \boldsymbol{\varphi} \mid \mathbf{z}_{\setminus ji}, \mathbf{v}_{\setminus ji}, \mathbf{o}_{\setminus ji}, \mathbf{s})\, d\boldsymbol{\theta}\, d\boldsymbol{\varphi} \quad (6.15)$$

Here, $\mathbf{o}_{j\setminus i}$ denotes the set $\mathbf{o}_j$ of object assignments for features in image j, excluding o_{ji}. As in Sec. 5.3.5, dependency between the part and transformation parameters makes this marginalized likelihood intractable. We therefore approximate it via the parameters' posterior mode:

$$(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\varphi}}) = \arg\max_{\boldsymbol{\theta}, \boldsymbol{\varphi}} p(\boldsymbol{\theta}, \boldsymbol{\varphi} \mid \mathbf{z}_{\setminus ji}, \mathbf{v}_{\setminus ji}, \mathbf{o}_{\setminus ji}, \mathbf{s}) \quad (6.16)$$

Given these parameters, the predictive likelihood of eq. (6.15) reduces to a simple Gaussian integral. We optimize parameters via a variant of the EM algorithm [107, 161] (see Sec. 2.3.3), in which the E–step determines Gaussian posteriors for reference transformations, and the M–step provides corresponding parameter estimates.

Expectation Step

In the E–step, we assume fixed values for the transformation parameters $\widehat{\boldsymbol{\varphi}} = \{\hat{\zeta}_s, \hat{\Upsilon}_s\}_{s=1}^S$ and part position parameters $\widehat{\boldsymbol{\theta}} = \{\hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K$, and determine posterior distributions

for the reference transformations $\boldsymbol{\rho} = \{\rho_j\}_{j=1}^J$. From the graph of Fig. 6.1, these distributions take the following form:

$$p(\rho_j \mid s_j = s, \mathbf{o}_j, \mathbf{z}_j, \mathbf{v}_j, \widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\varphi}}) \propto \mathcal{N}(\rho_j; \hat{\zeta}_s, \hat{\Upsilon}_s) \prod_{k=1}^{K} \prod_{i|z_{ji}=k} \mathcal{N}\big(v_{ji} - \rho_{jo_{ji}}; \hat{\mu}_k, \hat{\Lambda}_k\big) \tag{6.17}$$

This expression is identical to that arising in the auxiliary variable Gibbs sampler of Sec. 6.1.1 (see eq. (6.11)). Reference transformation ρ_j thus has a Gaussian posterior distribution $\mathcal{N}(\chi_j, \Xi_j)$, with mean and covariance as in eq. (6.13). While Alg. 6.1 uses this posterior to sample a new transformation, the variational approach of this section instead estimates parameters analytically in the subsequent M–step.

Maximization Step

In the M–step, we use Gaussian reference transformation distributions $\rho_j \sim \mathcal{N}(\chi_j, \Xi_j)$ from the previous E–step to lower bound the posterior distribution of eq. (6.16). Let $\mathcal{NW}(\kappa_s, \vartheta_s, \nu_s, \Delta_s)$ denote the hyperparameters of the normal–inverse–Wishart prior R on transformation parameters, as defined in Sec. 2.1.4. Constructing a likelihood bound as in eq. (2.134) and taking derivatives, we find that the maximizing transformation parameters equal

$$\hat{\zeta}_s = \frac{1}{\kappa_s + J_s}\left(\kappa_s \vartheta_s + \sum_{j|s_j=s} \chi_j\right) \tag{6.18}$$

$$\hat{\Upsilon}_s = \frac{(\kappa_s + J_s + 1)}{(\kappa_s + J_s)(\nu_s + J_s + 2L + 1)}\left(\nu_s \Delta_s + \sum_{j|s_j=s} \Xi_j + (\chi_j - \hat{\zeta}_s)(\chi_j - \hat{\zeta}_s)^T\right) \tag{6.19}$$

Here, J_s is the number of training images of visual scene s, and $2L$ is the dimension of the reference transformation ρ_j. Intuitively, eq. (5.32) sets the transformation mean $\hat{\zeta}_s$ to a regularized average of the current transformations χ_j in images of scene s. Similarly, eq. (5.33) combines outer products of transformations $(\chi_j - \hat{\zeta}_s)$ with their uncertainties Ξ_j to determine $\hat{\Upsilon}_s$.

Let $\mathcal{NW}(\kappa_v, \vartheta_v, \nu_v, \Delta_v)$ denote the hyperparameters of the normal–inverse–Wishart prior H_v on part position parameters. The M–step's part parameter estimates depend on the marginal distributions $\rho_{j\ell} \sim \mathcal{N}(\chi_{j\ell}, \Xi_{j\ell})$ of individual object transformations. Note that $\chi_{j\ell}$ is a subvector of χ_j, while $\Xi_{j\ell}$ is a block diagonal submatrix of Ξ_j. The

Given a previous transformation posterior $\mathcal{N}(\rho_j; \chi_j, \Xi_j)$, and assignments $(\mathbf{o}_j^{(t-1)}, \mathbf{z}_j^{(t-1)})$ for the N_j features in an image depicting scene $s_j = s$, resample objects and parts as follows:

1. Sample a random permutation $\tau(\cdot)$ of the integers $\{1, \ldots, N_j\}$.
2. Set $(\mathbf{o}_j, \mathbf{z}_j) = (\mathbf{o}_j^{(t-1)}, \mathbf{z}_j^{(t-1)})$. For $i \in \{\tau(1), \ldots, \tau(N_j)\}$, sequentially resample (o_{ji}, z_{ji}):
   (a) Remove feature (w_{ji}, v_{ji}) from the cached statistics for its current part and object:
   $$\begin{aligned} M_{s\ell} &\leftarrow M_{s\ell} - 1 & \ell &= o_{ji} \\ N_{\ell k} &\leftarrow N_{\ell k} - 1 & k &= z_{ji} \\ C_{kw} &\leftarrow C_{kw} - 1 & w &= w_{ji} \end{aligned}$$
   Update $(\hat{\mu}_k, \hat{\Lambda}_k)$ by subtracting $(v_{ji} - \chi_{j\ell})$ from mean statistics (eq. (6.20)), and $\Xi_{j\ell} + (v_{ji} - \chi_{j\ell} - \hat{\mu}_k)(v_{ji} - \chi_{j\ell} - \hat{\mu}_k)^T$ from covariance statistics (eq. (6.21)).
   (b) *E–Step:* Update the reference transformation's Gaussian posterior distribution $\mathcal{N}(\rho_j; \chi_j, \Xi_j)$ using eq. (6.13), excluding the currently unassigned feature v_{ji}.
   (c) *M–Step:* Compute new transformation parameters $(\hat{\zeta}_s, \hat{\Upsilon}_s)$ using eqs. (6.18, 6.19), and new part position parameters $\{\hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K$ using eqs. (6.20, 6.21).
   (d) For each of the $L \cdot K$ pairs of objects and parts, determine the predictive likelihood
   $$f_{\ell k}(w_{ji} = w, v_{ji}) = \left(\frac{C_{kw} + \lambda/W}{\sum_{w'} C_{kw'} + \lambda} \right) \cdot \mathcal{N}(v_{ji}; \chi_{j\ell} + \hat{\mu}_k, \Xi_{j\ell} + \hat{\Lambda}_k)$$
   (e) Sample new object and part assignments from the following multinomial distribution:
   $$(o_{ji}, z_{ji}) \sim \frac{1}{Z_i} \sum_{\ell=1}^{L} \sum_{k=1}^{K} (M_{s\ell} + \gamma/L) \left(\frac{N_{\ell k} + \alpha/K}{\sum_{k'} N_{\ell k'} + \alpha} \right) f_{\ell k}(w_{ji}, v_{ji}) \delta(o_{ji}, \ell) \delta(z_{ji}, k)$$
   (f) Add feature (w_{ji}, v_{ji}) to the cached statistics for its new object and part:
   $$\begin{aligned} M_{s\ell} &\leftarrow M_{s\ell} + 1 & \ell &= o_{ji} \\ N_{\ell k} &\leftarrow N_{\ell k} + 1 & k &= z_{ji} \\ C_{kw} &\leftarrow C_{kw} + 1 & w &= w_{ji} \end{aligned}$$
   Update $(\hat{\mu}_k, \hat{\Lambda}_k)$ by adding $(v_{ji} - \chi_{j\ell})$ to mean statistics (eq. (6.20)), and $\Xi_{j\ell} + (v_{ji} - \chi_{j\ell} - \hat{\mu}_k)(v_{ji} - \chi_{j\ell} - \hat{\mu}_k)^T$ to covariance statistics (eq. (6.21)).
3. Set $(\mathbf{o}_j^{(t)}, \mathbf{z}_j^{(t)}) = (\mathbf{o}_j, \mathbf{z}_j)$. Optionally, part parameters $\{\eta_k^{(t)}, \mu_k^{(t)}, \Lambda_k^{(t)}\}_{k=1}^K$ may be sampled as in step 3 of Alg. 5.1. Object and part probabilities follow Dirichlet distributions.

Algorithm 6.2. Rao–Blackwellized Gibbs sampler for the fixed–order visual scene model of Fig. 6.1, using a variational approximation to marginalize reference transformations. We illustrate the sequential resampling of all object and part assignments $(\mathbf{o}_j, \mathbf{z}_j)$ in the j^{th} training image, based on incremental EM updates of the model's position parameters. A full iteration of the Gibbs sampler applies these updates to all images in random order. For efficiency, we cache and recursively update statistics $\{\hat{\zeta}_s, \hat{\Upsilon}_s)\}_{s=1}^S$ of each scene's reference transformations, counts $M_{s\ell}, N_{\ell k}$ of the features assigned to each object and part, and statistics $\{C_{kw}, \hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^K$ of those features' appearance and position.

likelihood bound is then maximized by the following parameter estimates:

$$\hat{\mu}_k = \frac{1}{\kappa_v + N_k}\left(\kappa_v \vartheta_v + \sum_{j=1}^{J} \sum_{i|z_{ji}=k} (v_{ji} - \chi_{jo_{ji}})\right) \tag{6.20}$$

$$\hat{\Lambda}_k = \frac{(\kappa_v + N_k + 1)}{(\kappa_v + N_k)(\nu_v + N_k + 3)} \cdots \times \left(\nu_v \Delta_v + \sum_{j=1}^{J} \sum_{i|z_{ji}=k} \Xi_{jo_{ji}} + (v_{ji} - \chi_{jo_{ji}} - \hat{\mu}_k)(v_{ji} - \chi_{jo_{ji}} - \hat{\mu}_k)^T\right) \tag{6.21}$$

In this expression, N_k is the total number of features which $\mathbf{z}_{\backslash ji}$ currently assigns to part k. As in the Gibbs sampler of Alg. 6.1, part statistics depend on the relative displacements $(v_{ji} - \chi_{jo_{ji}})$ of image features from the reference positions of their currently associated objects.

Likelihood Evaluation and Incremental EM Updates

Given fixed assignments $(\mathbf{o}_{\backslash ji}, \mathbf{z}_{\backslash ji})$ of features to objects and parts, the preceding EM updates converge to a local maximum of the posterior distribution of eq. (6.16). Conditioned on the parameters $\widehat{\boldsymbol{\varphi}} = \{\hat{\zeta}_s, \hat{\Upsilon}_s\}_{s=1}^{S}$ and $\widehat{\boldsymbol{\theta}} = \{\hat{\mu}_k, \hat{\Lambda}_k\}_{k=1}^{K}$ computed in the final M–step, the reference transformation follows the Gaussian posterior $\rho_j \sim \mathcal{N}(\chi_j, \Xi_j)$ determined in the E–step via eq. (6.13). Integrating over ρ_j, the feature position likelihood of eq. (6.15) then has the following closed–form approximation:

$$p(v_{ji} \mid o_{ji} = \ell, z_{ji} = k, \mathbf{o}_{\backslash ji}, \mathbf{z}_{\backslash ji}, \mathbf{v}_{\backslash ji}, \mathbf{s}) \approx \mathcal{N}(v_{ji}; \chi_{j\ell} + \hat{\mu}_k, \Xi_{j\ell} + \hat{\Lambda}_k) \tag{6.22}$$

This approximation will be accurate when the posterior distribution of eq. (6.16) is concentrated around a single mode. Empirically, this is usually true given "consistent" feature assignments $(\mathbf{o}_{\backslash ji}, \mathbf{z}_{\backslash ji})$ which have high joint posterior probability.

To apply this analysis, we extend the single–object Gibbs sampler of Alg. 5.3. At each iteration, we sample new object and part assignments (o_{ji}, z_{ji}) for some feature, and then update our estimate of the parameters' posterior mode. As summarized in Alg. 6.2, we again use incremental EM updates [225] to reduce the cost of each iteration to $\mathcal{O}(LK)$ operations. See Sec. 5.3.5 for further discussion of this approximation.

■ 6.1.3 Street and Office Scenes

To evaluate the contextual scene model of Fig. 6.1, we perform experiments with the two datasets of visual scenes depicted in Fig. 6.2. The first set contains 613 street scenes depicting four "objects": buildings, cars (side views), roads, and trees. To align with the assumptions underlying our 2D scene model, images were normalized so that cars appear at comparable scales. As shown in Fig. 6.2, some of these street scenes have labels for all four categories, while others are only partially segmented. Note that it is straightforward to incorporate such semi–supervised training data into the Gibbs

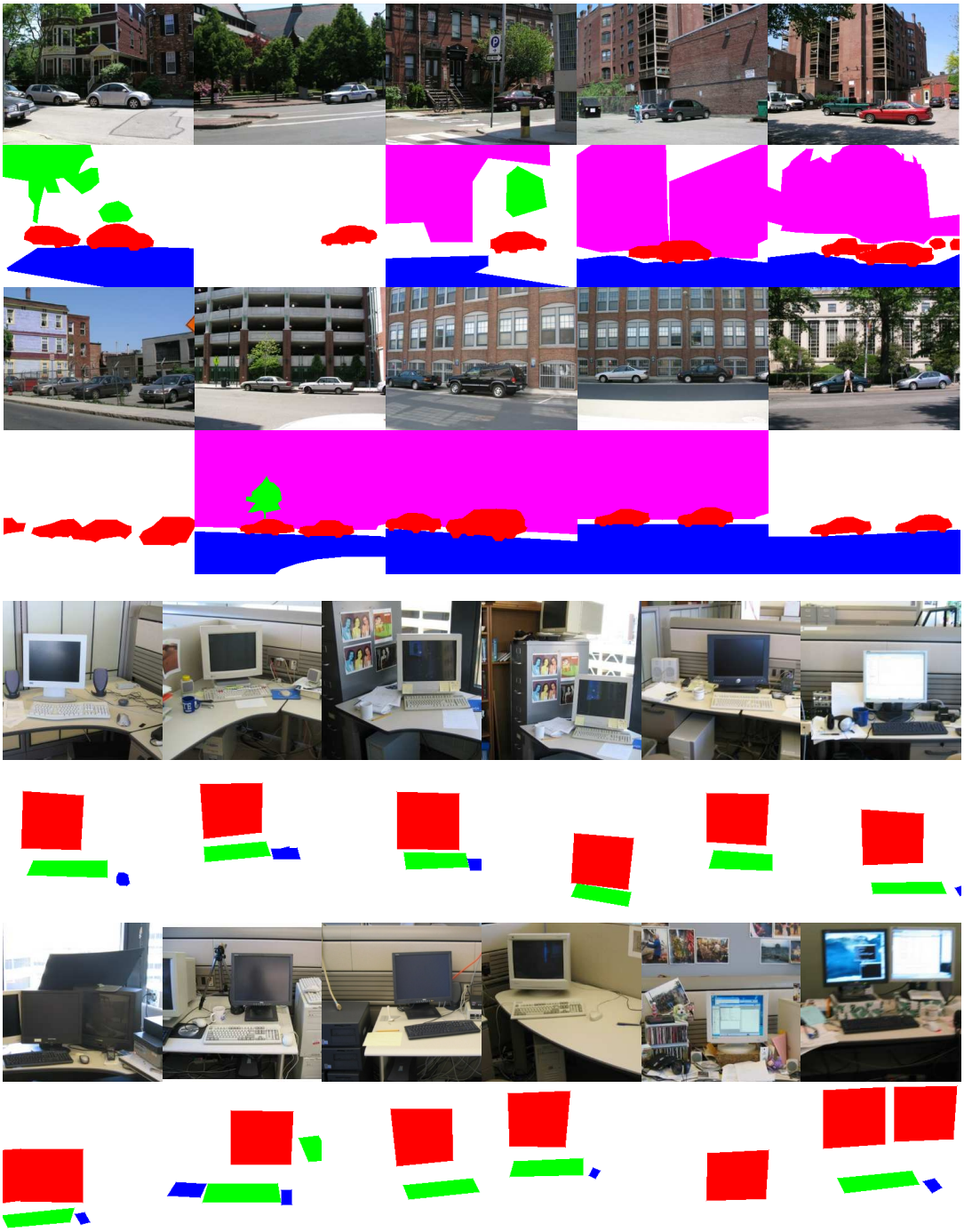


Figure 6.2. Scale–normalized images used to evaluate two–dimensional models for visual scenes. *Top:* Ten of 613 images from a partially labeled dataset of street scenes, and segmented regions corresponding to cars (red), buildings (magenta), roads (blue), and trees (green). *Bottom:* Twelve of 315 images from a fully labeled dataset of office scenes, and segmented regions corresponding to computer screens (red), keyboards (green), and mice (blue). Note the large variability in the image positions at which objects are observed, and in the number of instances of each object. Images available through the MIT-CSAIL Database of Objects and Scenes [299, 300].

sampler. In particular, we use these manual segmentations to fix the object category assignments o_{ji} of labeled features. For unlabeled features, object assignments are left unconstrained, and sampled as described in the preceding section.

The second dataset illustrated in Fig. 6.2 includes 315 pictures of office scenes containing four objects: computer screens (frontal views), keyboards, mice, and background clutter. In this case, images were normalized so that computer screens appeared at comparable scales. Also, all object instances were identified, so that the object assignment o_{ji} for every feature is fixed during training. For both datasets, we represent training and test images by interest regions extracted using the three operators described in Sec. 5.1.1. We then characterize each region's appearance by a vector quantized SIFT descriptor [188]. Each dataset used a separate appearance dictionary, which after expansion to encode region shape (see Sec. 5.1.2) contained $W = 1{,}600$ visual words.

Learning Part–Based Scene Models

For both datasets, we learn model parameters using the auxiliary variable Gibbs sampler of Alg. 6.1. The performance of the marginalized Gibbs sampler of Alg. 6.2 is similar, but it is slower due to the computational overhead of performing an incremental EM update after each feature reassignment. For training, we used 400 street scenes and 250 office scenes; the remaining images then provide a segmented test set. To estimate model parameters, we first ran the Gibbs sampler for 500 iterations using only the training images. Each scene model employed thirty shared parts, and Dirichlet precision parameters set as $\gamma = 4$, $\alpha = 15$ via cross–validation. The sampler assumed normal–inverse–Wishart priors H_v which weakly favored parts covering 10% of the image range, and appearance priors $H_w \sim \mathrm{Dir}(W/10)$ biased towards sparse distributions.

Figs. 6.3 and 6.4 illustrate the part–based models that were learned for street and office scenes. Although objects share a common set of parts within each scene model, we can approximately count the number of parts used by each object by thresholding the posterior part distributions $\boldsymbol{\pi}_\ell$ (see Fig. 6.1). For street scenes, cars are allocated roughly four parts, while buildings and roads use large numbers of parts to uniformly tile regions which correspond to their typical size. Several parts are shared between the tree and building categories, presumably due to the many training images in which buildings are only partially occluded by foliage.

The office scene model describes computer screens with roughly ten parts, which primarily align with their edges and corners. Due to their smaller size, keyboards are described by five parts, and mice by two. The background clutter category then uses several parts, which move little from scene to scene, to distribute features across the full image. Most parts are unshared, although the screen and keyboard categories reuse a few parts to describe similar edge–like features. Fig. 6.4 also illustrates the contextual relationships learned by this scene model. Intuitively, the keyboard is typically located beneath the monitor, and the mouse to the keyboard's right.

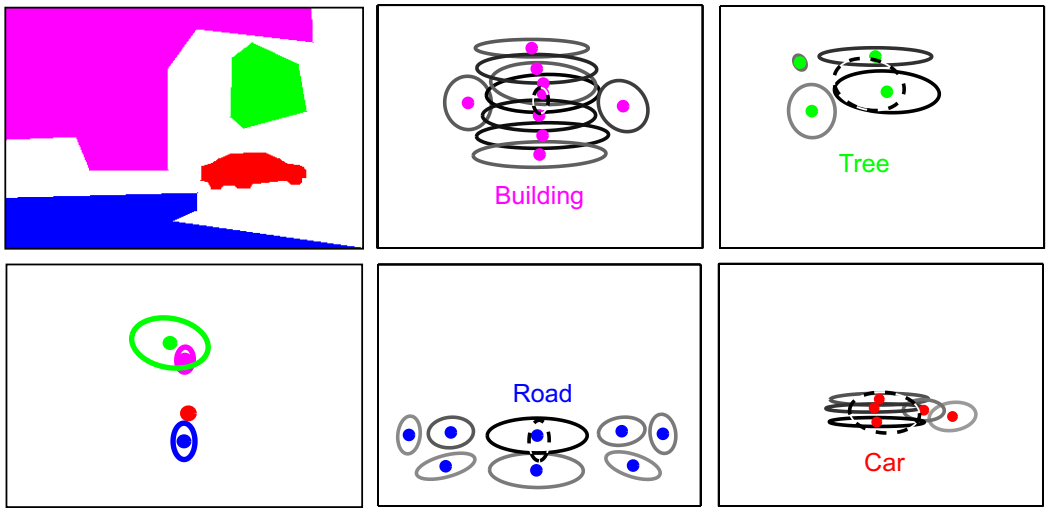


Figure 6.3. Learned contextual, fixed–order model of street scenes containing four objects. *Left:* Gaussian distributions over building (magenta), road (blue), and tree (green) positions conditioned on the car's location (red). *Right:* Parts (solid) generating at least 5% of each category's features, with intensity proportional to probability. Parts are tranlated by that object's mean position, while the dashed ellipses indicate each object's marginal transformation covariance.

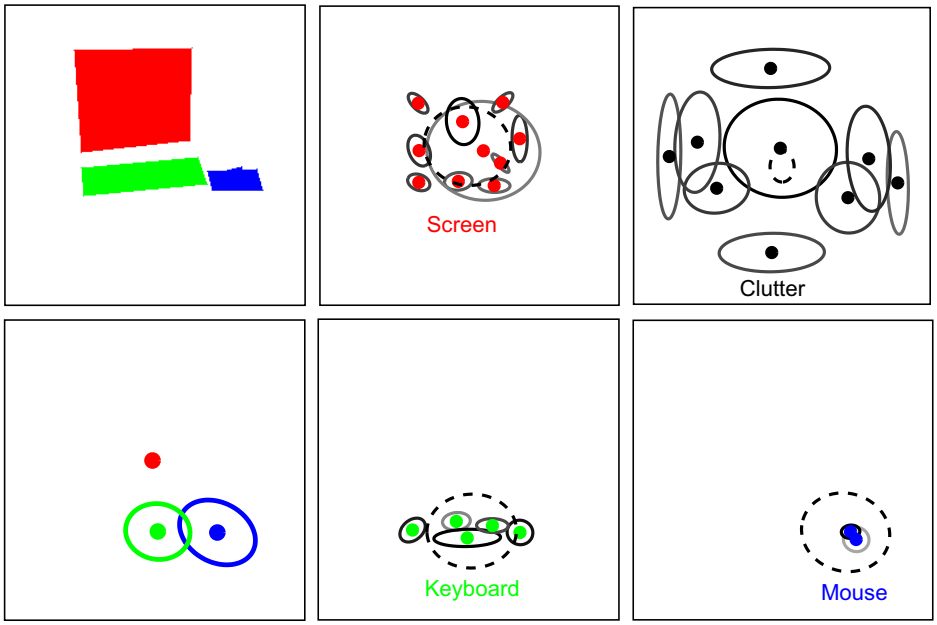


Figure 6.4. Learned contextual, fixed–order model of office scenes containing three objects and background clutter. *Left:* Gaussian distributions over keyboard (green) and mouse (blue) positions conditioned on the computer screen's location (red). *Right:* Parts (solid) generating at least 5% of each category's features, with intensity proportional to probability. Parts are translated by that object's mean position, while the dashed ellipses indicate each object's marginal transformation covariance.

Segmentation of Novel Visual Scenes

To analyze test images, we fix the part and object assignments corresponding to the final Gibbs sampling iteration on the training set. To avoid local optima, we then run the test image Gibbs sampler for 20 iterations from each of ten different random initializations. Given reference transformations sampled in this fashion, we used the conditional likelihoods of Sec. 6.1.1 to estimate the posterior probability that test features were generated by each candidate object category. Analogously to the approach of Sec. 5.3.6, we then averaged the probabilities corresponding to different sampled transformations to determine an overall segmentation.

In Fig. 6.5, we illustrate feature segmentations for several typical test street scenes, and transformed parts corresponding to the highest likelihood iteration of the Gibbs sampler. Segmentations of building and road features are typically very accurate, as the contextual model learns the vertical layering inherent in street scenes. Note that a number of test images violate our parametric model's assumption that a single reference transformation explains all of each object's observed features. To partially correct for this, the model learns horizontally elongated car parts which extend beyond an average car. Although this allows better segmentations for pairs of adjacent cars, nearby background clutter is often mislabeled. In images containing widely separated cars, one car is usually missed entirely. The assumption that every image contains one tree is also problematic, since some features are typically classified as foliage even when no trees are present.

Figure 6.6 shows similar segmentation results for office scenes. Because most test images do indeed contain a single computer screen, the model's use of a fixed, parametric transformation causes fewer errors for office scenes. Contextual information is especially important for detecting computer mice (see Fig. 6.6). Very few features are detected in the region corresponding to the mouse, and they are not very distinctive. However, as the screen can be reliably located, this provides a strong constraint on the expected location of the mouse. In fact, for test images in which no mouse is present the system often hallucinates one in other appropriately positioned clutter.

For comparison, Fig. 6.7 shows segmentation results for a "bag of features" model. This model was derived from the full contextual model of Fig. 6.1 by ignoring feature positions, and thus the latent reference transformation. As confirmed by the ROC curves of Fig. 6.8, the appearance–only model is substantially less accurate for all categories except trees. For street scenes, the full, position–based model recognizes car features reasonably well despite employing a single reference position, and roads are very accurately segmented. For office scenes, it exploits contextual relationships to detect mice and keyboards with accuracy comparable to the more visually distinctive computer screens. These improvements over the bag of features model highlight the importance of spatial structure in visual scene understanding.

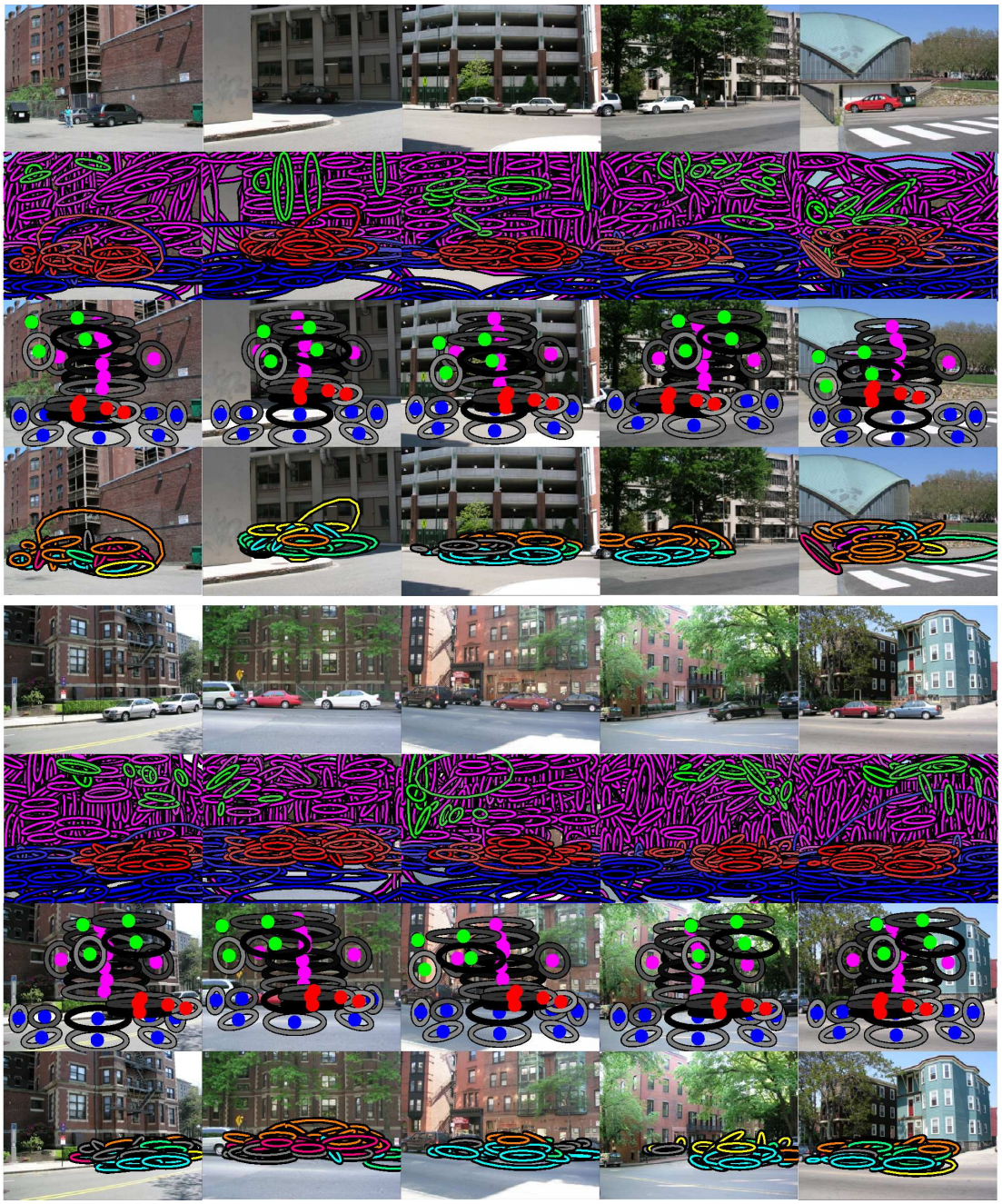


Figure 6.5. Feature segmentations produced by a contextual, fixed–order model of street scenes containing cars (red), buildings (magenta), roads (blue), and trees (green). Each block of four rows depicts five test images (first row). Feature segmentations (second row) assign each feature to the object category with the highest posterior probability. We also show model parts translated according to each image's reference transformation (third row), and color–coded assignments of features to the different parts associated with cars (fourth row).

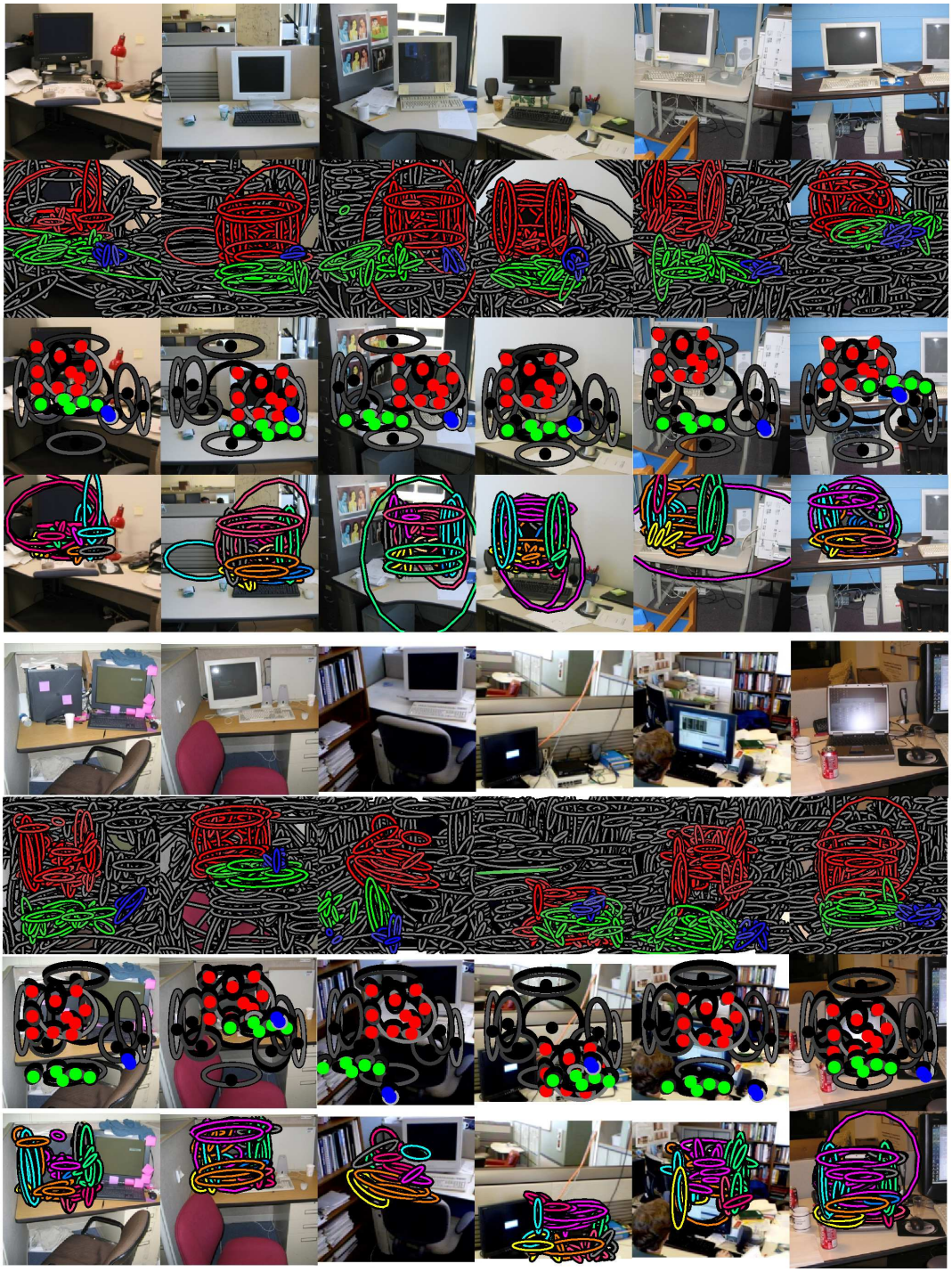


Figure 6.6. Feature segmentations produced by a contextual, fixed–order model of office scenes containing computer screens (red), keyboards (green), mice (blue), and background clutter (gray). Each block of four rows depicts six test images (first row). Feature segmentations (second row) assign each feature to the object category with the highest posterior probability. We also show model parts translated according to each image's reference transformation (third row), and color–coded assignments of features to the different parts associated with computer screens (fourth row).

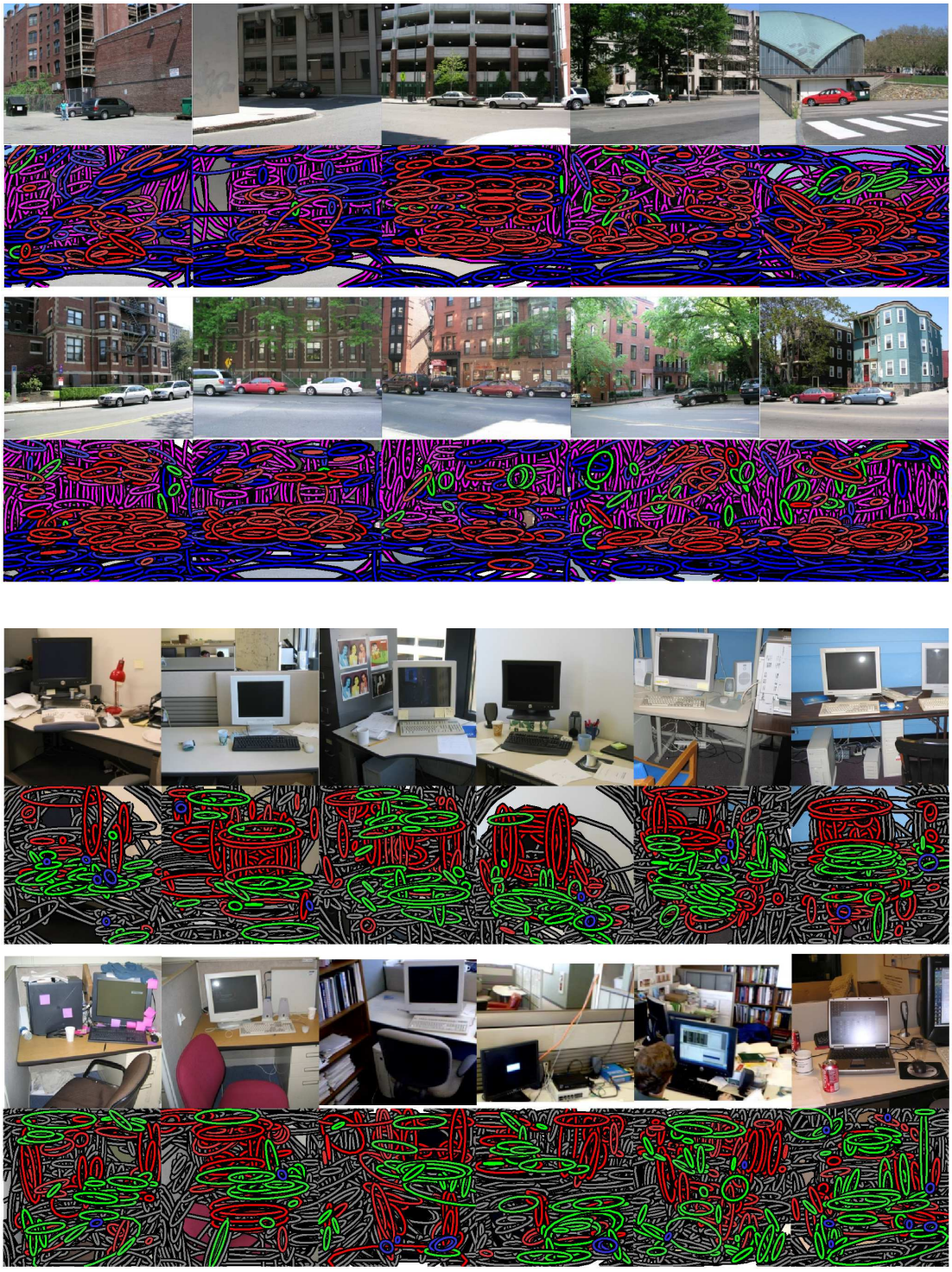


Figure 6.7. Segmentations produced by a "bag of features" model which neglects spatial structure, and learns histogram models of feature appearance. *Top rows:* Street scenes containing cars (red), buildings (magenta), roads (blue), and trees (green). *Bottom rows:* Office scenes containing computer screens (red), keyboards (green), mice (blue), and background clutter (gray).

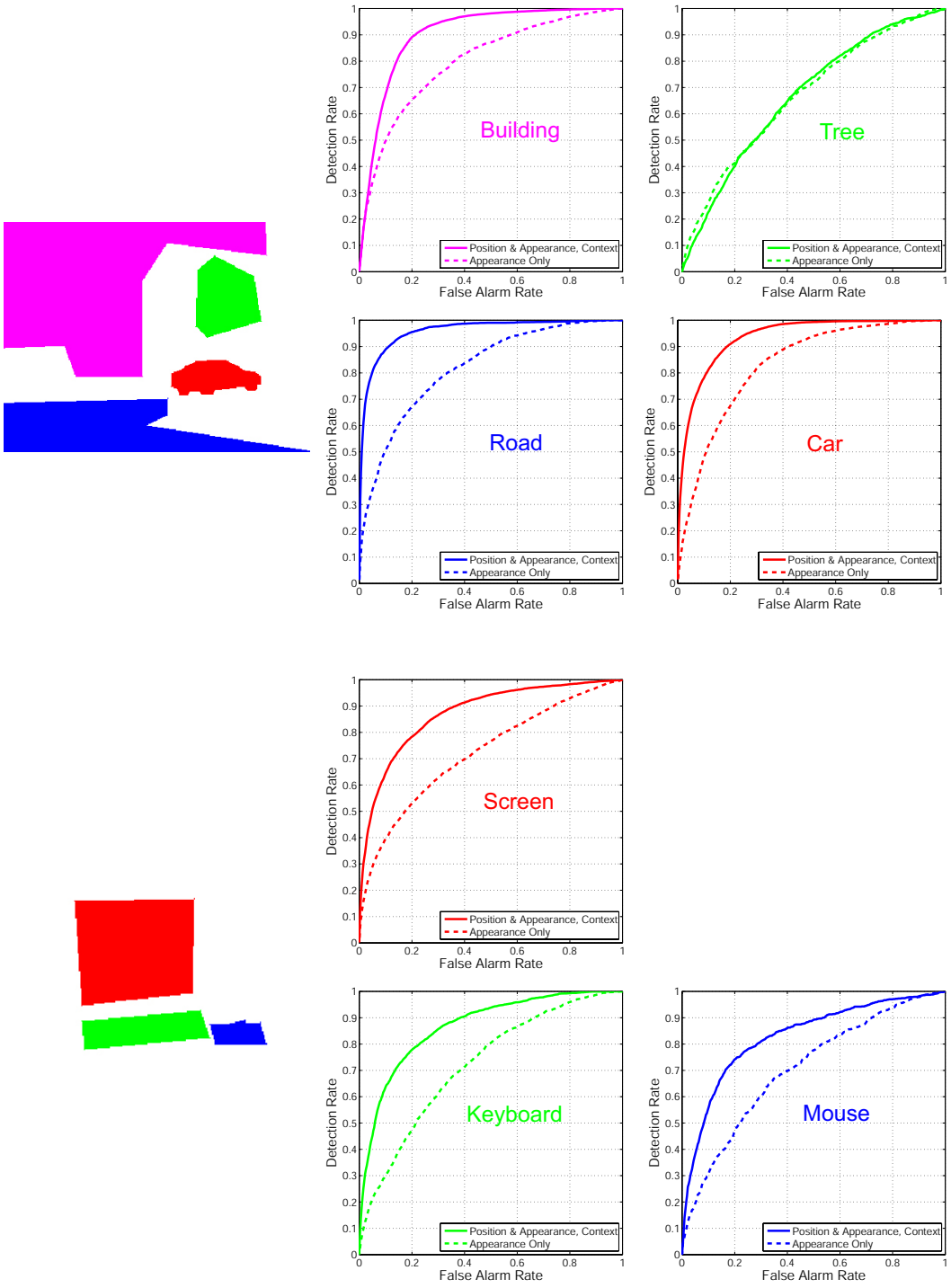


Figure 6.8. ROC curves summarizing segmentation performance for the features composing street scenes (top) and office scenes (bottom). We compare a bag of features model based solely only local appearance (dashed lines) to the full, fixed–order scene model (solid lines) of Fig. 6.1.

6.2 Transformed Dirichlet Processes

As demonstrated by the preceding experimental results, the fixed–order visual scene model of Fig. 6.1 provides a characterization of spatial structure which improves on "bag of features" approximations. Its incorporation of contextual relationships is particularly effective for small, visually indistinct objects. However, the assumption that each image depicts a known set of objects is obviously unrealistic. In particular, the results of Sec. 6.1.3 demonstrate the fixed–order model's inability to detect multiple instances of structured objects like cars and computer screens.

To address these limitations, we now develop a family of hierarchical scene models based on the Dirichlet process [28, 76, 83, 254]. In Sec. 5.5, we adapted the hierarchical Dirichlet process (HDP) [289] to allow uncertainty in the number of parts underlying a set of object categories. Extending this approach, we now develop models which capture the uncertain number of object *instances* depicted in each image. We begin by describing a *transformed Dirichlet process* (TDP), which generalizes the HDP by applying a random *set* of transformations to each global cluster. Sec. 6.3 then uses the TDP to develop robust nonparametric models for visual scenes.

6.2.1 Sharing Transformations via Stick–Breaking Processes

To simplify our presentation of the TDP, we revisit the hierarchical clustering framework described in Sec. 2.5.4. Let $\theta \in \Theta$ denote the parameters defining a cluster or topic distribution $F(\theta)$. To more flexibly share these clusters among related groups of data, we consider a family of transformations $\tau(\theta; \rho)$ of these parameters, indexed by $\rho \in \wp$. See Sec. 5.2 for examples of transformations adapted to clusters of spatial data.

The TDP is derived by considering *distributions over transformations* $\rho \sim Q(\varphi)$, indexed by $\varphi \in \Phi$. For example, if ρ is a vector defining a translation as in Sec. 5.2.1, φ could parameterize a family of Gaussian distributions $Q(\cdot)$. We equivalently denote the transformation density corresponding to this distribution by $q(\rho \mid \varphi)$. Finally, to define consistent hierarchical models, we also consider a prior measure R on the space of transformation distributions Φ. Similarly, we let H denote a prior measure on the space of cluster parameters Θ.

We begin by augmenting the Dirichlet process' stick–breaking construction [254], as in eq. (2.198), to define a global measure describing both cluster parameters θ and transformations ρ:

$$G_0(\theta, \rho) = \sum_{\ell=1}^{\infty} \beta_\ell \, \delta(\theta, \theta_\ell) q(\rho \mid \varphi_\ell) \qquad \begin{aligned} \boldsymbol{\beta} &\sim \mathrm{GEM}(\gamma) \\ \theta_\ell &\sim H \\ \varphi_\ell &\sim R \end{aligned} \tag{6.23}$$

Note that each cluster θ_ℓ is associated with a different, continuous transformation distribution $Q(\varphi_\ell)$. As in the HDP, we then independently sample a measure $G_j \sim \mathrm{DP}(\alpha, G_0)$ for each of J related groups of data. Because samples from Dirichlet processes are dis-

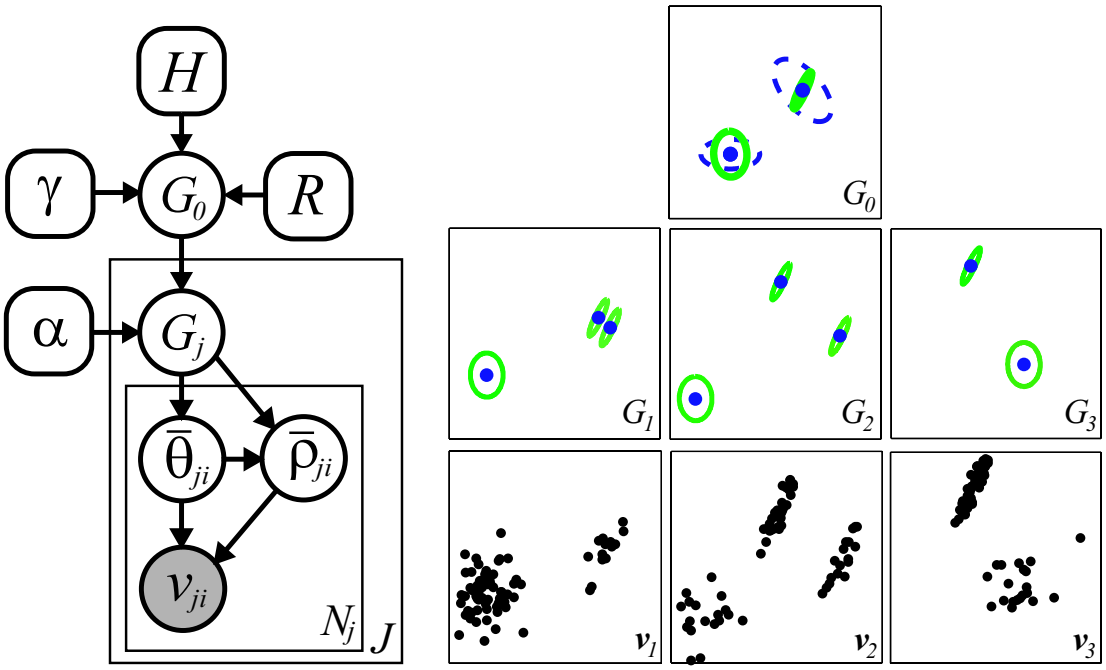


Figure 6.9. Directed graphical representation of a transformed Dirichlet process (TDP) mixture model. Each group is assigned an infinite discrete distribution $G_j \sim \mathrm{DP}(\alpha, G_0)$, which is sampled from a global distribution $G_0(\theta, \rho)$ over transformations ρ of cluster parameters θ. $(\bar{\theta}_{ji}, \bar{\rho}_{ji}) \sim G_j$ are then the *transformed* cluster parameters which generate an observation $v_{ji} \sim F\big(\tau(\bar{\theta}_{ji}; \bar{\rho}_{ji})\big)$. We illustrate the TDP with a model of two–dimensional spatial data. G_0 is composed of a collection of 2D Gaussian distributions (green covariance ellipses), and a corresponding Gaussian prior (blue dashed ellipses) on translations of each cluster. For each of three groups, we show transformed Gaussian mixtures G_j which make a random set of copies of each global cluster, and resulting observations $\mathbf{v}_j$.

crete with probability one (see Thm. 2.5.3), the joint measure for group j equals

$$G_j(\theta, \rho) = \sum_{t=1}^{\infty} \widetilde{\pi}_{jt} \delta(\theta, \widetilde{\theta}_{jt}) \delta(\rho, \rho_{jt}) \qquad \begin{aligned} \widetilde{\boldsymbol{\pi}}_j &\sim \mathrm{GEM}(\alpha) \\ (\widetilde{\theta}_{jt}, \rho_{jt}) &\sim G_0 \end{aligned} \tag{6.24}$$

Each *local* cluster in group j has parameters $\widetilde{\theta}_{jt}$, and a corresponding transformation ρ_{jt}, derived from some global cluster. Anticipating our later identification of global clusters with object categories, we let $o_{jt} \sim \boldsymbol{\beta}$ indicate this correspondence, so that $\widetilde{\theta}_{jt} = \theta_{o_{jt}}$.

As summarized in the graph of Fig. 6.9, each observation v_{ji} in group j is then independently sampled according to the *transformed* parameters of some local cluster:

$$\begin{aligned} (\bar{\theta}_{ji}, \bar{\rho}_{ji}) &\sim G_j \\ v_{ji} &\sim F\big(\tau(\bar{\theta}_{ji}; \bar{\rho}_{ji})\big) \end{aligned} \tag{6.25}$$

For computational convenience, we typically define $F(\theta)$ to be an appropriate exponential family, and H a corresponding conjugate prior. As with standard mixtures,

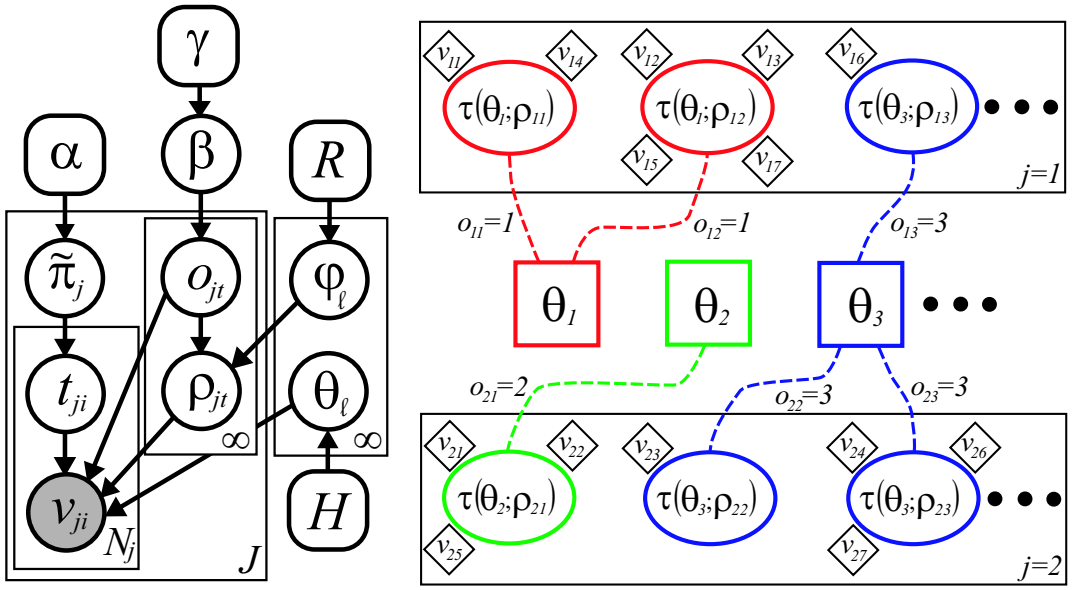


Figure 6.10. Chinese restaurant franchise representation of the TDP model of Fig. 6.9. *Left:* Global cluster parameters are assigned independent priors $\theta_k \sim H$, and reused by groups with frequencies $\boldsymbol{\beta} \sim \text{GEM}(\gamma)$. Each group j has infinitely many local clusters (tables) t, which are associated with a transformation $\rho_{jt} \sim Q(\varphi_{o_{jt}})$ of some global cluster (dish) $o_{jt} \sim \boldsymbol{\beta}$. Observations (customers) v_{ji} are independently assigned to some table $t_{ji} \sim \widetilde{\boldsymbol{\pi}}_j$, and thus indirectly associated with that table's transformed (seasoned) global cluster $\tau(\theta_{z_{ji}}; \rho_{jt_{ji}})$, where $z_{ji} = o_{jt_{ji}}$. *Right:* Example in which a franchise menu with dishes θ_ℓ (squares, center) is shared among tables (ovals, top and bottom) in two different restaurants (groups). All customers (diamonds) seated at a given table share the same dish (global cluster parameter), which is uniquely seasoned (transformed) each time it is ordered.

eq. (6.25) can be equivalently expressed via a discrete variable t_{ji} indicating the transformed cluster associated with the i^{th} observation:

$$\begin{aligned} t_{ji} &\sim \widetilde{\boldsymbol{\pi}}_j \\ v_{ji} &\sim F\big(\tau\big(\widetilde{\theta}_{jt_{ji}}; \rho_{jt_{ji}}\big)\big) \end{aligned} \tag{6.26}$$

Fig. 6.10 shows an alternative graphical representation of the TDP, based on these explicit assignments of observations to local clusters, and local clusters to transformations of particular global clusters.

As discussed in Sec. 2.5.4, the HDP models groups by reusing an *identical* set of global clusters in different proportions. In contrast, the TDP modifies the shared, global clusters via a set of group–specific stochastic transformations. As we demonstrate in later sections, this allows us to model richer datasets in which only a subset of the global clusters' properties are naturally shared.

6.2.2 Characterizing Transformed Distributions

Recall that the global measure G_0 underlying the TDP (see eq. (6.23)) defines a discrete distribution over cluster parameters θ_ℓ. In contrast, the distributions $Q(\varphi_\ell)$ associated with transformations of these clusters are continuous. Each group j will thus create many different copies $\widetilde{\theta}_{jt}$ of a global cluster θ_ℓ, but associate each with a *different* transformation ρ_{jt}. Aggregating the probabilities assigned to these copies, we can directly express G_j in terms of the distinct global cluster parameters:

$$G_j(\theta,\rho) = \sum_{\ell=1}^{\infty} \pi_{j\ell}\delta(\theta,\theta_\ell)\left[\sum_{s=1}^{\infty}\omega_{j\ell s}\delta(\rho,\check{\rho}_{j\ell s})\right] \qquad \pi_{j\ell} = \sum_{t|o_{jt}=\ell}\widetilde{\pi}_{jt} \tag{6.27}$$

In this expression, we have grouped the infinite set of transformations which group j associates with each global cluster ℓ:

$$\{\check{\rho}_{j\ell s} \mid s = 1,2,\ldots\} = \{\rho_{jt} \mid o_{jt} = \ell\} \tag{6.28}$$

The weights $\boldsymbol{\omega}_{j\ell} = (\omega_{j\ell 1}, \omega_{j\ell 2}, \ldots)$ then equal the proportion of the total cluster probability $\pi_{j\ell}$ contributed by each transformed cluster $\widetilde{\pi}_{jt}$ satisfying $o_{jt} = \ell$.

The following proposition provides a direct probabilistic characterization of the transformed measures arising in the TDP.

Proposition 6.2.1. *Let $G_0(\theta,\rho)$ be a global measure constructed as in eq. (6.23), and $G_j(\theta,\rho) \sim \mathrm{DP}(\alpha, G_0(\theta,\rho))$ be expressed as in eq. (6.27). The marginal distributions of G_j with respect to parameters and transformations then also follow Dirichlet processes:*

$$G_j(\theta) \sim \mathrm{DP}(\alpha, G_0(\theta)) \qquad G_0(\theta) = \sum_{\ell=1}^{\infty}\beta_\ell\delta(\theta,\theta_\ell) \tag{6.29}$$

$$G_j(\rho) \sim \mathrm{DP}(\alpha, G_0(\rho)) \qquad G_0(\rho) = \sum_{\ell=1}^{\infty}\beta_\ell Q(\varphi_\ell) \tag{6.30}$$

Proof. Consider the base measure $G_0(\theta,\rho)$ of eq. (6.23), where $\theta \in \Theta$ and $\rho \in \wp$. Let $(\Theta_1,\ldots,\Theta_K)$ be any finite, measurable partition of Θ. Because $G_j(\theta,\rho)$ follows a Dirichlet process, Thm. 2.5.1 then implies that

$$\begin{aligned}(G_j(\Theta_1,\wp),\ldots,G_j(\Theta_K,\wp)) &\sim \mathrm{Dir}(\alpha G_0(\Theta_1,\wp),\ldots,\alpha G_0(\Theta_K,\wp))\\ (G_j(\Theta_1),\ldots,G_j(\Theta_K)) &\sim \mathrm{Dir}(\alpha G_0(\Theta_1),\ldots,\alpha G_0(\Theta_K))\end{aligned}$$

The second line follows from the definition of a marginal distribution $G_0(\theta) \triangleq G_0(\theta,\wp)$. Again invoking Thm. 2.5.1, this expression shows that $G_j(\theta) \sim \mathrm{DP}(\alpha, G_0(\theta))$, establishing eq. (6.29). Eq. (6.30) then follows from a complementary argument using measurable partitions $(\wp_1,\ldots,\wp_K)$ of $\wp$. □

Examining eq. (6.29), we see that the TDP induces discrete marginal distributions on parameters exactly like those arising in the HDP [289]. The HDP can thus be seen as a limiting case of the TDP in which transformations are insignificant or degenerate.

Of course, the TDP is intended for applications in which transformations facilitate information transfer. The following proposition considers the dependencies between parameters and transformations induced by the TDP's hierarchical construction.

Proposition 6.2.2. *Let $G_0(\theta, \rho)$ be a global measure constructed as in eq. (6.23), and $G_j(\theta, \rho) \sim \mathrm{DP}(\alpha, G_0(\theta, \rho))$ be expressed as in eq. (6.27). Assume that Θ is a Hausdorff space. Given any discrete parameter θ_ℓ from the global measure, we then have*

$$G_j(\rho \mid \theta = \theta_\ell) \sim \mathrm{DP}(\alpha\beta_\ell, Q(\varphi_\ell)) \tag{6.31}$$

The weights associated with different transformations of θ_ℓ thus follow a stick–breaking process, so that $\boldsymbol{\omega}_{j\ell} \sim \mathrm{GEM}(\alpha\beta_\ell)$.

Proof. Let Δ_ℓ denote a set containing θ_ℓ, and excluding $\{\theta_{\ell'} \mid \ell' \neq \ell\}$. The existence of Δ_ℓ is guaranteed by the Hausdorff condition. Define $\bar{\Delta}_\ell = \Theta \setminus \Delta_\ell$, and let $(\wp_1, \ldots, \wp_K)$ be any partition of the transformation space $\wp$. From Thm. 2.5.1, we then have

$$\begin{aligned}
\big(G_j(\Delta_\ell, \wp_1),.., G_j(\Delta_\ell, \wp_K), G_j(\bar{\Delta}_\ell, \wp)\big) &\sim \mathrm{Dir}\big(\alpha G_0(\Delta_\ell, \wp_1),.., \alpha G_0(\Delta_\ell, \wp_K), \alpha G_0(\bar{\Delta}_\ell, \wp)\big) \\
\left(\pi_{j\ell} \sum_{s \in \mathcal{S}_1} \omega_{j\ell s},.., \pi_{j\ell} \sum_{s \in \mathcal{S}_K} \omega_{j\ell s}, 1 - \pi_{j\ell}\right) &\sim \mathrm{Dir}(\alpha\beta_\ell Q(\wp_1; \varphi_\ell),.., \alpha\beta_\ell Q(\wp_K; \varphi_\ell), \alpha(1-\beta_\ell))
\end{aligned}$$

In this expression, $\mathcal{S}_k = \{s \mid \check{\rho}_{j\ell s} \in \wp_k\}$, the discrete subset of the transformations of θ_ℓ contained within $\wp_k$. Marginalizing the last element of this probability vector via the formulas of Sec. 2.1.3, we then have

$$\begin{aligned}
\frac{1}{\pi_{j\ell}}\left(\pi_{j\ell} \sum_{s \in \mathcal{S}_1} \omega_{j\ell s}, \ldots, \pi_{j\ell} \sum_{s \in \mathcal{S}_K} \omega_{j\ell s}\right) &\sim \mathrm{Dir}(\alpha\beta_\ell Q(\wp_1; \varphi_\ell), \ldots, \alpha\beta_\ell Q(\wp_K; \varphi_\ell)) \\
G_j(\rho \mid \theta = \theta_\ell) &\sim \mathrm{DP}(\alpha\beta_\ell, Q(\varphi_\ell))
\end{aligned}$$

The last line again follows from Thm. 2.5.1, and the one–to–one correspondence between partitions $\{\wp_k\}_{k=1}^K$ of the transformation space and discrete subsets $\{\mathcal{S}_k\}_{k=1}^K$ of the sampled transformations $\check{\rho}_{j\ell s}$. A related argument was used to establish properties of the hierarchical Dirichlet process [289]. □

Recall that Dirichlet processes approach the base measure by assigning roughly uniform weights to a large number of discrete samples (see Sec. 2.5.2). This result shows that parameters θ_ℓ with small weight β_ℓ will also have greater variability in their transformation distributions, because (on average) they are allocated fewer samples. Intuitively, the concentration parameters $\{\alpha\beta_\ell\}_{\ell=1}^\infty$ associated with transformations of all global clusters sum to α, the overall concentration of G_j around G_0.

As discussed in Sec. 2.5.4, the HDP is a special case of a very general *dependent Dirichlet process* (DDP) [191] framework. Viewing cluster parameters and transformations as one augmented parameter vector, TDPs are also a special case of the DDP framework. However, this perspective obscures the interplay between the discrete and continuous portions of the TDP base measure, and the manner in which transformations modify parameters to achieve a very rich class of dependencies.

6.2.3 Learning via Gibbs Sampling

To develop computational methods for learning transformed Dirichlet processes, we consider a generalization of the HDP's Chinese restaurant franchise representation [289]. As in the HDP analogy described in Sec. 2.5.4, customers (observations) v_{ji} sit at tables t_{ji} according to the clustering bias of eq. (2.203), and new tables choose dishes according to their popularity across the franchise (eq. (2.204)). As illustrated in Fig. 6.10, however, the dish (parameter) $\theta_{o_{jt}}$ at table t is now seasoned (transformed) according to $\rho_{jt} \sim Q(\varphi_{o_{jt}})$. Each time a dish is ordered, the recipe is seasoned differently, and every dish θ_ℓ has different typical seasonings $Q(\varphi_\ell)$.

Using this representation, we extend the HDP Chinese restaurant franchise Gibbs sampler detailed in [289], which is in turn based on standard methods for DP mixture models [222]. In Sec. 5.5.1, we adapted this sampler to our nonparametric object appearance model. While that model associated a single reference transformation with each image, the TDP instead describes groups via a *set* of randomly transformed clusters. The proposed Gibbs sampler thus has three sets of state variables: assignments $\mathbf{t}$ of observations to tables (transformed clusters), assignments $\mathbf{o}$ of tables to global clusters, and the transformations $\boldsymbol{\rho}$ associated with each table. Given these variables, the weights associated with the global (eq. (6.23)) and local (eq. (6.24)) cluster distributions can be analytically marginalized.

To avoid cumbersome notation, we let $z_{ji} = o_{jt_{ji}}$ denote the global cluster associated with observation v_{ji}. Note that z_{ji} is uniquely determined by that observation's table assignment $t_{ji} = t$, and the corresponding table's cluster assignment o_{jt}. As in Sec. 5.2, we assume that the chosen family of parameter transformations has a complementary *data transformation* defined so that $f(v \mid \tau(\theta;\rho)) \propto f(\tilde{\tau}(v;\rho) \mid \theta)$.

Table Assignment Resampling

We first consider the posterior distribution of the table assignment t_{ji} for observation v_{ji}, given all other state variables. Letting $\mathbf{t}_{\setminus ji}$ denote all table assignments excluding t_{ji}, the Markov properties of the TDP (see Fig. 6.10) imply that

$$p(t_{ji} \mid \mathbf{t}_{\setminus ji}, \mathbf{o}, \mathbf{v}, \boldsymbol{\rho}) \propto p(t_{ji} \mid \mathbf{t}_{\setminus ji})\, p(v_{ji} \mid \mathbf{t}, \mathbf{o}, \mathbf{v}_{\setminus ji}, \boldsymbol{\rho}) \tag{6.32}$$

Let N_{jt}^{-i} denote the number of observations assigned to each of the T_j tables which $\mathbf{t}_{\setminus ji}$ currently instantiates in group j. The clustering bias induced by the Chinese restaurant

process (see eq. (2.203)) then implies that

$$p(t_{ji} \mid \mathbf{t}_{\setminus ji}) \propto \sum_{t=1}^{T_j} N_{jt}^{-i} \delta(t_{ji}, t) + \alpha \delta(t_{ji}, \bar{t}) \tag{6.33}$$

where $\bar{t}$ denotes an assignment to a new, previously unoccupied table. For existing tables, the likelihood term of eq. (6.32) depends on those observations currently assigned to the same global, shared cluster:

$$\begin{aligned} p(v_{ji} \mid z_{ji} = \ell, \mathbf{t}_{\setminus ji}, \mathbf{o}, \mathbf{v}_{\setminus ji}, \boldsymbol{\rho}) &= \int_\Theta h(\theta_\ell) \prod_{j'i' \mid z_{j't_{j'i'}} = \ell} f\big(v_{j'i'} \mid \tau(\theta_\ell; \rho_{j't_{j'i'}})\big) \, d\theta_\ell \\ &\propto \int_\Theta h(\theta_\ell) \prod_{j'i' \mid z_{j't_{j'i'}} = \ell} f\big(\tilde{\tau}\big(v_{j'i'}; \rho_{j't_{j'i'}}\big) \mid \theta_\ell\big) \, d\theta_\ell \end{aligned} \tag{6.34}$$

Here, we have used eq. (5.1) to reexpress the posterior distribution of θ_ℓ in terms of a transformed dataset. If $H(\theta)$ is conjugate to $F(\theta)$, this likelihood may be evaluated in closed form (see Prop. 2.1.4). For efficiency, we base this computation on cached sufficient statistics of the *relative* offset of each observation $v_{j'i'}$ from its associated table's transformation $\rho_{j't_{j'i'}}$.

For new tables $\bar{t}$, we improve sampling efficiency by integrating over potential assignments $o_{j\bar{t}}$ to global parts. Because the transformations associated with these tables are uncertain, exact evaluation of Rao–Blackwellized predictive likelihoods (as in eq. (6.34)) is intractable. We thus begin by using the current transformations and assignments for other observations to sample auxiliary parameters [222] for each of the L currently instantiated global clusters:

$$\hat{\theta}_\ell \sim h(\theta_\ell) \prod_{j'i' \mid z_{j't_{j'i'}} = \ell} f\big(\tilde{\tau}\big(v_{j'i'}; \rho_{j't_{j'i'}}\big) \mid \theta_\ell\big) \tag{6.35}$$

$$\hat{\varphi}_\ell \sim r(\varphi_\ell) \prod_{j't' \mid o_{j't'} = \ell} q\big(\rho_{j't'} \mid \varphi_\ell\big) \tag{6.36}$$

As in Sec. 5.5.1, we typically approximate these auxiliary samples by the corresponding posterior modes. The observation likelihood for new tables then equals

$$p(v_{ji} \mid t_{ji} = \bar{t}, \mathbf{o}, \widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\varphi}}) \propto \sum_\ell p\big(o_{j\bar{t}} = \ell \mid \mathbf{o}\big) \int_\wp f(\tilde{\tau}(v_{ji}; \rho) \mid \hat{\theta}_\ell) \, q(\rho \mid \hat{\varphi}_\ell) \, d\rho \tag{6.37}$$

From eq. (2.204), the probability of each of the L existing global clusters depends on the number of other tables M_ℓ which $\mathbf{o}$ assigns to that part:

$$p\big(o_{j\bar{t}} \mid \mathbf{o}\big) \propto \sum_{\ell=1}^{L} M_\ell \delta(o_{j\bar{t}}, \ell) + \gamma \delta(o_{j\bar{t}}, \bar{\ell}) \tag{6.38}$$

In this expression, $\bar{\ell}$ indicates a potential new global cluster, to which no tables are currently assigned. As in the DP mixture sampler of Alg. 2.3, our implementation maintains a dynamically resized list of likelihood statistics for those clusters associated with at least one observation.

To sample according to eq. (6.32), we first evaluate the transformed likelihood of eq. (6.34) for all existing tables, and the integral likelihood of eq. (6.37) for all global clusters. In the simplest case, suppose that $\hat{\theta}_\ell = (\hat{\mu}_\ell, \hat{\Lambda}_\ell)$ parameterizes a Gaussian distribution, and that $\hat{\varphi}_\ell = (\hat{\zeta}_\ell, \hat{\Upsilon}_\ell)$ defines a corresponding Gaussian prior on translations as in Sec. 5.2.1. The marginalized position likelihood then equals

$$\int_{\wp} \mathcal{N}(v_{ji} - \rho; \hat{\mu}_\ell, \hat{\Lambda}_\ell)\, \mathcal{N}(\rho; \hat{\zeta}_\ell, \hat{\Upsilon}_\ell)\, d\rho = \mathcal{N}(v_{ji}; \hat{\mu}_\ell + \hat{\zeta}_\ell, \hat{\Lambda}_\ell + \hat{\Upsilon}_\ell) \tag{6.39}$$

For more complex transformation families, further numerical or Monte Carlo approximations may be needed. Finally, combining these likelihoods with the Dirichlet process clustering biases of eqs. (6.33, 6.38), we sample a new table assignment t_{ji}, and if a new table $t_{ji} = \bar{t}$ is chosen, a corresponding cluster assignment $o_{j\bar{t}}$ and transformation $\rho_{j\bar{t}}$.

Global Cluster and Transformation Resampling

We now consider the assignments o_{jt} of tables to global clusters, and corresponding transformations ρ_{jt}, given fixed associations $\mathbf{t}$ between observations and tables. Although each group of observations j has infinitely many tables, we only explicitly resample variables associated with the T_j tables currently occupied by at least one observation ($N_{jt} > 0$).

For the TDP, the global cluster assignment o_{jt} and transformation ρ_{jt} associated with any particular table t are strongly dependent. In particular, to adequately explain the same set of data with a different cluster $\bar{o}_{jt}$, a complementary change of the transformation $\bar{\rho}_{jt}$ is typically needed. For this reason, we achieve *much* more rapid convergence via a blocked Gibbs sampler [9, 185, 246] which considers the joint distribution over both assignments and transformations:

$$p\big(o_{jt} = \ell, \rho_{jt} \mid \mathbf{o}_{\setminus jt}, \boldsymbol{\rho}_{\setminus jt}, \mathbf{t}, \widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\varphi}}\big) \propto p(\ell \mid \mathbf{o}_{\setminus jt})\, q(\rho_{jt} \mid \hat{\varphi}_\ell) \prod_{i \mid t_{ji} = t} f(\tilde{\tau}(v_{ji}; \rho_{jt}) \mid \hat{\theta}_\ell) \tag{6.40}$$

Here, we have again used auxiliary parameters, as in eqs. (6.35, 6.36), to provide a more tractable likelihood function. The prior clustering bias is as in eq. (6.38), except that o_{jt} is excluded when counting the number of tables M_ℓ^{-t} assigned to each global cluster. To sample from this distribution, we first choose a new cluster $\bar{o}_{jt}$ by marginalizing potential transformations, and then sample $\bar{\rho}_{jt}$ from the resulting conditional distribution. For Gaussian priors on translations $\tilde{\tau}(v_{ji}; \rho_{jt}) = (v_{ji} - \rho_{jt})$, this likelihood has a closed form which generalizes that of eq. (6.39).

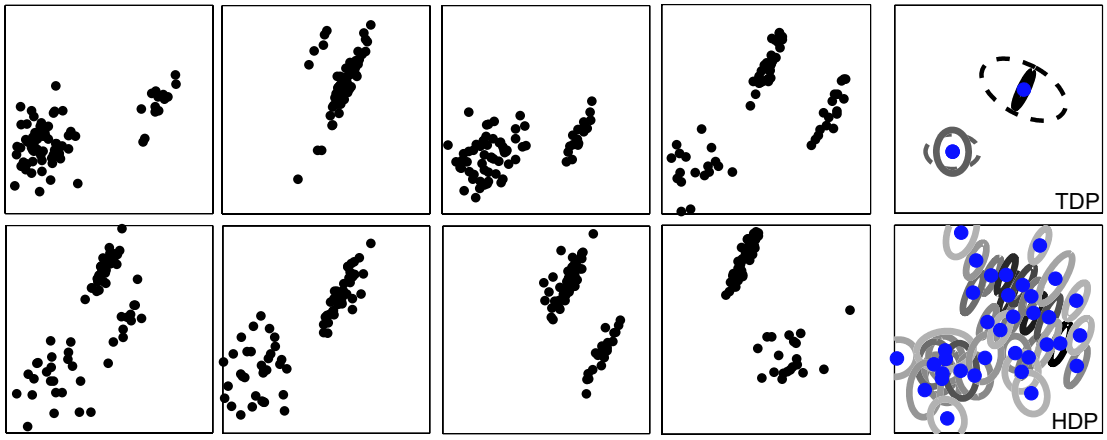


Figure 6.11. Learning HDP and TDP models from a toy set of 2D spatial data. *Left:* Eight of fifty training "images" containing diagonally oriented bars and round blobs. *Upper right:* Global distribution $G_0(\theta, \rho)$ over Gaussian clusters (solid) and translations (dashed) learned by the TDP Gibbs sampler. *Lower right:* Global distribution $G_0(\theta)$ over Gaussian clusters (intensity proportional to probability β_ℓ) learned by the HDP Gibbs sampler.

Concentration Parameter Resampling

The preceding sampling equations assumed fixed values for the concentration parameters γ and α defining the TDP's stick–breaking priors (see eqs. (6.23, 6.24)). These parameters have intuitive interpretations: γ controls the expected number of global clusters, while α determines the average number of transformed clusters instantiated in each group. In many applications, these statistics are unknown, and it would be preferable to also learn them from training data. As is common with Dirichlet process models [76], we thus use vague gamma priors to resample concentration parameters following each Gibbs sampling iteration. The posterior distributions of these parameters depend only on the *number* of currently instantiated clusters, so that auxiliary variable methods developed for the HDP [289] can be applied unchanged to the TDP.

■ 6.2.4 A Toy World: Bars and Blobs

To provide intuition for the spatial structure captured by the TDP, we consider a toy world in which "images" depict a collection of two–dimensional points. As illustrated in Fig. 6.11, the training images we consider typically depict one or more diagonally oriented "bars" in the upper right portion of the image frame. Frequently, round "blobs" of points also appear in the lower left. As in more realistic datasets, the exact locations of these "objects" vary from image to image.

We compare the descriptions of this dataset learned by the previously described TDP Gibbs sampler, as well as a corresponding HDP sampler. Both models use global clusters $\theta_\ell = (\mu_\ell, \Lambda_\ell)$ which parameterize Gaussian distributions, and choose H to be a vague normal–inverse–Wishart prior. For the TDP, transformations ρ define translations of

global cluster means, as in Sec. 5.2.1, and R is taken to be an inverse–Wishart prior on zero–mean Gaussians. For both models, we run the Gibbs sampler for 100 iterations, and resample concentration parameters at each iteration.

As shown in Fig. 6.11, the TDP sampler learns a global distribution $G_0(\theta, \rho)$ which parsimoniously describes these images via translations of two bar and blob–shaped global clusters. In contrast, because the HDP models absolute feature positions, it defines a large set of global clusters which discretize the range of observed object positions. Because a smaller number of features are used to estimate the shape of each cluster, they less closely approximate the true shapes of bars and blobs. More importantly, the HDP model cannot predict the appearance of these objects in new image positions. We thus see that the TDP's use of transformations is needed to adequately transfer information among different object instances, and generalize to novel spatial scenes.

■ 6.3 Modeling Scenes with Unknown Numbers of Objects

The transformed Dirichlet process developed in the preceding section defines global clusters via a parametric, exponential family $F(\theta)$. As suggested by the toy example of Fig. 6.11, this approach could be directly used to construct simple, weakly structured models of object geometry [282]. However, realistic objects have complex internal structure, and signicant local appearance variations. In this section, we thus extend the basic TDP of Fig. 6.9 to learn richer, part–based descriptions of object categories.

As in the nonparametric model of isolated objects developed in Sec. 5.5, we associate parts with clusters of features (v_{ji}, w_{ji}) which have a distinctive, predictable appearance. In particular, each part $\theta_{\ell k} = (\eta_{\ell k}, \mu_{\ell k}, \Lambda_{\ell k})$ of object category ℓ is defined by a Gaussian position distribution $\mathcal{N}(\mu_{\ell k}, \Lambda_{\ell k})$, and a multinomial appearance distribution $\eta_{\ell k}$. Letting H denote a prior measure on part parameters $\theta_{\ell k} \in \Theta$, we then take $F_\ell \sim \mathrm{DP}(\kappa, H)$ to be a discrete distribution characterizing the potentially infinite set of parts underlying the ℓ^{th} visual category:

$$F_\ell(\theta) = \sum_{k=1}^{\infty} \varepsilon_{\ell k} \delta(\theta, \theta_{\ell k}) \qquad \begin{aligned} \boldsymbol{\varepsilon}_\ell &\sim \mathrm{GEM}(\kappa) \\ (\eta_{\ell k}, \mu_{\ell k}, \Lambda_{\ell k}) = \theta_{\ell k} &\sim H \end{aligned} \tag{6.41}$$

The Gaussian parameters $(\mu_{\ell k}, \Lambda_{\ell k})$ associated with each part model feature positions in an object–centered coordinate frame. In the visual scenes considered by this chapter, we expect there to be little direct overlap in the appearance of different categories. For simplicity, eq. (6.41) thus describes categories using independent parts, rather than hierarchically sharing parts as in Sec. 5.5. As in previous sections, we choose H to define a Dirichlet prior $\eta_{\ell k} \sim \mathrm{Dir}(\lambda)$ on each part's discrete appearance distribution, and a normal–inverse–Wishart prior on the corresponding Gaussian shape parameters.

The TDP model of Sec. 6.2.1 employed a global measure G_0 modeling transformations ρ of an infinite set of cluster parameters. Generalizing this construction, we allow infinitely many potential visual object categories o, and characterize transformations

of these part–based models as follows:

$$G_0(o,\rho) = \sum_{\ell=1}^{\infty} \beta_\ell \delta(o,\ell) q(\rho \mid \varphi_\ell) \qquad \begin{aligned} \boldsymbol{\beta} &\sim \mathrm{GEM}(\gamma) \\ \varphi_\ell &\sim R \end{aligned} \tag{6.42}$$

In this distribution, the random variable o indicates the part–based model, as in eq. (6.41), corresponding to some category. The appearance of the j^{th} image is then determined by a set of randomly transformed objects $G_j \sim \mathrm{DP}(\alpha, G_0)$, so that

$$G_j(o,\rho) = \sum_{t=1}^{\infty} \widetilde{\pi}_{jt} \delta(o, o_{jt}) \delta(\rho, \rho_{jt}) \qquad \begin{aligned} \widetilde{\boldsymbol{\pi}}_j &\sim \mathrm{GEM}(\alpha) \\ (o_{jt}, \rho_{jt}) &\sim G_0 \end{aligned} \tag{6.43}$$

In this expression, t indexes the set of object *instances* in image j, which are associated with visual categories o_{jt}.

Each of the N_j features in image j is independently sampled from some object instance $t_{ji} \sim \widetilde{\boldsymbol{\pi}}_j$. As summarized in the graph of Fig. 6.12, this process can be equivalently expressed as follows:

$$(\bar{o}_{ji}, \bar{\rho}_{ji}) \sim G_j \tag{6.44}$$

Here, $\bar{o}_{ji}$ is the global category corresponding to the chosen instance, and the transformation $\bar{\rho}_{ji}$ specifies that instance's location. Parameters corresponding to one of this object's parts are then chosen, and used to generate the observed feature:

$$\begin{aligned} (\bar{\eta}_{ji}, \bar{\mu}_{ji}, \bar{\Lambda}_{ji}) = \bar{\theta}_{ji} &\sim F_{\bar{o}_{ji}} \\ w_{ji} &\sim \bar{\eta}_{ji} \\ v_{ji} &\sim \mathcal{N}\big(\bar{\mu}_{ji} + \bar{\rho}_{ji}, \bar{\Lambda}_{ji}\big) \end{aligned} \tag{6.45}$$

In later sections, we let $k_{ji} \sim \boldsymbol{\varepsilon}_{\bar{o}_{ji}}$ indicate the part underlying the i^{th} feature. As in Sec. 6.1, we consider scale–normalized datasets, and thus associate reference transformations with image–based translations. Generalizations involving more complex transformations, like those discussed in Sec. 5.2.2, are also possible.

The hierarchical, TDP scene model of Fig. 6.12 employs three different stick–breaking processes, allowing uncertainty in the number of visual categories ($\mathrm{GEM}(\gamma)$), parts composing each category ($\mathrm{GEM}(\kappa)$), and object instances depicted in each image ($\mathrm{GEM}(\alpha)$). It thus generalizes the parametric model of Fig. 6.1, which assumed fixed, known sets of parts and objects. In the limit as $\kappa \to 0$, each category uses a single part, and we recover a variant of the simpler TDP model developed in Sec. 6.2.

■ 6.3.1 Learning Transformed Scene Models

To learn the parameters of the visual scene model depicted in Fig. 6.12, we generalize the TDP Gibbs sampler described in Sec. 6.2.3. As before, we maintain a dynamically resized list of the instantiated object instances in each image. Recall that each instance t is defined by a transformation ρ_{jt} of some global object category o_{jt}. Features (w_{ji}, v_{ji})

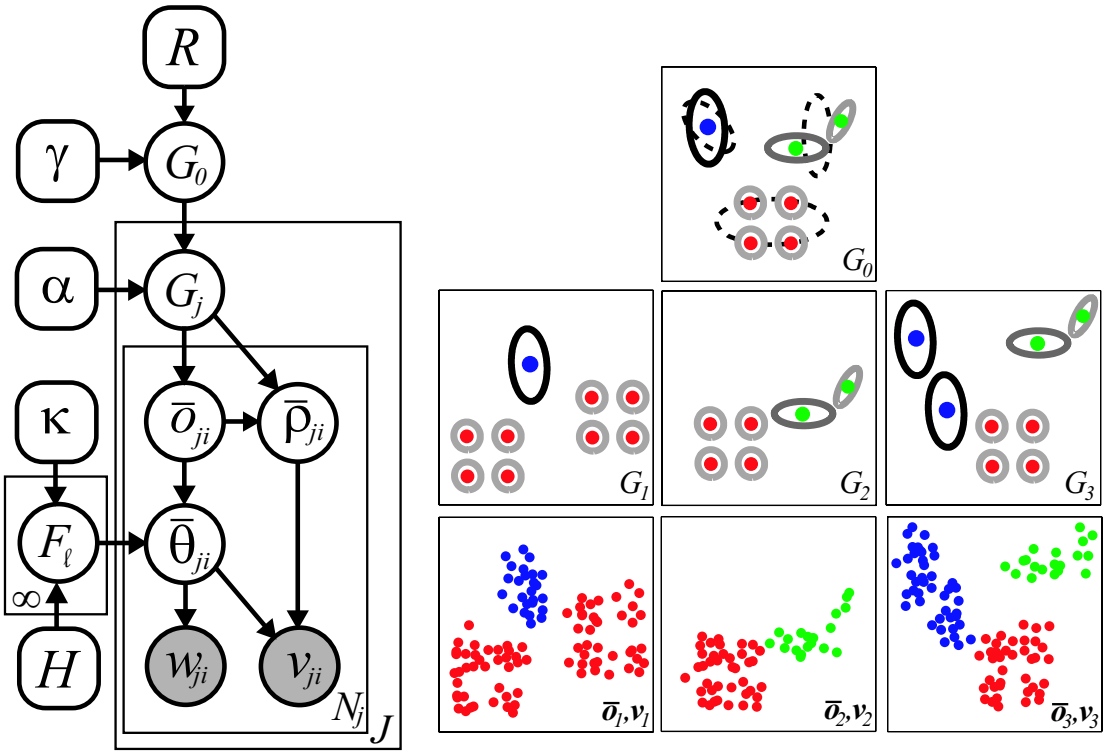


Figure 6.12. TDP model for 2D visual scenes (left), and cartoon illustration of the generative process (right). Global mixture G_0 describes the expected frequency and image position of visual categories, whose internal structure is represented by part–based appearance models $\{F_\ell\}_{\ell=1}^{\infty}$. Each image distribution G_j instantiates a randomly chosen set of objects at transformed locations ρ. Image features with appearance w_{ji} and position v_{ji} are then sampled from transformed parameters $\tau(\bar{\theta}_{ji}; \bar{\rho}_{ji})$ corresponding to different parts of object $\bar{o}_{ji}$. The cartoon example defines three color–coded object categories, which are composed of one (blue), two (green), and four (red) Gaussian parts, respectively. Dashed ellipses indicate transformation priors for each category.

are then characterized by a pair of assignment variables, which identify some part k_{ji} of their associated object instance t_{ji}. Using a blocked Gibbs sampler, we resample these four sets of variables, and thus simultaneously segment and recognize objects.

Resampling Assignments to Object Instances and Parts

Because each object category is defined by an independent set of parts, it is critical to develop a blocked Gibbs sampler which jointly considers the instance and part assignments (t_{ji}, k_{ji}) associated with each feature. Let $\mathbf{t}_{\setminus ji}$ denote all object instance assignments except t_{ji}, and define $\mathbf{k}_{\setminus ji}$ similarly. The Markov properties of the TDP

scene model (see Fig. 6.12) then imply that

$$\begin{aligned} &p(t_{ji}, k_{ji} \mid \mathbf{t}_{\setminus ji}, \mathbf{k}_{\setminus ji}, \mathbf{w}, \mathbf{v}, \mathbf{o}, \boldsymbol{\rho}) \\ &\qquad \propto p(t_{ji} \mid \mathbf{t}_{\setminus ji})\, p(k_{ji} \mid \mathbf{k}_{\setminus ji}, \mathbf{t}, \mathbf{o})\, p(w_{ji} \mid \mathbf{t}, \mathbf{k}, \mathbf{o}, \mathbf{w}_{\setminus ji})\, p(v_{ji} \mid \mathbf{t}, \mathbf{k}, \mathbf{o}, \mathbf{v}_{\setminus ji}, \boldsymbol{\rho}) \end{aligned} \tag{6.46}$$

The first term encourages assignments to object instances associated with many other features, exactly as in eq. (6.33). Similarly, the second term is derived from the stick–breaking prior $\boldsymbol{\varepsilon}_\ell \sim \mathrm{GEM}(\kappa)$ on the probabilities associated with each object's parts:

$$p(k_{ji} \mid t_{ji} = t, o_{jt} = \ell, \mathbf{k}_{\setminus ji}, \mathbf{t}_{\setminus ji}, \mathbf{o}_{\setminus jt}) \propto \sum_{k=1}^{K_\ell} B_{\ell k}^{-i} \delta(k_{ji}, k) + \kappa \delta(k_{ji}, \bar{k}) \tag{6.47}$$

Here, $B_{\ell k}^{-i}$ denotes the number of other features currently assigned to each of the K_ℓ instantiated parts of object ℓ. An assignment to one of the infinitely many equivalent, unoccupied parts is then tractably represented by $\bar{k}$.

The likelihood terms of eq. (6.46) are determined by the global object category associated with each instance t_{ji}, which as before we denote by $z_{ji} = o_{jt_{ji}}$. For existing object instances, or tables, our use of Dirichlet priors induces the following predictive appearance likelihood:

$$p(w_{ji} = w \mid z_{ji} = \ell, k_{ji} = k, \mathbf{t}_{\setminus ji}, \mathbf{k}_{\setminus ji}, \mathbf{o}, \mathbf{w}_{\setminus ji}) = \frac{C_{\ell k w}^{-i} + \lambda / W}{\sum_{w'} C_{\ell k w'}^{-i} + \lambda} \tag{6.48}$$

Here, $C_{\ell k w}^{-i}$ is the number of times appearance descriptor w is assigned to part k of visual category ℓ by $(\mathbf{t}_{\setminus ji}, \mathbf{k}_{\setminus ji}, \mathbf{o})$. If we similarly specialize eq. (6.34), we find that the position likelihood depends on the transformed locations of those features assigned to the same object and part:

$$\begin{aligned} &p(v_{ji} \mid z_{ji} = \ell, k_{ji} = k, \mathbf{t}_{\setminus ji}, \mathbf{k}_{\setminus ji}, \mathbf{o}, \mathbf{v}_{\setminus ji}, \boldsymbol{\rho}) \\ &\qquad \propto p\big(v_{ji} - \rho_{jt_{ji}} \mid \big\{(v_{j'i'} - \rho_{j't_{j'i'}}) \mid z_{j'i'} = \ell, k_{j'i'} = k, (j', i') \neq (j, i)\big\}\big) \\ &\qquad \approx \mathcal{N}\big(v_{ji} - \rho_{jt_{ji}}; \hat{\mu}_{\ell k}, \hat{\Lambda}_{\ell k}\big) \end{aligned} \tag{6.49}$$

Aside from the bookkeeping associated with indexing both object and part assignments, these expressions are equivalent to those arising in our earlier transformed Dirichlet process models. Efficient likelihood evaluation is thus possible by caching transformed statistics of the features associated with each part of every global object category.

To sample according to eq. (6.46), we first evaluate these likelihoods for every existing part, and a potential new part, of each instantiated object. We also determine the likelihood of creating a new object instance by marginalizing potential category assignments and transformations as in eqs. (6.37, 6.39). Combining these likelihoods with the Dirichlet process clustering biases of eqs. (6.33, 6.47), we jointly sample a new instance and part assignments (t_{ji}, k_{ji}). If a new object $t_{ji} = \bar{t}$ is chosen, its corresponding visual category $o_{j\bar{t}}$ and transformation $\rho_{j\bar{t}}$ are then sampled as described in the following section.

Global Object and Transformation Resampling

In the second phase of each Gibbs sampling iteration, we fix object assignments $\mathbf{t}$, and consider potential reinterpretations of each instance t using a new global object category o_{jt}. Because parts and transformations are defined with respect to particular categories, blocked resampling of $(o_{jt}, \rho_{jt}, \{k_{ji} \mid t_{ji} = t\})$ is necessary. As in earlier sections, we resample transformations using auxiliary parameters:

$$\begin{aligned} \hat{\eta}_{\ell k} &\sim p(\eta_{\ell k} \mid \{w_{ji} \mid z_{ji} = \ell, k_{ji} = k\}) \\ (\hat{\mu}_{\ell k}, \hat{\Lambda}_{\ell k}) &\sim p\big(\mu_{\ell k}, \Lambda_{\ell k} \mid \big\{(v_{ji} - \rho_{jt_{ji}}) \mid z_{ji} = \ell, k_{ji} = k\big\}\big) \end{aligned} \tag{6.50}$$

A single sample is drawn for each part of every global category which is associated with at least one feature.

Suppose first that $o_{jt} = \ell$ is fixed. Due to the exponentially large number of joint assignments of this instance's features to parts, the marginal distribution of ρ_{jt} is intractable. However, given ρ_{jt}, part assignments k_{ji} are conditionally independent:

$$\begin{aligned} p(k_{ji} = k \mid w_{ji}, v_{ji}, t_{ji} = t, o_{jt} = \ell, \rho_{jt}, \mathbf{k}_{\setminus ji}, \mathbf{t}_{\setminus ji}, \mathbf{o}_{\setminus jt}) \\ \propto p(k \mid \mathbf{k}_{\setminus ji}, \mathbf{t}, \mathbf{o})\, \hat{\eta}_{\ell k}(w_{ji})\, \mathcal{N}(v_{ji} - \rho_{jt}; \hat{\mu}_{\ell k}, \hat{\Lambda}_{\ell k}) \end{aligned} \tag{6.51}$$

Here, the Dirichlet clustering bias is as in eq. (6.47). Alternatively, given fixed part assignments for all features, ρ_{jt} has a Gaussian posterior:

$$p(\rho_{jt} \mid o_{jt} = \ell, \{k_{ji}, v_{ji} \mid t_{ji} = t\}) \propto \mathcal{N}(\rho_{jt}; \hat{\zeta}_\ell, \hat{\Upsilon}_\ell) \prod_{k=1}^{K_\ell} \prod_{i \mid k_{ji} = k} \mathcal{N}(v_{ji} - \rho_{jt}; \hat{\mu}_{\ell k}, \hat{\Lambda}_{\ell k}) \tag{6.52}$$

The Gaussian parameters $\hat{\varphi}_\ell = (\hat{\zeta}_\ell, \hat{\Upsilon}_\ell)$ of this transformation prior, and the resulting posterior, follow equations identical to those derived for the isolated–object Dirichlet process model in Sec. 5.5.1. Intuitively, fixing $\mathbf{t}$ effectively segments the scene's features into individual objects.

For each candidate visual category o_{jt}, we first perform a small number of auxiliary Gibbs iterations which alternatively sample eqs. (6.51, 6.52). Fixing the final transformations, part assignments may then be directly marginalized to compute the likelihood of o_{jt}. Typically, the posterior distribution of ρ_{jt} is tightly concentrated given fixed $\mathbf{t}$, and 3–5 auxiliary iterations provide an accurate approximation. Combining this likelihood with the Dirichlet clustering bias of eq. (6.38), we resample o_{jt}, and then conditionally choose $(\rho_{jt}, \{k_{ji} \mid t_{ji} = t\})$ via eqs. (6.51, 6.52).

Concentration Parameter Resampling

As in the simpler TDP model of Sec. 6.2.3, we place vague gamma priors on our visual scene model's concentration parameters, and resample them following each Gibbs sampling iteration [76, 289]. This leads to a robust model which makes very weak prior assumptions regarding the true numbers of object categories, parts per object, and objects per scene.

6.3.2 Street and Office Scenes

To evaluate the TDP scene model of Fig. 6.12, we revisit the datasets of street and office scenes illustrated in Fig. 6.2. We consider the same training and test images, and image features, that were used to test the fixed–order scene model in Sec. 6.1.3. During training, we distinguish the manually labeled *object categories* from the *visual categories* composing the TDP's global distribution G_0. We restrict the Gibbs sampler from assigning different objects to the same visual category, but multiple visual categories may be used to describe different forms of a particular object.

When learning TDP scene models, it is also useful to distinguish *rigid objects* (e.g., computer screens, keyboards, mice, and cars) from *textural objects* such as buildings, roads, trees, and office clutter. For rigid objects, we restrict all features composing each labeled training instance to be associated with the *same* transformed global cluster. This constraint, which is enforced by fixing the table assignments t_{ji} for features of rigid objects, ensures that the TDP learns descriptions of complete objects rather than object pieces. For textural categories, we allow the sampler to partition labeled training regions into transformed object instances, and thus automatically discover smaller regions with consistent, predictable structure.

Learning TDP Models of 2D Scenes

For both datasets, we learn model parameters using the Rao–Blackwellized Gibbs sampler developed in Sec. 6.3.1. As before, we used 400 street scenes and 250 office scenes for training, and evaluated recognition performance with the remaining images. To estimate model parameters, we first ran the Gibbs sampler for 500 training iterations using only those features with manually specified object category labels. For street scenes, we then ran another 100 Gibbs sampling iterations using all features. Empirically, this sequential training converges faster because it initializes visual categories with cleanly segmented objects. For each dataset, we compare the full TDP scene model of Fig. 6.12 to a simplified model which constrains each visual category to a single part [282]. This single–part TDP is similar to the model illustrated in Fig. 6.9, except that each visual category also has an associated multinomial appearance distribution.

One of the strengths of the TDP is that the learning process is reasonably insensitive to the particular values of the hyperparameters. The prior distribution H characterizing object parts was set as in Sec. 6.1.3, while the inverse–Wishart transformation prior R weakly favored zero–mean Gaussians covering the full image range. The concentration parameters defining the numbers of visual categories $\gamma \sim \text{Gamma}(1.0, 0.1)$ and parts per category $\kappa \sim \text{Gamma}(1.0, 0.1)$ were then assigned vague gamma priors, and resampled during the learning process. To encourage the learning of larger global clusters for textural categories, the concentration parameter controlling the number of object instances was more tightly constrained as $\alpha \sim \text{Gamma}(1.0, 1.0)$.

In Fig. 6.13, we illustrate the global, visual categories that were learned from the dataset of street scenes. The single–part TDP uses compact global categories, and

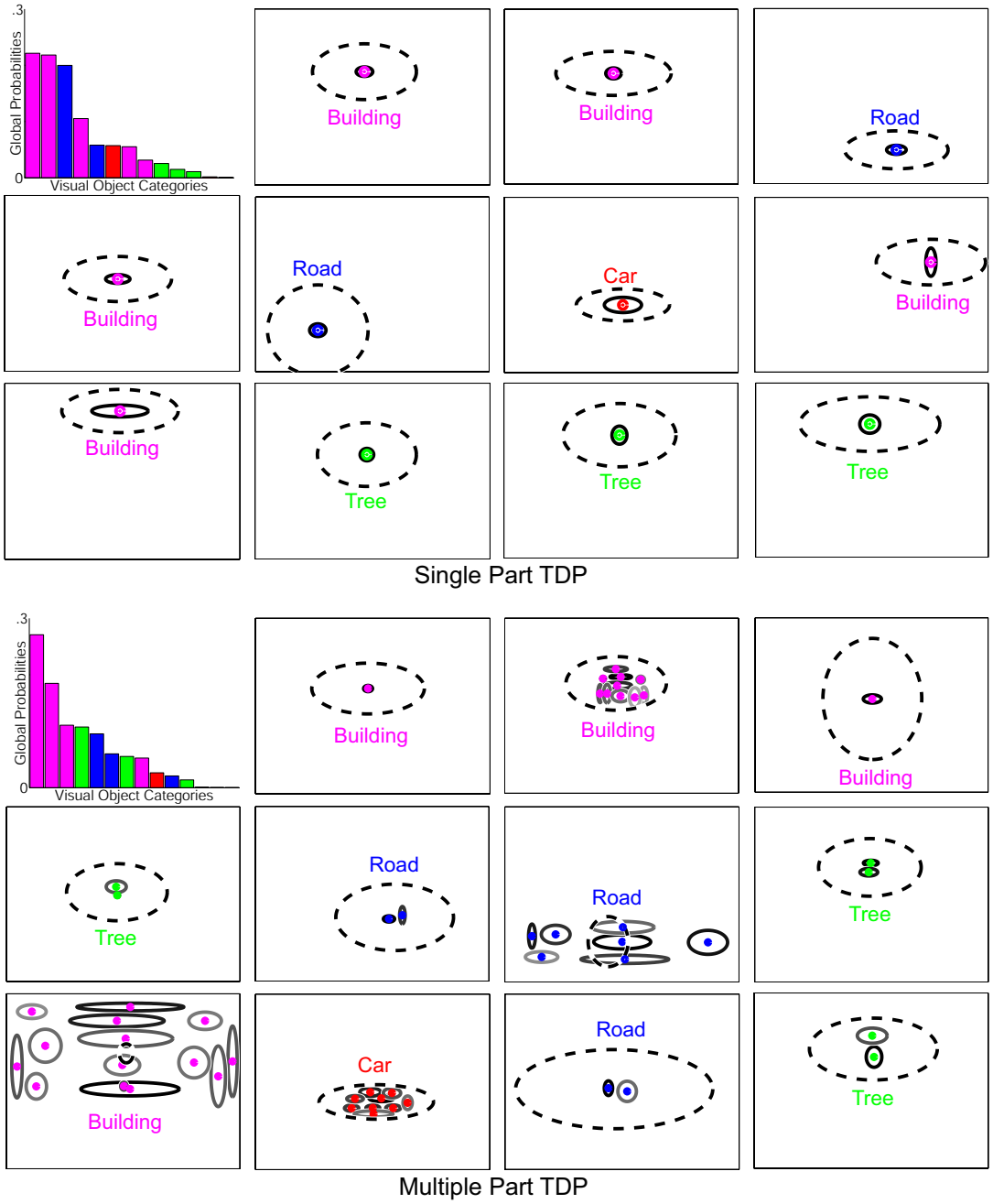


Figure 6.13. Learned TDP models for street scenes containing cars (red), buildings (magenta), roads (blue), and trees (green). *Top:* Simplified, single–part TDP in which the shape of each visual category is described by a single Gaussian (solid ellipses). We show the 11 most common visual categories at their mean positions, and also plot their transformation covariances (dashed ellipses). *Bottom:* Multiple–part TDP in which the number of parts (solid ellipses, intensity proportional to probability) underlying each category is learned automatically. We again show the 11 most probable categories, and their Gaussian transformation distributions (dashed ellipses).

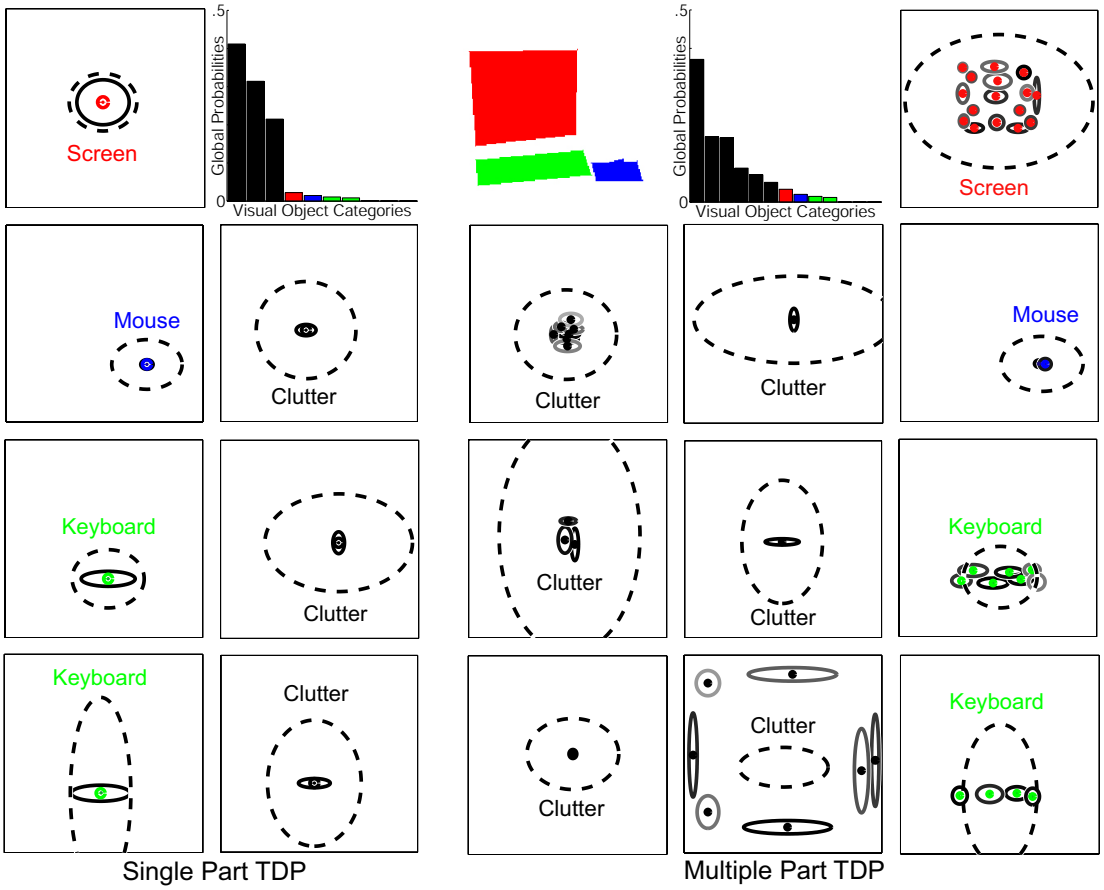


Figure 6.14. Learned TDP models for office scenes containing computer screens (red), keyboards (green), mice (blue), and background clutter (black). *Left:* Simplified, single–part TDP in which the shape of each visual category is described by a single Gaussian (solid ellipses). We show the 7 most common visual categories at their mean positions, and also plot their transformation covariances (dashed ellipses). *Right:* Multiple–part TDP in which the number of parts (solid ellipses, intensity proportional to probability) underlying each category is learned automatically. We show the 10 most probable categories, and their Gaussian transformation distributions (dashed ellipses).

many transformed object instances, to more uniformly spread features across the image. Buildings, roads, and trees are each split into several visual categories, which describe different characteristic structural features. The full TDP scene model creates a more detailed, 9–part car appearance model. It also learns extended, multiple–part models of the large building and road regions which appear in many training images. The full part–based model thus captures some of the coarse–scale structure of street scenes, while the simpler single–part TDP is limited to modeling local feature dependencies.

As shown in Fig. 6.14, the single–part TDP model of office scenes is qualitatively similar to the street scene model: images are described by large numbers of compact transformed clusters. The multiple–part TDP, however, reveals interesting differences

in the global structure of these scene categories. Due to their internal regularities, computer screens and keyboards are each described by detailed visual categories with many parts. To model background clutter, the TDP learns several small clusters of parts which uniformly distribute features within image regions. Because the TDP currently lacks an explicit occlusion model, it also defines a frame–like visual category which captures the background features often found at image boundaries.

Segmentation of Novel Visual Scenes

To analyze test images, we fix the part and object assignments corresponding to the final Gibbs sampling iteration on the training set. To avoid local optima, we then run the test image Gibbs sampler for 50 iterations from each of ten different random initializations. Given the transformed object instances created at each test iteration, the conditional likelihoods of Sec. 6.3.1 were used to estimate the posterior probability that test features were generated by each candidate object category. Analogously to the approaches of Secs. 5.3.6 and 6.1.3, we then averaged the probabilities corresponding to different sampled scene interpretations to determine an overall segmentation.

In Figs. 6.15 and 6.16, we illustrate feature segmentations for several typical test street scenes, and transformed object instances corresponding to one iteration of the Gibbs sampler. In contrast to the fixed–order model of Sec. 6.1, TDPs allow each object category to occur at multiple locations within a single image. This allows the TDP to correctly find multiple cars in several scenes where the fixed–order model only detects a single car. Conversely, because the TDP does not model object relationships, it sometimes incorrectly detects cars in textured regions of buildings. For the fixed–order model, the contextual Gaussian prior suppresses these false alarms by forcing cars to lie beneath the transformed building region.

We show similar segmentation results for office scenes in Figs. 6.17 and 6.18. Computer screens are typically reliably detected, particularly by the multiple–part TDP model. Perhaps surprisingly, mice are also detected with reasonable accuracy, although there are more false alarms than with the contextual model. In addition to accurately segmenting screen features, the part–based TDP model correctly associates a single transformed object cluster with most screen instances. In contrast, the weaker appearance model of the single–part TDP causes it to create several transformed clusters for many computer screens, and thereby incorrectly label adjacent background features.

As confirmed by the ROC curves of Fig. 6.19, both TDP models provide segmentations which improve substantially on a model based solely on feature appearance (see Fig. 6.7). For large, rigid objects like computer screens and keyboards, including parts substantially improves recognition performance. The two TDP models perform similarly when segmenting cars, perhaps due to their lower typical resolution. However, the street scene interpretations illustrated in Figs. 6.15 and 6.16 show that the part–based TDP does a better job of *counting* the true number of car instances depicted in each image. While including parts leads to more intuitive global models of textural categories, for these simple datasets it does not improve segmentation accuracy.

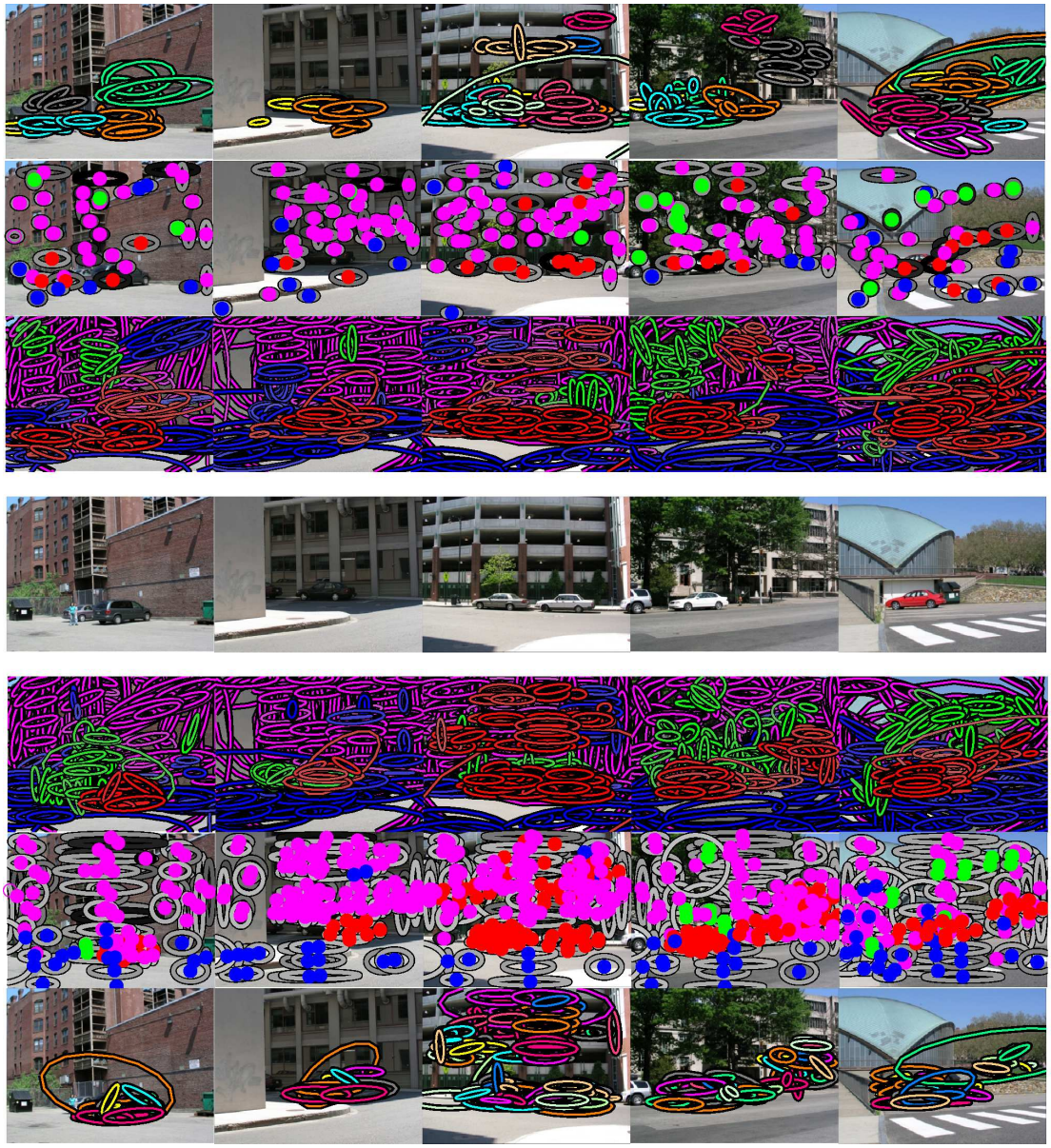


Figure 6.15. Feature segmentations produced by TDP models of street scenes containing cars (red), buildings (magenta), roads (blue), and trees (green). We compare a simplified TDP model which describes object shape via a single Gaussian cluster (top rows) to the full, multiple–part TDP model (bottom rows) of Fig. 6.12. *Row 4:* Five test images. *Rows 3 & 5:* Segmentations for each model, in which features are assigned to the object category with the highest posterior probability. *Rows 2 & 6:* Parts corresponding to the objects instantiated at a single Gibbs sampling iteration. *Rows 1 & 7:* Color–coded assignments of features to different parts and instances of the screen category.

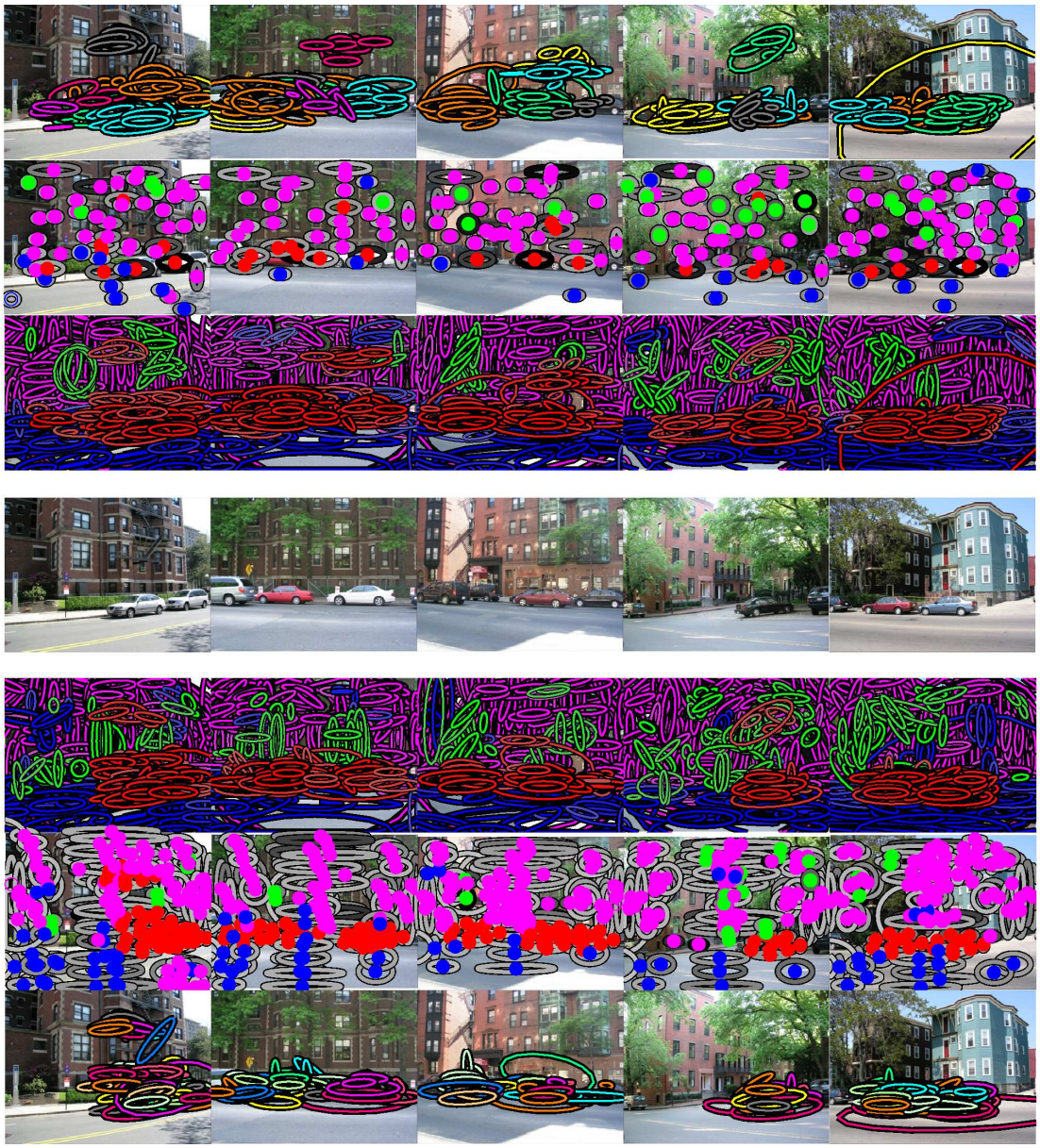


Figure 6.16. Additional feature segmentations produced by TDP models of street scenes containing cars (red), buildings (magenta), roads (blue), and trees (green). We compare a simplified TDP model which describes object shape via a single Gaussian cluster (top rows) to the full, multiple–part TDP model (bottom rows) of Fig. 6.12. *Row 4:* Five test images. *Rows 3 & 5:* Segmentations for each model, in which features are assigned to the object category with the highest posterior probability. *Rows 2 & 6:* Parts corresponding to the objects instantiated at a single Gibbs sampling iteration. *Rows 1 & 7:* Color–coded assignments of features to different parts and instances of the screen category.

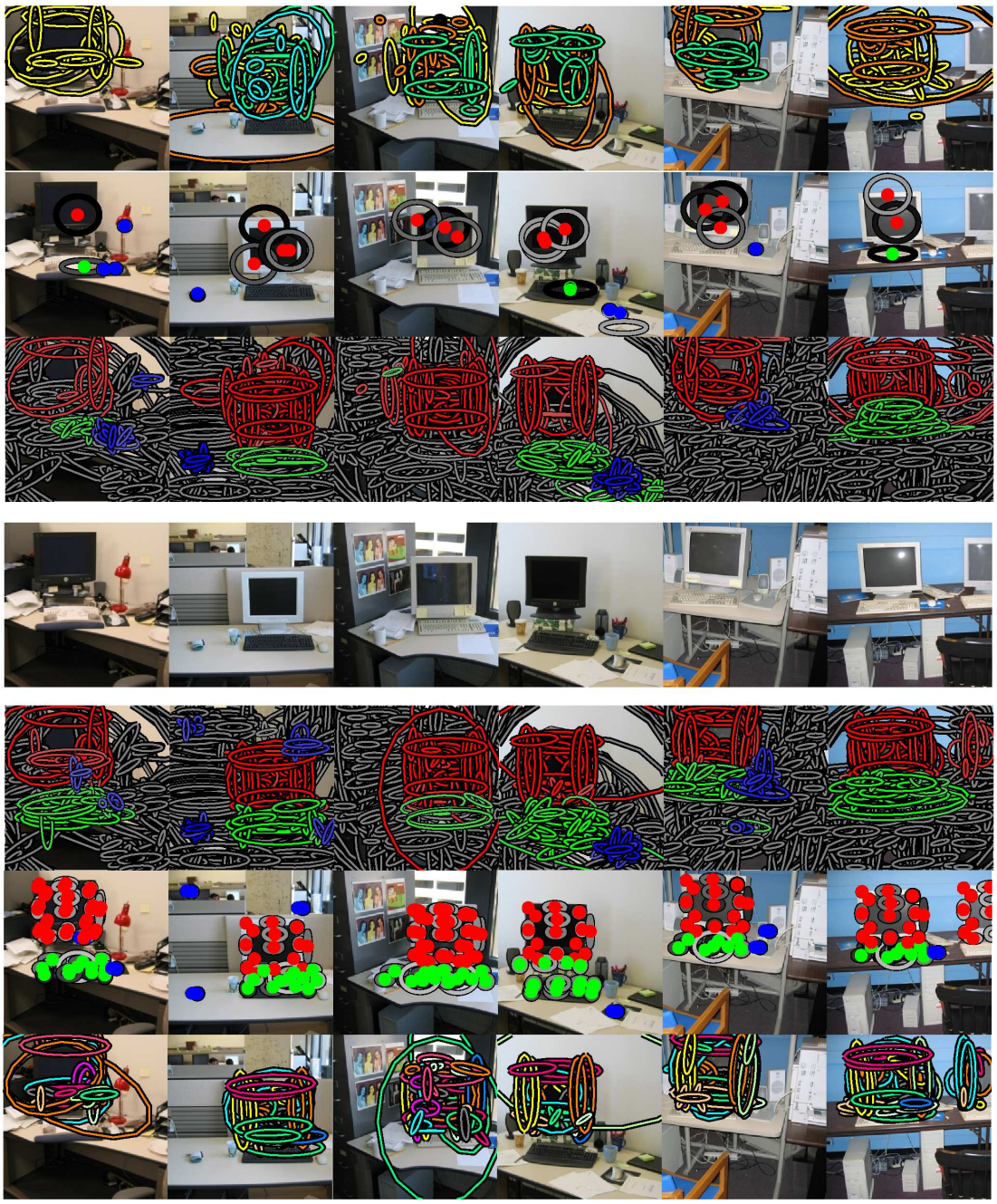


Figure 6.17. Feature segmentations produced by TDP models of office scenes containing computer screens (red), keyboards (green), mice (blue), and background clutter (gray). We compare a simplified TDP model which describes object shape via a single Gaussian cluster (top rows) to the full, multiple–part TDP model (bottom rows) of Fig. 6.12. *Row 4:* Six test images. *Rows 3 & 5:* Segmentations for each model, in which features are assigned to the object category with the highest posterior probability. *Rows 2 & 6:* Parts corresponding to the objects instantiated at a single Gibbs sampling iteration (background clutter not shown). *Rows 1 & 7:* Color–coded assignments of features to different parts and instances of the screen category.

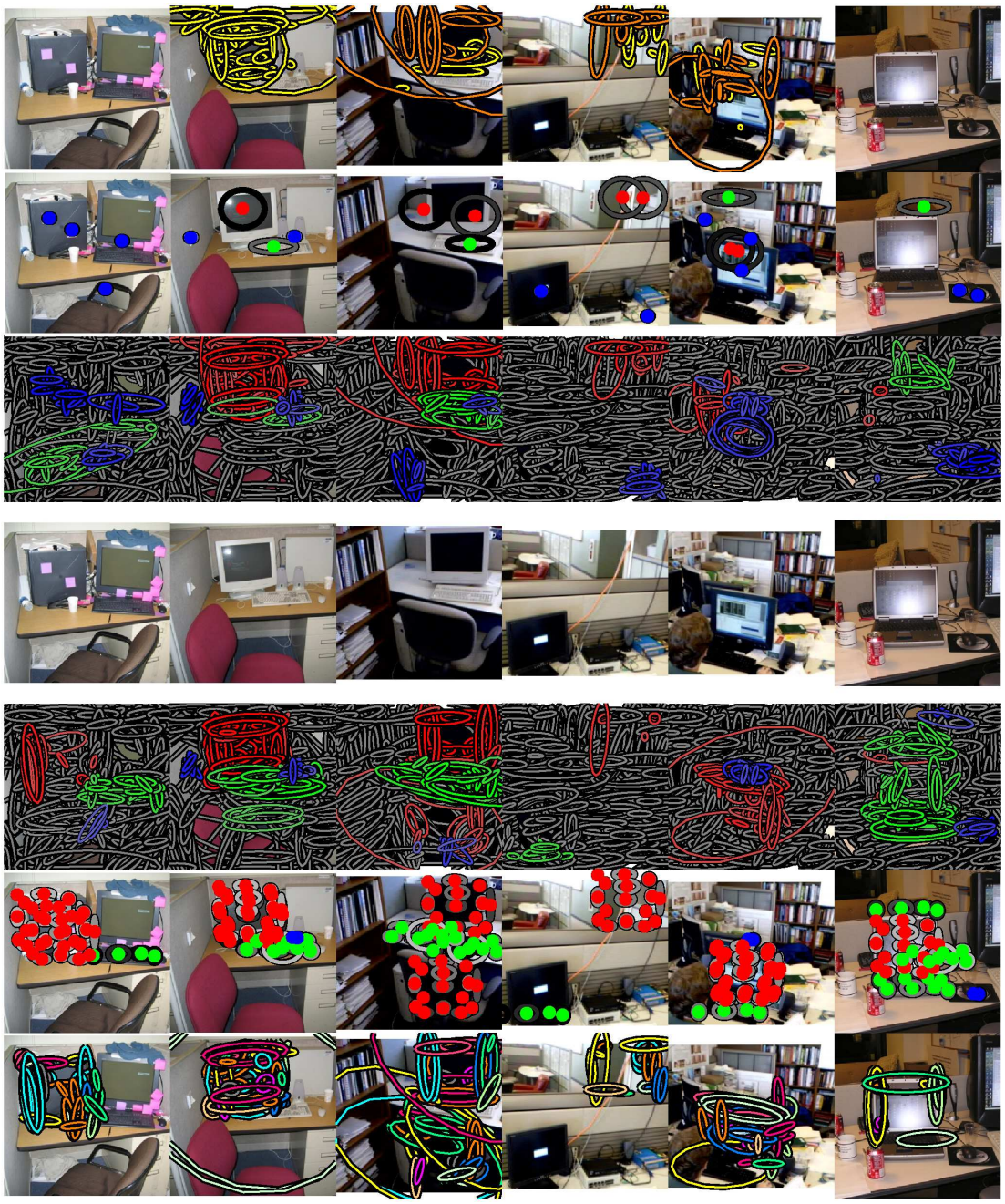


Figure 6.18. Additional feature segmentations produced by TDP models of office scenes containing computer screens (red), keyboards (green), mice (blue), and background clutter (gray). We compare a simplified TDP model which describes object shape via a single Gaussian cluster (top rows) to the full, multiple–part TDP model (bottom rows) of Fig. 6.12. *Row 4:* Six test images. *Rows 3 & 5:* Segmentations for each model, in which features are assigned to the object category with the highest posterior probability. *Rows 2 & 6:* Parts corresponding to the objects instantiated at a single Gibbs sampling iteration (background clutter not shown). *Rows 1 & 7:* Color–coded assignments of features to different parts and instances of the screen category.

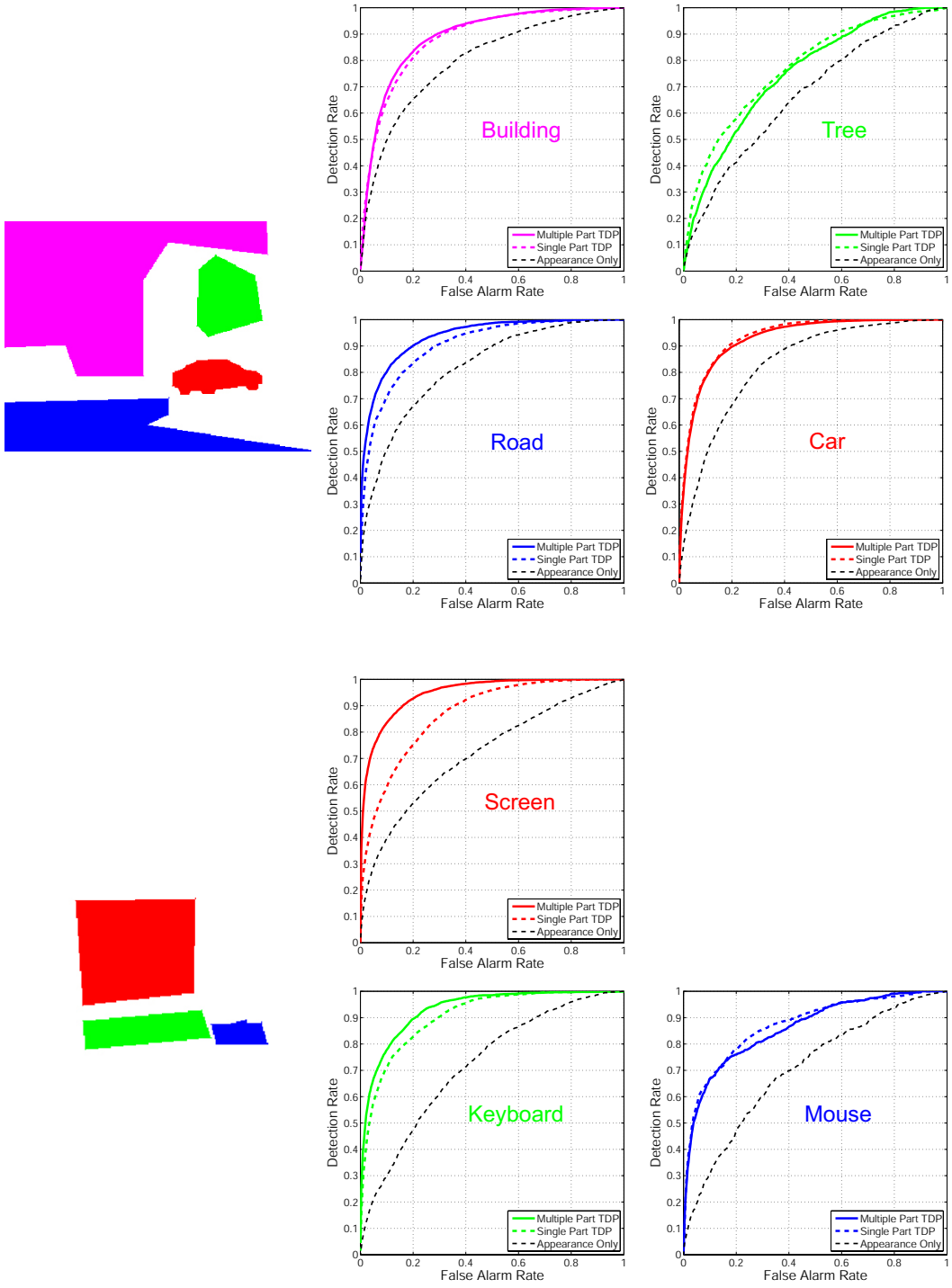


Figure 6.19. ROC curves summarizing segmentation performance for the features composing street scenes (top) and office scenes (bottom). We compare the full TDP scene model of Fig. 6.12 (solid lines) to a simplified, single–part TDP model (dashed lines, colored) and a bag of features model based solely on local appearance (dashed lines, black).

Comparing the TDP's performance to results for the fixed–order, contextual scene model (see Fig. 6.8), we find that their complementary strengths are useful in different situations. For example, the fixed–order model's very strong spatial prior leads to improved building and road detection, but worse performance for the less structured features composing trees. The TDP more cleanly segments individual cars from the background, but also makes additional false alarms in contextually implausible regions of buildings; the overall performance of the two models is comparable. For computer screens, the TDP's allowance for multiple instances, and creation of additional parts to form a stronger appearance model, leads to substantial performance improvements. Finally, we emphasize that the TDP also estimates the *number* of objects composing each scene, a task which is beyond the scope of the fixed–order model.

■ 6.4 Hierarchical Models for Three–Dimensional Scenes

The preceding scene models decompose images via translations of 2D object appearance models, and thus implicitly assume they have been normalized to account for scale and viewpoint variations. Many algorithms relax these assumptions by considering more complex image–based transformations [3, 82, 181]. In this section, we instead propose a hierarchical model which describes object categories via their 3D structure and appearance. Scale–invariant recognition is then achieved via the translation of 3D objects, and the perspective projections underlying the imaging process.

We first discuss methods for depth estimation from binocular stereo images, which we use to calibrate our 3D scene models. We then extend the TDP to model 3D object structure, and develop Monte Carlo methods which simultaneously recognize objects and reconstruct scene geometry.

■ 6.4.1 Depth Calibration via Stereo Images

Binocular stereo vision systems employ a pair of adjacent cameras. Each point in an image taken by one camera corresponds to some point on an *epipolar line* in the second camera's image. If the relative position and orientation of the cameras is known, the displacement or *disparity* between matching points can then be used to infer the 3D locations of observed features [91].

Let $u = (u^x, u^y, u^z)$ denote the world coordinates of a 3D point. For simplicity, we assume that the z-axis has been chosen to align with the camera's optical axis. Then, indexing pixels (v^x, v^y) from the optical center, the perspective projection of u equals

$$v^x = \xi \frac{u^x}{u^z} \qquad v^y = \xi \frac{u^y}{u^z} \tag{6.53}$$

where ξ denotes the magnification, in pixels, corresponding to the camera's focal length. Other coordinate systems are easily accomodated by appropriate transformations [91].

The training images used in Sec. 6.4.4 were captured by a calibrated stereo camera (the MEGA-D, by Videre Design). As in recent approaches to sparse wide baseline

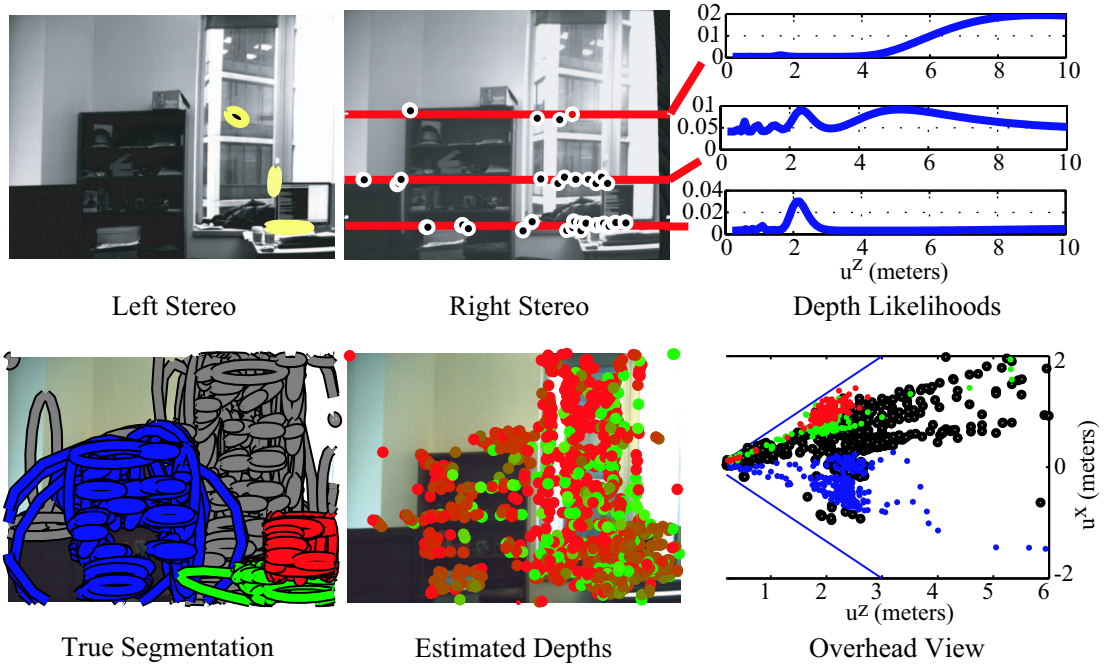


Figure 6.20. Stereo likelihoods for an office scene depicting a computer screen, desk, and bookshelf (color–coded, lower left). *Top row:* For three features, we show matches along epipolar lines in the right stereo image, and corresponding depth likelihoods. *Bottom row:* Greedy depth estimates are independently chosen for each feature. In the frontal view (center), close features are green and distant red. The overhead view (right) colors features according to their associated object (left).

stereo [199], we begin by extracting regions of interest in both the left and right images. For each interest point in the reference (left) image, we then search for the best matching regions along the corresponding epipolar line (see Fig. 6.20). Match quality is measured via the Euclidean distance between SIFT descriptors [188]. Let v^d denote the disparity, in pixels, corresponding to a candidate pair of matching features. Each match corresponds to some set of world coordinates:

$$u^x = \frac{v^x}{\xi} u^z \qquad u^y = \frac{v^y}{\xi} u^z \qquad u^z = \frac{\xi D}{v^d} \tag{6.54}$$

Here, D is the baseline distance between cameras (in our case, 89 mm). Note that we have written the world u^x and u^y in terms of the unknown depth u^z, rather than the disparity v^d. This form emphasizes our primary interest in the underlying 3D geometry, and is more easily incorporated with our generative model of visual scenes.

Robust Disparity Likelihoods

Because we represent images by a sparse set of interest regions, we must only estimate scene depths at these points. While this problem is simpler than the estimation of

dense depth maps, it is still ill–posed based solely on local feature correspondences. In particular, repetitive scene structures and occlusion effects near object boundaries often lead to inaccurate disparity estimates for some features. In Fig. 6.20, we illustrate the noisy depth estimates produced by local matching in stereo images of office scenes. Wide baseline stereo algorithms typically employ a geometric validation stage to discard such outliers [199]. This approach would work poorly for our application, however, because features near object boundaries are often the most informative for recognition tasks. We instead propose a probabilistic model which robustly converts approximate disparity matches to depth distributions. The learning algorithms developed in Sec. 6.4.3 then use geometric models of objects to impose a scene structure which resolves local ambiguities.

Consider a feature which has candidate stereo matches at C different disparities $\{\bar{v}_c^d\}_{c=1}^C$, and let $\bar{v}_c^s$ denote the matching score (distance between SIFT descriptors) for $\bar{v}_c^d$. Features with no matches induce an uniformative likelihood on the underlying scene depth u^z. Otherwise, at most one match can correspond to the true scene depth, and the others must be outliers. Let a be an unobserved random variable indicating which of the C matches is *not* an outlier, and take $a = 0$ if all matches are outliers. Neglecting possible correlations due to scene structure, we assume that inlier and outlier matches are independently sampled as

$$\begin{aligned} p\Big(\{\bar{v}_c^d, \bar{v}_c^s\}_{c=1}^C \mid u^z\Big) &= \sum_{a=0}^{C} p\Big(\{\bar{v}_c^d, \bar{v}_c^s\}_{c=1}^C, a \mid u^z\Big) \\ &\propto \sum_{a=0}^{C} p(a) \prod_{c=1}^{C} p(\bar{v}_c^d \mid a, u^z)\, p(\bar{v}_c^s \mid a) \end{aligned} \tag{6.55}$$

Let ϵ denote the prior probability that all observations are outliers ($a = 0$), so that all other outlier hypotheses have equal probability $(1-\epsilon)/C$. We assume that correct matches are corrupted by Gaussian noise, while outlier disparities are sampled uniformly over a range determined by the camera geometry:

$$p(\bar{v}_c^d \mid a = c, u^z) = \mathcal{N}\Big(\bar{v}_c^d; \frac{\xi D}{u^z}, \sigma_d^2\Big) \tag{6.56}$$

$$p(\bar{v}_c^d \mid a \neq c, u^z) = \mathcal{U}\Big(\bar{v}_c^d; D_{\min}, D_{\max}\Big) \tag{6.57}$$

We also assign the inlier and outlier matching scores $\bar{v}_c^s$ log–normal densities with differing mean and variance.

Parameter Estimation using the EM Algorithm

To estimate the parameters of this likelihood function, we collected disparity matches for 16,000 monitor and bookshelf features from the stereo training images used in Sec. 6.4.4. Because each selected object was approximately orthogonal to the optical axis, the median depth of each instance's raw stereo matches provides an accurate estimate of true depth for all features. We may then compute maximum likelihood parameter

estimates by extending standard EM updates for mixture models [107]. The E–step averages over possible outlier hypotheses a, producing a lower bound on the likelihood which is maximized in the M–step.

From our training set, we estimated the probability that all matches are outliers to be $\epsilon = 0.22$, and the noise level for correct matches as $\sigma_d = 2.4$ pixels. Fig. 6.20 illustrates depth likelihoods correponding to three sample features. Intuitively, matches with small disparities lead to greater depth uncertainty, due to the inversion induced by the perspective projection of eq. (6.54). When there are many conflicting matches, the likelihood becomes uniform.

■ 6.4.2 Describing 3D Scenes using Transformed Dirichlet Processes

To develop a model for 3D scene features, we generalize the 2D TDP scene model of Sec. 6.3. As illustrated in Fig. 6.21, our part–based model for the ℓ^{th} object category is again defined by an infinite discrete distribution, whose complexity is controlled by a stick–breaking prior:

$$F_\ell(\theta) = \sum_{k=1}^{\infty} \varepsilon_{\ell k} \delta(\theta, \theta_{\ell k}) \qquad \begin{aligned} \varepsilon_\ell &\sim \text{GEM}(\kappa) \\ (\eta_{\ell k}, \mu_{\ell k}, \Lambda_{\ell k}) = \theta_{\ell k} &\sim H \end{aligned} \tag{6.58}$$

As before, $\eta_{\ell k} \sim \text{Dir}(\lambda)$ defines a multinomial appearance distribution for the k^{th} part of object ℓ. Now, however, $(\mu_{\ell k}, \Lambda_{\ell k})$ parameterizes a 3D Gaussian distribution, which specifies the expected world coordinates of object features, relative to the camera.

Given these part–based, 3D object models, the global distribution G_0 defining visual object categories, and local distributions G_j specifying each image's object instances, are sampled as in eqs. (6.42, 6.43). Each feature in image j is then independently sampled in three stages. First, a visual category $\bar{o}_{ji}$ and transformation $\bar{\rho}_{ji}$ are chosen from G_j, selecting a particular object instance. Second, parameters corresponding to one of that objects' parts are selected, and a 3D feature position sampled relative to that instance's location:

$$\begin{aligned} (\bar{\eta}_{ji}, \bar{\mu}_{ji}, \bar{\Lambda}_{ji}) = \bar{\theta}_{ji} &\sim F_{\bar{o}_{ji}} \\ u_{ji} &\sim \mathcal{N}(\tau(\bar{\mu}_{ji}, \bar{\Lambda}_{ji}; \bar{\rho}_{ji})) = \mathcal{N}\big(\bar{\mu}_{ji} + \bar{\rho}_{ji}, \bar{\Lambda}_{ji}\big) \end{aligned} \tag{6.59}$$

While the simulations of this chapter transform objects via 3D translations, more general rigid body motion could be incorporated as described in Sec. 5.2.2. Finally, we observe a 2D feature with appearance $w_{ji} \sim \bar{\eta}_{ji}$, and position v_{ji} determined from u_{ji} through the deterministic perspective projection of eq. (6.53). See Fig. 6.21 for an example illustrating this generative process.

Given a feature at image location v_{ji}, the corresponding world coordinates u_{ji} have a single remaining degree of freedom, which can be expressed in terms of the unknown depth u_{ji}^z (see eq. (6.54)). When stereo cameras are available, the disparity likelihoods depicted in Fig. 6.20 provide noisy depth estimates. Otherwise, other cues must be used to resolve uncertain scene structure. As we demonstrate in the following sections, the identification of known object categories can provide a powerful geometric constraint.

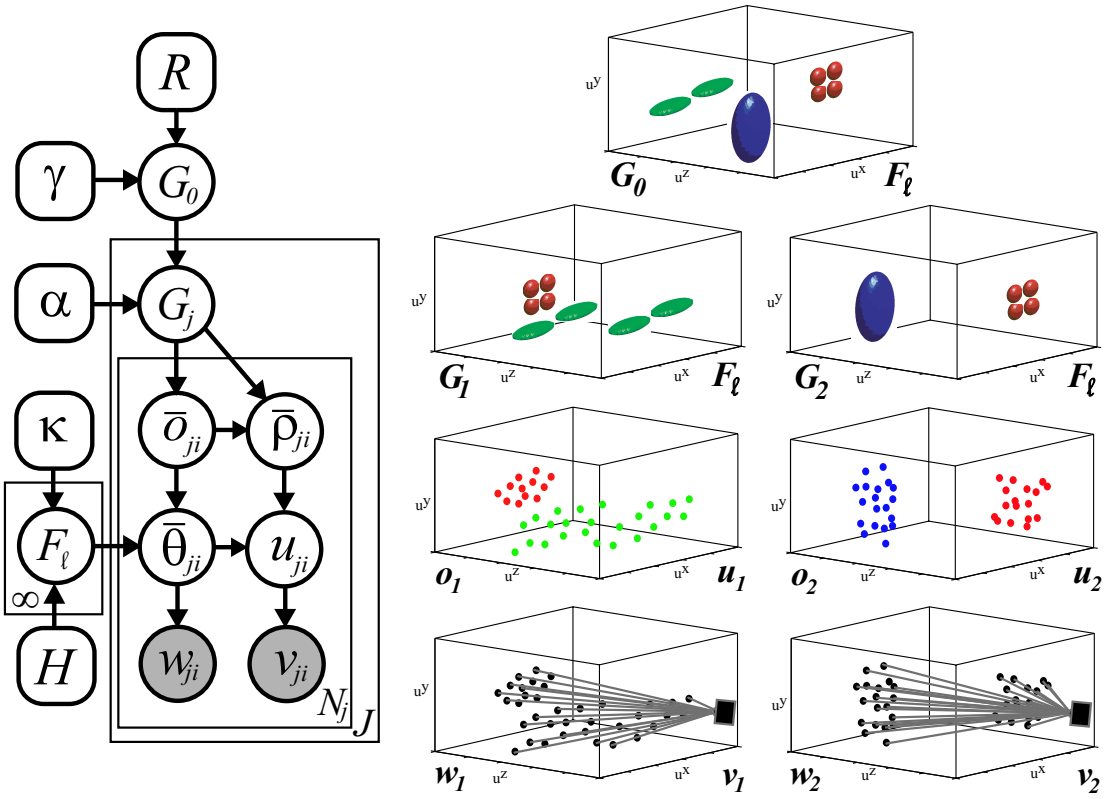


Figure 6.21. TDP model for 3D visual scenes (left), and cartoon illustration of the generative process (right). Global mixture G_0 describes the expected frequency and 3D position of visual categories, whose internal structure is represented by part–based appearance models $\{F_\ell\}_{\ell=1}^{\infty}$. Each image mixture G_j instantiates a randomly chosen set of objects at transformed locations ρ. 3D feature positions u_{ji} are then sampled from transformed parameters $\tau(\bar{\theta}_{ji}; \bar{\rho}_{ji})$ corresponding to parts of object $\bar{o}_{ji}$. The camera observes projections v_{ji} of these features, with part–dependent appearance w_{ji}. The cartoon example defines three color–coded object categories, which are composed of one (blue), two (green), and four (red) Gaussian parts, respectively. For clarity, Gaussian transformation priors are not explicitly shown.

■ 6.4.3 Simultaneous Depth Estimation and Object Categorization

To learn the parameters of our 3D TDP scene model, we extend the Rao–Blackwellized Gibbs sampler developed in Sec. 6.3.1. For each observed feature (w_{ji}, v_{ji}), we resample the corresponding 3D depth u_{ji}^z, as well as the assignments (t_{ji}, k_{ji}) of that feature to object instances t and parts k. Then, for each instance t in image j, we jointly resample assignments o_{jt} to visual categories with corresponding transformations ρ_{jt} and part assignments $\{k_{ji} \mid t_{ji} = t\}$. Iterating these steps, we approximately sample from the model's posterior distribution over scene interpretations, simultaneously recognizing objects and reconstructing 3D geometry.

Intuitively, the most likely depth u_{ji}^z for a particular feature is strongly dependent on the 3D object instance t_{ji}, and corresponding part k_{ji}, generating that feature. For

adequate convergence of the Gibbs sampler, we thus employ blocked sampling updates of $(t_{ji}, k_{ji}, u^z_{ji})$. By the Markov properties of the TDP, we then have

$$\begin{aligned} p\big(t_{ji}, k_{ji}, u^z_{ji} \mid v_{ji}, \mathbf{t}_{\setminus ji}, \mathbf{k}_{\setminus ji}, \mathbf{w}, \mathbf{u}_{\setminus ji}, \mathbf{o}, \boldsymbol{\rho}\big) &\propto p(t_{ji} \mid \mathbf{t}_{\setminus ji})\, p(k_{ji} \mid \mathbf{k}_{\setminus ji}, \mathbf{t}, \mathbf{o}) \\ \cdots \times p(w_{ji} \mid \mathbf{t}, \mathbf{k}, \mathbf{o}, \mathbf{w}_{\setminus ji})\, p(v^x_{ji}, v^y_{ji}, u^z_{ji} \mid \mathbf{t}, \mathbf{k}, \mathbf{o}, \mathbf{u}_{\setminus ji}, \boldsymbol{\rho})\, p\big(v^d_{ji} \mid u^z_{ji}\big) \end{aligned} \quad (6.60)$$

The first three terms are unchanged from Sec. 6.3.1. However, the position likelihood for feature (v^x_{ji}, v^y_{ji}) is complicated by the imaging process. In particular, each candidate depth u^z_{ji} selects a different 3D point $\tilde{v} u^z_{ji}$ along a ray $\tilde{v}$ defined by eq. (6.54). The fourth term of eq. (6.60) is then the probability that the transformed 3D Gaussian corresponding to the chosen instance and part (see Fig. 6.21) assigns to this point:

$$p(v^x_{ji}, v^y_{ji}, u^z_{ji} \mid t_{ji} = t, k_{ji} = k, \mathbf{t}_{\setminus ji}, \mathbf{k}_{\setminus ji}, \mathbf{o}, \mathbf{u}_{\setminus ji}, \boldsymbol{\rho}) \propto \mathcal{N}\big(\tilde{v} u^z_{ji}; \hat{\mu}_{o_{jt}k} + \rho_{jt}, \hat{\Lambda}_{o_{jt}k}\big) \quad (6.61)$$

Here, $(\hat{\mu}_{\ell k}, \hat{\Lambda}_{\ell k})$ denote the regularized mean and covariance induced by the features currently assigned to those parts. Letting $\tilde{\mu}_{tk} = \hat{\mu}_{o_{jt}k} + \rho_{jt}$ denote the transformed position for part k of instance t, and conditioning this 3D Gaussian to the projection ray $\tilde{v}$, we recover a *scaled* 1D Gaussian distribution in depth:

$$\begin{aligned} p\big(u^z_{ji} \mid k_{ji} = k, t_{ji} = t, o_{jt} = \ell\big) &\propto \omega_{tk} \mathcal{N}\big(u^z_{ji}; \zeta_{tk}, \chi_{tk}\big) \\ \chi^{-1}_{tk} = \tilde{v}^T \hat{\Lambda}^{-1}_{\ell k} \tilde{v} \qquad & \chi^{-1}_{tk} \zeta_{tk} = \tilde{v}^T \hat{\Lambda}^{-1}_{\ell k} \tilde{\mu}_{tk} \\ \log \omega_{tk} = \frac{1}{2} \log \frac{\chi_{tk}}{|\hat{\Lambda}_{\ell k}|} &- \frac{1}{2} (\tilde{v}\zeta_{tk} - \tilde{\mu}_{tk})^T \hat{\Lambda}^{-1}_{\ell k} (\tilde{v}\zeta_{tk} - \tilde{\mu}_{tk}) \end{aligned} \quad (6.62)$$

Note that transformed parts whose mean is farther from the projection ray are given lower overall weight ω_{tk}. To evaluate the likelihood of new object instances $\bar{t}$, we integrate over potential transformations $\rho_{j\bar{t}}$ as in eq. (6.39), and evaluate eq. (6.62) with an appropriately inflated 3D covariance.

The final term of eq. (6.60) is the depth likelihood corresponding to stereo–based disparity matches. For monocular images, we jointly resample $(t_{ji}, k_{ji}, u^z_{ji})$ by using the Dirichlet process clustering biases, and appearance likelihood, to reweight the Gaussian mixture of eq. (6.62). For stereo training images, we evaluate the likelihood learned in Sec. 6.4.1 on a uniformly spaced grid determined by the largest expected scene geometry. We then evaluate eq. (6.62) on the same grid for each candidate instance and part, and resample from that discrete distribution. Given Z depths, and T_j object instances with (on average) K parts, this resampling step requires $\mathcal{O}(ZT_jK)$ operations.

In the second phase of each Gibbs sampling iteration, we fix feature depths $\mathbf{u}^z$ and object assignments $\mathbf{t}$, and consider potential reinterpretations of each instance t using a new global object category o_{jt}. Because this stage fixes the world coordinates associated with each feature, no changes to the sampling updates derived in Sec. 6.3.1 are needed. Importantly, our use of continuous, Gaussian position densities avoids an expensive discretization of 3D world coordinates. Finally, we again place vague gamma priors on the three TDP concentration parameters, and resample them at each iteration.

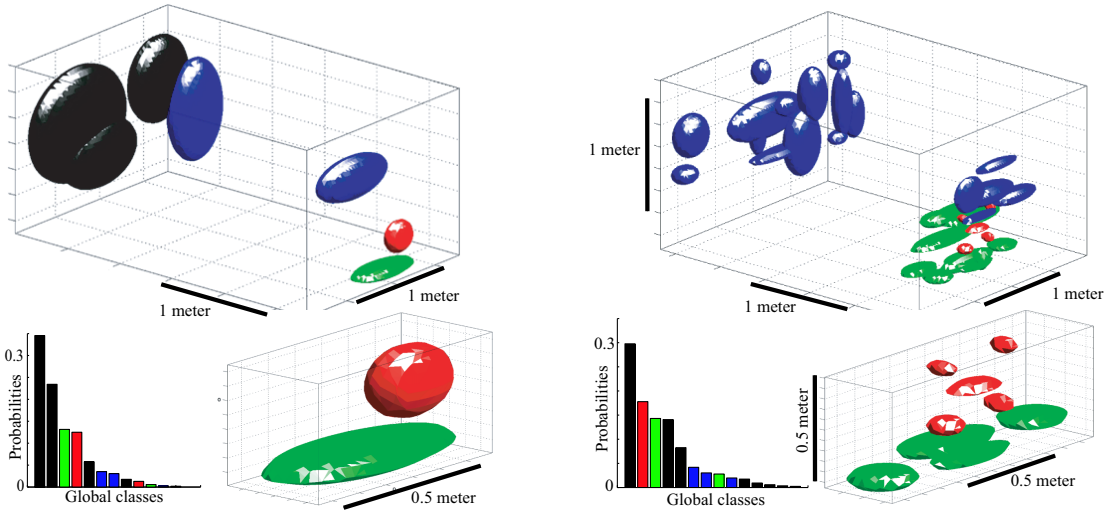


Figure 6.22. Visual object categories learned from stereo images of office scenes containing computer screens (red), desks (green), bookshelves (blue), and background clutter (black). Covariance ellipses model 3D part geometry, and are positioned at their mean transformed locations. Bar charts show posterior probabilities for all instantiated global categories. *Left:* Simplified TDP model which describes object shape via a single Gaussian cluster. We show the seven most likely visual categories (top), and a close–up view of the screen and desk models (bottom). *Right:* Multiple part TDP model as in Fig. 6.21. For clarity, we show the most likely parts (those generating 85% of observed features) for the five most common non–background categories (top). The close–up view shows a five–part screen model, and a four–part desk model (bottom).

■ 6.4.4 Scale–Invariant Analysis of Office Scenes

We now consider a dataset of stereo office scenes containing four labeled objects: computer screens, desks, bookshelves, and background clutter. With 120 training images segmented as in Fig. 6.20, we used the Gibbs sampler of Sec. 6.4.3 to learn TDP scene models. Fig. 6.22 shows the visual categories created by the full TDP model of Fig. 6.21, and a simpler model which constrains each category to a single part, after 100 sampling iterations. While the single–part TDP captures coarse geometric relationships, parts allow more accurate descriptions of object structure. Note, for example, that the screen model defines parts characterizing each of its four corners.

As in Sec. 6.3.2, the Gibbs sampler allows each manually labeled object category to be associated with several visual categories. For the 3D office scene dataset, both TDP models learn (without supervision) two shapes for bookshelves, one horizontal and the other vertical. Note that our allowance for transformations causes the TDP to model scaling via 3D translations, rather than by creating multiple visual categories.

In Fig. 6.24, we show several typical test image interpretations for the part–based TDP scene model. For stereo test images, TDP depth estimates consistently improve on the raw estimates of Fig. 6.20. In addition, as shown by the ROC curves of Fig. 6.23, the TDP more accurately segments features into object categories than a histogram

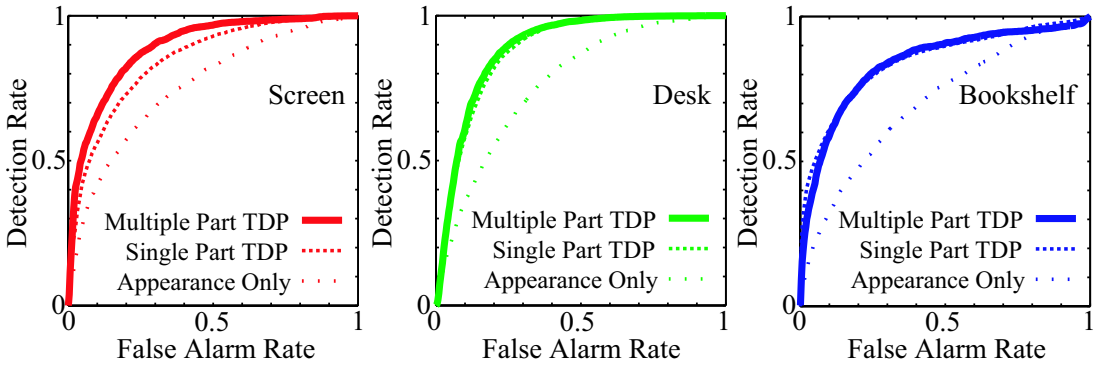


Figure 6.23. ROC curves for the segmentation of features corresponding to computer screens (red), desks (green), and bookshelves (blue). Using stereo test images, we compare the single and multiple part TDPs of Fig. 6.22 to a classifier based on feature appearance.

model based solely on feature appearance. Parts improve segmentation performance for monitors, but not for the less structured desk and bookshelf categories.

Fig. 6.24 also shows preliminary inference results for monocular test images. For these results, we set depth likelihoods to slightly favor features which are placed farther from the camera. This heuristic avoids a degenerate configuration sometimes seen with uninformative depth likelihoods, in which a single object instance is placed very close to the camera and used to explain all observed features. By running the Gibbs sampler on monocular images, we detect monitors at multiple scales, and thus approximately infer scene geometry via the presence of familiar objects. Although the TDP produces more false alarms for monocular scenes than for stereo test images, the overall scene interpretation is often still consistent. Further investigation of the accuracy and robustness of the TDP's 3D reconstructions from monocular scenes is ongoing.

■ 6.5 Discussion

By coupling Dirichlet processes with spatial transformations, we have developed flexible, hierarchical descriptions of multiple object scenes. Our results clearly demonstrate that simple bag of features models neglect valuable spatial relationships, which may dramatically improve object localization performance. Importantly, our use of nonparametric priors leads to algorithms which automatically partition scenes into visual objects categories, and objects into parts. Furthermore, using stereo training images we have learned effective 3D scene models which jointly recognize objects and reconstruct geometric structures. These TDP models suggest several promising research directions, which Chap. 7 discusses in more detail.

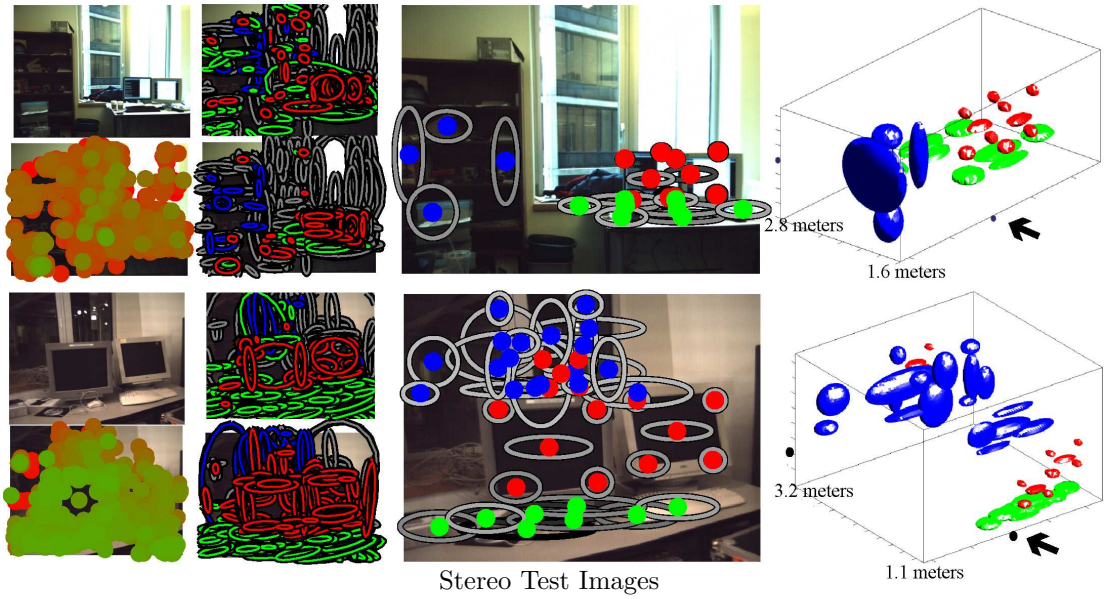


Figure 6.24. Analysis of stereo (top) and monocular (bottom) test images using the 3D, part–based TDP model of Fig. 6.22. For comparison, the first monocular test image is the same as the first stereo image, but ignores the disparity–based depth likelihoods. Each result group (left, clockwise) shows the test image, a segmentation based solely on feature appearance, a TDP segmentation, and corresponding TDP depth estimates (green features are near, red far). We also show transformed 3D parts corresponding to non–background object instances inferred by the TDP (right), and overlay perspective projections of these parts on the test image (center).

Chapter 7

Contributions and Recommendations

PRECEDING chapters developed statistical methods for the visual detection, categorization, and tracking of objects. We now survey the principal contributions underlying our results, and outline several promising avenues for further research.

7.1 Summary of Methods and Contributions

Computer vision systems must be robust to wide variations in object appearance, the often small size of training databases, and ambiguities induced by articulated or partially occluded objects. We believe that structured statistical models, which explicitly characterize the uncertainties inherent in natural scenes, will play an important role in addressing these challenges. This thesis develops several models which integrate graphical representations with nonparametric statistical methods. This approach allows us to minimize potentially damaging assumptions about the often uncertain statistics of visual scenes, while still permitting efficient learning and inference algorithms.

We examine these general themes in the context of two particular computer vision tasks. The first half of this thesis considers distributed representations for articulated objects, and in particular develops a Monte Carlo method for tracking hand motion from video sequences. We then turn to the problem of object detection and categorization, and develop methods for learning hierarchical models of objects, the parts composing them, and the scenes surrounding them. These applications raise different and complementary issues: the hand tracker estimates the pose of a particular high–dimensional geometric model, while our scene hierarchy learns less precise visual descriptions of entire object categories. Nevertheless, we show that nonparametric methods can flexibly exploit the local, geometric structure characterizing both applications.

Motivated by visual tracking problems, Chap. 3 first considers more general inference tasks defined in graphical models containing continuous, non–Gaussian random variables. We formulate a *nonparametric belief propagation* (NBP) algorithm which approximates continuous sufficient statistics via nonparametric, sample–based messages. NBP updates the particles underlying these messages using a very flexible family of Monte Carlo methods, and can thus be easily adapted to a huge range of applications.

Moreover, we use multiscale, KD–tree density representations to derive sampling algorithms which efficiently and accurately fuse information during each message update.

Turning to the hand tracking application, Chap. 4 begins by developing a distributed, graphical representation of the hand's kinematic, structural, and dynamic characteristics. We also provide a set of auxiliary occlusion masks which permit consistent, local decompositions of color and edge–based likelihoods. Applying NBP to this model, we develop a tracking algorithm which enforces global constraints via lower–dimensional estimates of the rigid bodies composing the hand. In particular, by locally propagating information among the hand's fingers, we avoid explicitly considering the high–dimensional pose space which plagues traditional articulated trackers. Via an additional analytic approximation, the NBP algorithm also consistently infers occlusion events in a distributed fashion.

In many vision applications, precise geometric models like the one used in our hand tracker are unavailable. Chap. 5 thus considers the complementary problem of learning descriptions of object categories from training images. In particular, we define a hierarchical model which describes several related object categories via a common set of shared parts. This approach couples topic models originally used for text analysis with spatial transformations, and thus describes the geometric structure underlying image features. We show that geometric relationships encode important information, and that shared representations improve performance when few training images are available. In addition, by adapting the Dirichlet process we develop learning algorithms which automatically identify an appropriate *number* of latent parts. Empirical results then show that these nonparametric methods desirably increase the learned model's complexity as additional training images become available.

Generalizing these object models, Chap. 6 develops integrated, hierarchical representations of multiple object scenes. We first develop a parametric, fixed–order model which describes contextual relationships among known sets of objects. To model more general scenes, we then propose a nonparametric framework which couples Dirichlet processes with random *sets* of spatial transformations. The resulting *transformed Dirichlet process* (TDP) provides a consistent, generative model for scenes in which the numbers of parts composing each object, objects depicted in each image, and total object categories are all uncertain. Applied to a challenging dataset of street and office scenes, the TDP automatically segments image features into object categories. Finally, using binocular stereo images we extend the TDP to learn three–dimensional descriptions of object categories. Efficient Monte Carlo methods then simultaneously recognize objects and reconstruct scene geometry. Importantly, our use of nonparametric priors leads to robust learning algorithms which require few manually specified parameters.

■ 7.2 Suggestions for Future Research

We conclude by discussing a variety of open research directions suggested by our approaches to articulated tracking and scene understanding. In addition, we briefly survey

potential implications of our statistical and computational methods for other application domains.

7.2.1 Visual Tracking of Articulated Motion

The NBP algorithm uses random samples to approximate the true, continuous sufficient statistics needed for optimal inference. As the number of samples becomes large, standard asymptotics guarantee that these Monte Carlo methods provide accurate approximations (see Sec. 2.4). In practical applications, however, computational considerations frequently limit NBP to more moderate sample sizes. For example, our hand tracking results used a few hundred samples for each message update, which is insufficient to densely populate the six–dimensional pose space of each hand component. Because NBP propagates information among the hand's fingers, it is less prone to global tracking failures than traditional particle filters. Nevertheless, for rapid or partially occluded motion sequences, undersampled message approximations may produce noisy, inaccurate pose estimates.

Because NBP places few restrictions on the potentials of the underlying graphical model, these sampling issues are its main practical limitation. The brute force solution, of course, is to use more samples when approximating each message. Because NBP may be easily parallelized, this approach is sometimes feasible. For example, the computational cost of visual tracking is usually dominated by image–based likelihood evaluations. Modern graphics hardware provides one natural candidate for more rapidly evaluating particle likelihoods, and thus tractably representing each message by additional samples. More generally, one can consider modifications of NBP which construct samples with improved statistical properties. The following sections describe several variants of this idea in more detail.

Improved Proposal Distributions

In many applications of NBP, including our hand tracker, importance sampling methods are used to correct for otherwise intractable analytic potential functions. As discussed in Sec. 3.1, there is an extensive literature on improved proposal distributions for particle filters [11, 70, 72, 183], which could be easily adapted to NBP. However, the image–based likelihoods arising in visual tracking are typically too complex for these standard methods. Alternatively, image–based feature detectors could be used to identify candidate hand configurations in a bottom–up fashion [261]. Because our graphical model uses a distributed hand representation, these detectors would have the simpler task of finding hand components, such as fingertips, rather than global hand configurations.

In Sec. 3.2.4, we contrasted two forms of the NBP message updates. The second, "belief sampling" form reduces computational costs by reusing a common sample set among several outgoing messages. However, in applications such as our hand tracker, this approach shows increased sensitivity to outlier particles, and may exhibit instability. While Alg. 4.2 heuristically addressed this issue by thresholding particle weights, a complete conceptual understanding of this phenomenon remains to be developed.

Informative Kinematics and Dynamics

Although the graphical model developed in Chap. 4 captures the hand's kinematic constraints, it assumes all kinematically valid poses are equally likely. In contrast, biomechanical [333] and empirical [293, 334] studies have shown strong dependencies among the hand's various joints. Hand motion also exhibits a great deal of temporal structure, particularly when application domains such as sign language recognition are considered [333]. While these relationships are difficult to express analytically, there are a variety of statistical methods for learning improved kinematic and dynamic models from training data. For example, kernel density estimates of these relationships could be easily incorporated into an NBP tracking algorithm, and thus better focus samples on likely hand poses.

Alternative Density Representations

The NBP algorithm developed in Chap. 3 constructs continuous message functions by convolving each particle with a Gaussian smoothing kernel. As described in Sec. 2.4.2, asymptotically motivated methods then automatically determine this kernel's variance. Empirically, however, these bandwidth selection rules are sometimes unstable given small sample sets. The robustness of NBP might thus be improved by considering other approaches to nonparametric density estimation, like those surveyed in Sec. 2.5. For example, a Dirichlet process prior could be used to summarize the samples from each message or belief update by a smaller Gaussian mixture. This approach would allow the expected scale of the true beliefs to be encoded via the Dirichlet process' base measure. Furthermore, reductions in outgoing message size may lower the computational cost of the sampling iterations employed by subsequent message updates.

■ 7.2.2 Hierarchical Models for Objects and Scenes

Our hierarchical scene models use nonparametric, Dirichlet process priors to robustly learn data–driven appearance models, and transfer knowledge among object categories. The following sections discuss potential generalizations which capture additional properties of real scenes, and alternative frameworks for learning and inference.

Transferring Knowledge Among Object Categories

In Chap. 5, we adapted the hierarchical Dirichlet process (HDP) [289] to describe several object categories via a common set of shared parts. The experimental results of Sec. 5.6 then revealed an interesting tradeoff. Given few training images, shared parts lead to substantial gains for a *detection* task, in which objects are distinguished from background clutter. Conversely, for a *recognition* task in which one of several object categories is chosen, the HDP is slightly less effective than a set of independent, unshared Dirichlet process mixtures. This degradation is caused by additional confusion between pairs of categories which reuse shared parts in similar proportions. Of course, one would ideally like to learn shared representations which perform well for both tasks.

One potential solution considers an alternative learning criterion. In particular, the Gibbs sampler derived in Sec. 5.5.1 explores the HDP parameters' posterior distribution, and favors parameters which assign high likelihood to the training images. If two categories are visually similar, the Dirichlet process' stick–breaking prior will then favor simpler models which associate them with similar parts. Alternatively, one could consider discriminative learning techniques which optimize the *conditional* probability of each object category label, given its associated training image [304]. In principle, such approaches might allocate additional parts to better distinguish visually similar categories. However, while discriminative methods are widely used for parametric learning, they do not seem to have been previously adapted to Dirichlet processes. Variational methods, which have been used for generative learning of Dirichlet process mixtures [29], may provide a useful starting point to further develop these ideas.

Another explanation for the HDP's performance degradation is that object categories reuse *identical* parts in different proportions. Adapting ideas from Chap. 6, if categories instead reused *transformed* parts, they might become more distinguishable while still transferring useful information. In the simplest case, such transformations could shift the position of each part, as in Sec. 5.2. However, we expect that appearance transformations, which account for the unique "style" of different categories [291], would provide greater benefits. Note that this approach would require an alternative, continuous feature appearance descriptor.

Richer Descriptions of Visual Scenes

Chap. 6 develops a pair of 2D visual scene models with complementary strengths. The parametric, fixed–order model of Sec. 6.1 describes contextual correlations in the positions at which objects are observed, but assumes each image depicts some fixed, *known* set of objects. Conversely, the nonparametric, TDP model of Sec. 6.3 allows uncertainty in the number of object instances, but neglects contextual relationships. Both models employ part–based descriptions of internal object structure.

While the fixed–order model's assumptions are only appropriate for toy datasets, it would be desirable to develop TDP models which also capture contextual relationships. Recently, a correlated topic model [30] was proposed which generalizes LDA (see Sec. 2.2.4) by modeling dependencies in the probabilities which documents associate with different topics. Contextual relationships have a similar form: given that one object category is present, certain other categories are also likely to be observed. However, we would additionally like to model correlations in the locations of those objects. Currently, the extension of correlated topic models to nonparametric, Dirichlet process priors remains an open problem.

Sec. 6.4 describes a TDP model for 3D scenes which raises several additional challenges. In particular, methods for recognizing objects from multiple viewpoints, and dealing with partial occlusions, must play a role in any realistic scene model. One approach to modeling multiple viewpoints is to simply let the TDP partition those views into several global, visual categories. However, we would expect better performance

from a generalized model which shares parts or features among views [299]. Alternatively, we could consider more general spatial transformations which model viewpoint via rotation of a single 3D model. This approach would require some method for dealing with the self–occlusions induced by each object's internal structure. It remains to be seen whether the occlusion masks we proposed for articulated tracking could also be adapted to such TDP scene models.

Finally, while we demonstrated promising reconstructions of 3D scenes from monocular test images, our current model is far less robust in this situation than when given stereo pairs. Better models of background scene structure, and some description of contextual relationships, will likely play an important role in improving these results.

Alternative Learning Algorithms and Nonparametric Methods

As discussed in Sec. 2.5, the Dirichlet process allows uncertainty in the number of clusters associated with a given dataset, has desirable asymptotic guarantees, and leads to simple, effective learning algorithms. However, it is certainly not the only distribution with these properties, and some alternative may prove more suitable for describing visual scenes. Similarly, while our Rao–Blackwellized Gibbs samplers are often effective, it would be interesting to consider generalizations based on the many other methods for learning Dirichlet process mixtures, as reviewed in Sec. 2.5.3.

■ 7.2.3 Nonparametric and Graphical Models

While we have focused on computer vision tasks, the statistical approaches developed in this thesis are more broadly useful. One common theme running through our work is the combination of variational and Monte Carlo methods for learning and inference. While Monte Carlo methods have strong asymptotic guarantees, and can be applied to complex models, in practice they are unreliable in high–dimensional spaces. We have thus made extensive use of variational and analytic approximations to reduce dimensionality, and combine local Monte Carlo estimates. For example, NBP uses a variational approximation to define factorized, local sufficient statistics, and then approximates those statistics via importance sampling. Similarly, our TDP Gibbs samplers make extensive use of Rao–Blackwellization to reduce model dimensionality and improve convergence.

A second theme is the integration of nonparametric statistical methods with graphical models. Using graphical models, we are able to flexibly exploit domain knowledge, and better utilize small, sparsely labeled training databases. Nonparametric methods then lead to robust models whose complexity grows as additional data is observed, and inference algorithms which consistently maintain complex, multimodal uncertainties.

While we are certainly not the first to utilize structured nonparametric models, we believe they play a critical role in our results, and will prove useful in other domains. For example, NBP has already been applied to other tracking problems [261, 284], and a challenging sensor network localization task [142]. The TDP generalizes models used for text analysis [31, 289], and would likely prove effective in a range of applications arising in speech processing, bioinformatics, and remote sensing.

Bibliography

[1] P. Abrahamsen. A review of Gaussian random fields and correlation functions. Technical Report 917, Norwegian Computing Center, April 1997.

[2] N. J. Adams and C. K. I. Williams. Dynamic trees for image modelling. *Image and Vision Computing*, 21:865–877, 2003.

[3] S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1475–1490, November 2004.

[4] S. M. Aji and R. J. McEliece. The generalized distributive law. *IEEE Transactions on Information Theory*, 46(2):325–343, March 2000.

[5] D. L. Alspach and H. W. Sorenson. Nonlinear Bayesian estimation using Gaussian sum approximations. *IEEE Transactions on Automatic Control*, 17(4):439–448, August 1972.

[6] S. Amari. Information geometry on hierarchy of probability distributions. *IEEE Transactions on Information Theory*, 47(5):1701–1711, July 2001.

[7] Y. Amit, D. Geman, and X. Fan. A coarse–to–fine strategy for multiclass shape detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26 (12):1606–1621, December 2004.

[8] B. D. O. Anderson and J. B. Moore. *Optimal Filtering*. Prentice Hall, New Jersey, 1979.

[9] C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50:5–43, 2003.

[10] C. E. Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *Annals of Statistics*, 2(6):1152–1174, 1974.

[11] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non–Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, February 2002.

[12] V. Athitsos and S. Sclaroff. Estimating 3D hand pose from a cluttered image. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 432–439, 2003.

[13] Z. D. Bai, C. Radhakrishna Rao, and L. C. Zhao. Kernel estimators of density function of directional data. *Journal of Multivariate Analysis*, 27:24–39, 1988.

[14] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. M. Blei, and M. I. Jordan. Matching words and pictures. *Journal of Machine Learning Research*, 3:1107–1135, 2003.

[15] O. Barndorff-Nielsen. *Information and Exponential Families*. John Wiley, New York, 1978.

[16] M. J. Beal, Z. Ghahramani, and C. E. Rasmussen. The infinite hidden Markov model. In *Neural Information Processing Systems 14*, pages 577–584. MIT Press, 2002.

[17] P. N. Belhumeur. A Bayesian approach to binocular stereopsis. *International Journal of Computer Vision*, 19(3):237–262, 1996.

[18] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, April 2002.

[19] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, September 1975.

[20] A. C. Berg, T. L. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondences. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 26–33, 2005.

[21] J. M. Bernardo and A. F. M. Smith. *Bayesian Theory*. John Wiley, New York, 2000.

[22] C. Berrou. The ten-year-old turbo codes are entering into service. *IEEE Communications Magazine*, pages 110–116, August 2003.

[23] C. Berrou, A. Glavieux, and P. Thitimajshima. Near Shannon limit error–correcting coding and decoding: Turbo–codes. In *International Conference on Communications*, volume 2, pages 1064–1070, 1993.

[24] D. P. Bertsekas. *Dynamic Programming and Optimal Control: Volume I*. Athena Scientific, Belmont, Massachusetts, 1995.

[25] D. Bertsimas and I. Popescu. Optimal inequalities in probability theory: A convex optimization approach. *SIAM Journal on Optimization*, 15(3):780–804, 2005.

[26] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society, Series B*, 36:192–223, 1974.

[27] E. Bienenstock, S. Geman, and D. Potter. Compositionality, MDL priors, and object recognition. In *Neural Information Processing Systems 9*, pages 838–844. MIT Press, 1997.

[28] D. Blackwell and J. B. MacQueen. Ferguson distributions via Pólya urn schemes. *Annals of Statistics*, 1(2):353–355, 1973.

[29] D. M. Blei and M. I. Jordan. Variational inference for Dirichlet process mixtures. *Bayesian Analysis*, 1(1):121–144, 2006.

[30] D. M. Blei and J. D. Lafferty. Correlated topic models. In *Neural Information Processing Systems 18*. MIT Press, 2006.

[31] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[32] J. Blinn. The algebraic properties of second–order surfaces. In J. Bloomenthal, editor, *Introduction to Implicit Surfaces*, pages 52–97. Morgan Kaufmann, 1997.

[33] E. Borenstein and S. Ullman. Class–specific, top–down segmentation. In *European Conference on Computer Vision*, volume 2, pages 109–122, 2002.

[34] C. Bouman and M. Shapiro. A multiscale random field model for Bayesian image segmentation. *IEEE Transactions on Image Processing*, 3(2):162–177, March 1994.

[35] P. Brémaud. *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues*. Springer-Verlag, New York, 1999.

[36] L. D. Brown. *Fundamentals of Statistical Exponential Families*. Institute of Mathematical Statistics, Hayward, California, 1986.

[37] W. L. Buntine. Operations for learning with graphical models. *Journal of Artificial Intelligence Research*, 2:159–225, 1994.

[38] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, November 1986.

[39] G. Casella and C. P. Robert. Rao–Blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94, 1996.

[40] E. Charniak. Bayesian networks without tears. *AI Magazine*, 12(4):50–63, 1991.

[41] K. C. Chou, A. S. Willsky, and A. Benveniste. Multiscale recursive estimation, data fusion, and regularization. *IEEE Transactions on Automatic Control*, 39(3): 464–478, March 1994.

[42] S-Y. Chung, G. D. Forney, T. J. Richardson, and R. Urbanke. On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit. *IEEE Communications Letters*, 4(2):58–60, February 2001.

[43] P. Clifford. Markov random fields in statistics. In G. R. Grimmett and D. J. A. Welsh, editors, *Disorder in Physical Systems*, pages 19–32. Oxford University Press, Oxford, 1990.

[44] D. Collett and T. Lewis. Discriminating between the Von Mises and wrapped normal distributions. *Austral. Journal of Statistics*, 23(1):73–79, 1981.

[45] G. F. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42(2-3):393–405, March 1990.

[46] A. Corduneanu and C. M. Bishop. Variational Bayesian model selection for mixture distributions. In *Artificial Intelligence and Statistics 8*, 2001.

[47] J. Coughlan and H. Shen. Shape matching with belief propagation: Using dynamic quantization to accomodate occlusion and clutter. In *CVPR Workshop on Generative Model Based Vision*, June 2004.

[48] J. M. Coughlan and S. J. Ferreira. Finding deformable shapes using loopy belief propagation. In *European Conference on Computer Vision*, volume 3, pages 453–468, 2002.

[49] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley, New York, 1991.

[50] R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer-Verlag, New York, 1999.

[51] D. Crisan and A. Doucet. A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on Signal Processing*, 50(3):736–746, March 2002.

[52] I. Csiszár. I–Divergence geometry of probability distributions and minimization problems. *Annals of Probability*, 3(1):146–158, 1975.

[53] I. Csiszár. A geometric interpretation of Darroch and Ratcliff's generalized iterative scaling. *Annals of Statistics*, 17(3):1409–1413, 1989.

[54] G. Csurka et al. Visual categorization with bags of keypoints. In *ECCV Workshop on Statistical Learning in Computer Vision*, 2004.

[55] D. B. Dahl. An improved merge–split sampler for conjugate Dirichlet process mixture models. Department of Statistics Technical Report 1086, University of Wisconsin at Madison, November 2003.

[56] J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log–linear models. *Annals of Mathematical Statistics*, 43(5):1470–1480, 1972.

[57] J. Dauwels and H.-A. Loeliger. Phase estimation by message passing. In *International Conference on Communications*, volume 1, pages 523–527, 2004.

[58] A. P. Dawid, U. Kjærulff, and S. L. Lauritzen. Hybrid propagation in junction trees. In *Advances in Intelligent Computing*, pages 87–97, 1995.

[59] A. P. Dawid and S. L. Lauritzen. Hyper Markov laws in the statistical analysis of decomposable graphical models. *Annals of Statistics*, 21(3):1272–1317, 1993.

[60] M. De Iorio, P. Müller, G. L. Rosner, and S. N. MacEachern. An ANOVA model for dependent random measures. *Journal of the American Statistical Association*, 99(465):205–215, March 2004.

[61] F. De la Torre and M. J. Black. Robust parameterized component analysis: Theory and applications to 2D facial modeling. In *European Conference on Computer Vision*, pages 653–669, 2002.

[62] S. Della Pietra, V. Della Pietra, and J. Lafferty. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4): 380–393, April 1997.

[63] J. W. Demmel. *Applied Numerical Linear Algebra*. SIAM, Philadelphia, 1997.

[64] A. P. Dempster. Covariance selection. *Biometrics*, 28:157–175, March 1972.

[65] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[66] K. Deng and A. W. Moore. Multiresolution instance-based learning. In *International Joint Conference on Artificial Intelligence*, pages 1233–1239, 1995.

[67] J. Deutscher, M. Isard, and J. MacCormick. Automatic camera calibration from a single Manhattan image. In *European Conference on Computer Vision*, volume 4, pages 161–174, 2002.

[68] P. Diaconis and D. Freedman. On the consistency of Bayes estimates. *Annals of Statistics*, 14(1):1–26, 1986.

[69] K. Doksum. Tailfree and neutral random probabilities and their posterior distributions. *Annals of Statistics*, 2(2):183–201, 1974.

[70] A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, New York, 2001.

[71] A. Doucet, N. de Freitas, K. Murphy, and S. Russell. Rao–Blackwellised particle filtering for dynamic Bayesian networks. In *Uncertainty in Artificial Intelligence 16*, pages 176–183. Morgan Kaufmann, 2000.

[72] A. Doucet, S. Godsill, and C. Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10:197–208, 2000.

[73] A. Doucet, N. J. Gordon, and V. Krishnamurthy. Particle filters for state estimation of jump Markov linear systems. *IEEE Transactions on Signal Processing*, 49 (3):613–624, March 2001.

[74] B. Efron. The geometry of exponential families. *Annals of Statistics*, 6(2):362–376, 1978.

[75] B. Efron and C. Morris. Data analysis using Stein's estimator and its generalizations. *Journal of the American Statistical Association*, 70(350):311–319, June 1975.

[76] M. D. Escobar and M. West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90(430):577–588, June 1995.

[77] L. Fei-Fei, R. Fergus, and P. Perona. A Bayesian approach to unsupervised one-shot learning of object categories. In *International Conference on Computer Vision*, volume 2, pages 1134–1141, 2003.

[78] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories. In *CVPR Workshop on Generative Model Based Vision*, 2004.

[79] L. Fei-Fei and P. Perona. A Bayesian hierarchical model for learning natural scene categories. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 524–531, 2005.

[80] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, 2005.

[81] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning object categories from Google's image search. In *International Conference on Computer Vision*, volume 2, pages 1816–1823, 2005.

[82] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 264–271, 2003.

[83] T. S. Ferguson. A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1(2):209–230, 1973.

[84] T. S. Ferguson. Prior distributions on spaces of probability measures. *Annals of Statistics*, 2(4):615–629, 1974.

[85] P. W. Fieguth, W. C. Karl, and A. S. Willsky. Efficient multiresolution counterparts to variational methods for surface reconstruction. *Computer Vision and Image Understanding*, 70(2):157–176, May 1998.

[86] P. W. Fieguth, W. C. Karl, A. S. Willsky, and C. Wunsch. Multiresolution optimal interpolation and statistical analysis of TOPEX/POSEIDON satellite altimetry. *IEEE Transactions on Geoscience and Remote Sensing*, 33(2):280–292, March 1995.

[87] M. A. T. Figueiredo and A. K. Jain. Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3): 381–396, March 2002.

[88] M. Fink and P. Perona. Mutual boosting for contextual inference. In *Neural Information Processing Systems 16*. MIT Press, 2004.

[89] M. A. Fischler and R. A. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, 22(1):67–92, January 1973.

[90] G. D. Forney, Jr. The Viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, March 1973.

[91] D. A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, New Jersey, 2003.

[92] A. B. Frakt, H. Lev-Ari, and A. S. Willsky. A generalized Levinson algorithm for covariance extension with application to multiscale autoregressive modeling. *IEEE Transactions on Information Theory*, 49(2):411–424, February 2003.

[93] W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):891–906, September 1991.

[94] W. T. Freeman and E. C. Pasztor. Markov networks for low–level vision. Technical Report 99-08, MERL, February 1999.

[95] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low–level vision. *International Journal of Computer Vision*, 40(1):25–47, 2000.

[96] B. J. Frey. Extending factor graphs so as to unify directed and undirected graphical models. In *Uncertainty in Artificial Intelligence 19*, pages 257–264. Morgan Kaufmann, 2003.

[97] B. J. Frey and N. Jojic. Transformation–invariant clustering using the EM algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(1): 1–17, January 2003.

[98] B. J. Frey and N. Jojic. A comparison of algorithms for inference and learning in probabilistic graphical models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(9):1392–1416, September 2005.

[99] B. J. Frey, R. Koetter, and N. Petrovic. Very loopy belief propagation for unwrapping phase images. In *Neural Information Processing Systems 14*, pages 737–743. MIT Press, 2002.

[100] B. J. Frey, F. R. Kschischang, and P. G. Gulak. Concurrent turbo–decoding. In *International Symposium on Information Theory*, page 431, 1997.

[101] B. J. Frey and D. J. C. MacKay. A revolution: Belief propagation in graphs with cycles. In *Neural Information Processing Systems 10*, pages 479–485. MIT Press, 1998.

[102] R. G. Gallager. *Low-Density Parity-Check Codes*. MIT Press, Cambridge, 1963.

[103] D. M. Gavrila. The visual analysis of human movement: A survey. *Computer Vision and Image Understanding*, 73(1):82–98, 1999.

[104] D. Geiger, D. Heckerman, H. King, and C. Meek. Stratified exponential families: Graphical models and model selection. *Annals of Statistics*, 29(2):505–529, April 2001.

[105] A. E. Gelfand, A. Kottas, and S. N. MacEachern. Bayesian nonparametric spatial modeling with Dirichlet process mixing. *Journal of the American Statistical Association*, 100(471):1021–1035, September 2005.

[106] A. E. Gelfand and A. F. M. Smith. Sampling–based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410):398–409, June 1990.

[107] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman & Hall, 2004.

[108] 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.

[109] Z. Ghahramani. Non-parametric Bayesian methods. Tutorial at *Uncertainty in Artificial Intelligence*, July 2005.

[110] Z. Ghahramani and M. J. Beal. Propagation algorithms for variational Bayesian learning. In *Neural Information Processing Systems 13*, pages 507–513. MIT Press, 2001.

[111] Z. Ghahramani and M. I. Jordan. Factorial hidden Markov models. *Machine Learning*, 29:245–273, 1997.

[112] S. Ghosal, J. K. Ghosh, and R. V. Ramamoorthi. Posterior consistency of Dirichlet mixtures in density estimation. *Annals of Statistics*, 27(1):143–158, 1999.

[113] J. K. Ghosh and R. V. Ramamoorthi. *Bayesian Nonparametrics*. Springer-Verlag, New York, 2003.

[114] W. R. Gilks and C. Berzuini. Following a moving target–Monte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society, Series B*, 63:127–146, 2001.

[115] W. R. Gilks, A. Thomas, and D. J. Spiegelhalter. A language and program for complex Bayesian modelling. *The Statistician*, 43(1):169–177, 1994.

[116] J. Goldberger and S. Roweis. Hierarchical clustering of a mixture model. In *Neural Information Processing Systems 17*, pages 505–512. MIT Press, 2005.

[117] S. Goldwater, T. L. Griffiths, and M. Johnson. Interpolating between types and tokens by estimating power-law generators. In *Neural Information Processing Systems 18*. MIT Press, 2006.

[118] G. H. Golub and C. F. Van Loan. *Matrix Computations*. John Hopkins University Press, Baltimore, 1996.

[119] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non–Gaussian Bayesian state estimation. *IEE Proc. F*, 140(2):107–113, April 1993.

[120] A. G. Gray and A. W. Moore. Very fast multivariate kernel density estimation via computational geometry. In *Joint Stat. Meeting*, 2003.

[121] P. J. Green and S. Richardson. Modelling heterogeneity with and without the Dirichlet process. *Scandinavian Journal of Statistics*, 28:355–375, 2001.

[122] J. E. Griffin and M. F. J. Steel. Order–based dependent Dirichlet processes. *Journal of the American Statistical Association*, 101(473):179–194, March 2006.

[123] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101:5228–5235, 2004.

[124] P. Hall, G. S. Watson, and J. Cabrera. Kernel density estimation with spherical data. *Biometrika*, 74(4):751–762, 1987.

[125] T. S. Han. Linear dependence structure of the entropy space. *Information and Control*, 29(4):337–368, December 1975.

[126] X. He, R. S. Zemel, and M. A. Carreira-Perpiñán. Multiscale conditional random fields for image labeling. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 695–702, 2004.

[127] D. Heath and W. Sudderth. De Finetti's theorem on exchangeable variables. *American Statistician*, 30(4):188–189, 1976.

[128] D. Heckerman. A tutorial on learning with Bayesian networks. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 301–354. MIT Press, 1999.

[129] S. Helmer and D. G. Lowe. Object class recognition with many local features. In *CVPR Workshop on Generative Model Based Vision*, 2004.

[130] H. V. Henderson and S. R. Searle. On deriving the inverse of a sum of matrices. *SIAM Review*, 23(1):53–60, January 1981.

[131] L. D. Hernández and S. Moral. Mixing exact and importance sampling propagation algorithms in dependence graphs. *Int. J. Intelligent Systems*, 12:553–576, 1997.

[132] T. Heskes. Stable fixed points of loopy belief propagation are minima of the Bethe free energy. In *Neural Information Processing Systems 15*, pages 343–350. MIT Press, 2003.

[133] T. Heskes. On the uniqueness of loopy belief propagation fixed points. *Neural Computation*, 16:2379–2413, 2004.

[134] T. Heskes, K. Albers, and B. Kappen. Approximate inference and constrained optimization. In *Uncertainty in Artificial Intelligence 19*, pages 313–320. Morgan Kaufmann, 2003.

[135] T. Heskes and O. Zoeter. Expectation propagation for approximate inference in dynamic Bayesian networks. In *Uncertainty in Artificial Intelligence 18*, pages 216–223. Morgan Kaufmann, 2002.

[136] T. Heskes, O. Zoeter, and W. Wiegerinck. Approximate expectation maximization. In *Neural Information Processing Systems 16*. MIT Press, 2004.

[137] G. E. Hinton. Products of experts. In *International Conference on Artificial Neural Networks*, volume 1, pages 1–6, 1999.

[138] G. E. Hinton. Training products of experts by minimizing contrastive divergence. Technical Report 2000-004, GCNU, 2000.

[139] G. E. Hinton, Z. Ghahramani, and Y. W. Teh. Learning to parse images. In *Neural Information Processing Systems 12*, pages 463–469. MIT Press, 2000.

[140] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42:177–196, 2001.

[141] A. T. Ihler. *Inference in Sensor Networks: Graphical Models and Particle Methods*. PhD thesis, Massachusetts Institute of Technology, June 2005.

[142] A. T. Ihler, J. W. Fisher, R. L. Moses, and A. S. Willsky. Nonparametric belief propagation for self–localization of sensor networks. *IEEE Journal on Selected Areas in Communications*, 23(4):809–819, April 2005.

[143] A. T. Ihler, J. W. Fisher, and A. S. Willsky. Loopy belief propagation: Convergence and effects of message errors. *Journal of Machine Learning Research*, 6: 905–936, 2005.

[144] A. T. Ihler, E. B. Sudderth, W. T. Freeman, and A. S. Willsky. Efficient multiscale sampling from products of Gaussian mixtures. In *Neural Information Processing Systems 16*. MIT Press, 2004.

[145] M. Isard. PAMPAS: Real–valued graphical models for computer vision. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 613–620, 2003.

[146] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *European Conference on Computer Vision*, pages 343–356, 1996.

[147] H. Ishwaran and L. F. James. Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96(453):161–173, March 2001.

[148] H. Ishwaran and M. Zarepour. Markov chain Monte Carlo in approximate Dirichlet and beta two-parameter process hierarchical models. *Biometrika*, 87(2):371–390, 2000.

[149] H. Ishwaran and M. Zarepour. Dirichlet prior sieves in finite normal mixtures. *Statistica Sinica*, 12:941–963, 2002.

[150] H. Ishwaran and M. Zarepour. Exact and approximate sum–representations for the Dirichlet process. *Canadian Journal of Statistics*, 30:269–283, 2002.

[151] S. Jain and R. M. Neal. A split-merge Markov chain Monte Carlo procedure for the Dirichlet process mixture model. *Journal of Computational and Graphical Statistics*, 13(1):158–182, 2004.

[152] L. F. James. Poisson calculus for spatial neutral to the right processes. To appear in *Annals of Statistics*, 2006.

[153] A. H. Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, New York, 1970.

[154] W. H. Jefferys and J. O. Berger. Ockham's razor and Bayesian analysis. *American Scientist*, 80:64–72, 1992.

[155] J. K. Johnson, D. M. Malioutov, and A. S. Willsky. Walk–sum interpretation and analysis of Gaussian belief propagation. In *Neural Information Processing Systems 18*, pages 579–586. MIT Press, 2006.

[156] N. Jojic and B. J. Frey. Learning flexible sprites in video layers. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 199–206, 2001.

[157] M. J. Jones and J. M. Rehg. Statistical color models with application to skin detection. *International Journal of Computer Vision*, 46(1):81–96, 2002.

[158] M. I. Jordan, editor. *Learning in Graphical Models*. MIT Press, Cambridge, 1999.

[159] M. I. Jordan. Graphical models. *Statistical Science*, 19(1):140–155, 2004.

[160] M. I. Jordan. Dirichlet processes, Chinese restaurant processes and all that. Tutorial at *Neural Information Processing Systems*, December 2005.

[161] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.

[162] S. J. Julier and J. K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, March 2004.

[163] T. Kailath. A view of three decades of linear filtering theory. *IEEE Transactions on Information Theory*, 20(2):146–181, March 1974.

[164] T. Kailath, A. H. Sayed, and B. Hassibi. *Linear Estimation*. Prentice Hall, New Jersey, 2000.

[165] K. Kanazawa, D. Koller, and S. Russell. Stochastic simulation algorithms for dynamic probabilistic networks. In *Uncertainty in Artificial Intelligence 11*, pages 346–351. Morgan Kaufmann, 1995.

[166] H. J. Kappen and W. J. Wiegerinck. Mean field theory for graphical models. In D. Saad and M. Opper, editors, *Advanced Mean Field Methods*. MIT Press, 2001.

[167] S. M. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice Hall, New Jersey, 1993.

[168] J. F. C. Kingman. *Poisson Processes*. Oxford University Press, Oxford, 1993.

[169] G. Kitagawa. Non-Gaussian state space modeling of nonstationary time series. *Journal of the American Statistical Association*, 82(400):1032–1041, December 1987.

[170] U. Kjærulff. HUGS: Combining exact inference and Gibbs sampling in junction trees. In *Uncertainty in Artificial Intelligence 11*, pages 368–375. Morgan Kaufmann, 1995.

[171] D. Koller, U. Lerner, and D. Angelov. A general algorithm for approximate inference and its application to hybrid Bayes nets. In *Uncertainty in Artificial Intelligence 15*, pages 324–333. Morgan Kaufmann, 1999.

[172] V. Kolmogorov and M. J. Wainwright. On the optimality of tree–reweighted max–product message–passing. In *Uncertainty in Artificial Intelligence 21*, 2005.

[173] S. Konishi, A. L. Yuille, J. M. Coughlan, and S. C. Zhu. Statistical edge detection: Learning and evaluating edge cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(1):57–74, January 2003.

[174] P. Kovesi. MATLAB and Octave functions for computer vision and image processing. School of Computer Science, University of Western Australia. Available from `http://www.csse.uwa.edu.au/~pk/research/matlabfns/`.

[175] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum–product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, February 2001.

[176] J. B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.

[177] S. L. Lauritzen. *Graphical Models*. Oxford University Press, Oxford, 1996.

[178] S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B*, 50(2):157–224, 1988.

[179] M. Lavine. Some aspects of polya tree distributions for statistical modelling. *Annals of Statistics*, 20(3):1222–1235, 1992.

[180] M. Lavine. More aspects of polya tree distributions for statistical modelling. *Annals of Statistics*, 22(3):1161–1176, 1994.

[181] S. Lazebnik, C. Schmid, and J. Ponce. A maximum entropy framework for part-based texture and object recognition. In *International Conference on Computer Vision*, volume 1, pages 832–838, 2005.

[182] J. C. Liter and H. H. Bülthoff. An introduction to object recognition. *Zeitschrift für Naturforschung*, 53c:610–621, 1998.

[183] J. S. Liu and R. Chen. Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93(443):1032–1044, September 1998.

[184] J. S. Liu and C. Sabatti. Generalised Gibbs sampler and multigrid Monte Carlo for Bayesian computation. *Biometrika*, 87(2):353–369, 2000.

[185] J. S. Liu, W. H. Wong, and A. Kong. Covariance structure of the Gibbs sampler with applications to the comparisons of estimators and augmentation schemes. *Biometrika*, 81(1):27–40, 1994.

[186] J. S. Liu, W. H. Wong, and A. Kong. Covariance structure and convergence rate of the Gibbs sampler with various scans. *Journal of the Royal Statistical Society, Series B*, 57(1):157–169, 1995.

[187] A. Y. Lo. On a class of Bayesian nonparametric estimates: I. Density estimates. *Annals of Statistics*, 12(1):351–357, 1984.

[188] D. G. Lowe. Distinctive image features from scale–invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[189] M. R. Luettgen, W. C. Karl, and A. S. Willsky. Efficient multiscale regularization with applications to the computation of optical flow. *IEEE Transactions on Image Processing*, 3(1):41–64, January 1994.

[190] J. MacCormick and M. Isard. Partitioned sampling, articulated objects, and interface–quality hand tracking. In *European Conference on Computer Vision*, volume 2, pages 3–19, 2000.

[191] S. N. MacEachern. Dependent nonparametric processes. In *Proc. Section on Bayesian Statistical Science*, pages 50–55. American Statistical Association, 1999.

[192] D. J. C. MacKay. Introduction to Monte Carlo methods. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 175–204. MIT Press, 1999.

[193] S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, San Diego, 1999.

[194] K. V. Mardia and P. E. Jupp. *Directional Statistics*. John Wiley and Sons, New York, 2000.

[195] D. Marr and H. K. Nishihara. Representation and recognition of the spatial organization of three-dimensional shapes. *Proceedings of the Royal Society of London, Series B*, 200(1140):269–294, February 1978.

[196] J. Marroquin, S. Mitter, and T. Poggio. Probabilistic solution of ill–posed problems in computational vision. *Journal of the American Statistical Association*, 82 (397):76–89, March 1987.

[197] B. Marthi, H. Pasula, S. Russell, and Y. Peres. Decayed MCMC filtering. In *Uncertainty in Artificial Intelligence 18*, pages 319–326. Morgan Kaufmann, 2002.

[198] A. M. Martínez and R. Benavente. The AR face database. Technical Report 24, CVC, June 1998.

[199] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *British Machine Vision Conf.*, pages 384–393, 2002.

[200] R. D. Mauldin, W. D. Sudderth, and S. C. Williams. Polya trees and random distributions. *Annals of Statistics*, 20(3):1203–1221, 1992.

[201] R. J. McEliece, D. J. C. MacKay, and J. Cheng. Turbo decoding as an instance of Pearl's "Belief Propagation" algorithm. *IEEE Journal on Selected Areas in Communications*, 16(2):140–152, February 1998.

[202] R. J. McEliece and M. Yildirim. Belief propagation on partially ordered sets. In D. Gilliam and J. Rosenthal, editors, *Mathematical Systems Theory in Biology, Communication, Computation, and Finance*. Springer, 2002.

[203] G. McLachlan and D. Peel. *Finite Mixture Models*. John Wiley and Sons, New York, 2000.

[204] I. Mikic, M. Trivedi, E. Hunter, and P. Cosman. Human body model acquisition and tracking using voxel data. *International Journal of Computer Vision*, 53(3): 199–223, 2003.

[205] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[206] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, October 2005.

[207] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schzffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1):43–72, 2005.

[208] B. Milch, B. Marthi, S. Russell, D. Sontag, D. L. Ong, and A. Kolobov. BLOG: Probabilistic models with unknown objects. In *International Joint Conference on Artificial Intelligence 19*, pages 1352–1359, 2005.

[209] E. G. Miller and C. Chefd'hotel. Practical nonparametric density estimation on a transformation group for vision. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 114–121, 2003.

[210] E. G. Miller, N. E. Matsakis, and P. A. Viola. Learning from one example through shared densities on transforms. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 464–471, 2000.

[211] T. Minka and J. Lafferty. Expectation-propagation for the generative aspect model. In *Uncertainty in Artificial Intelligence 18*, pages 352–359, 2002.

[212] T. Minka and Y. Qi. Tree–structured approximations by expectation propagation. In *Neural Information Processing Systems 16*. MIT Press, 2004.

[213] T. P. Minka. Expectation propagation for approximate Bayesian inference. In *Uncertainty in Artificial Intelligence 17*, pages 362–369. Morgan Kaufmann, 2001.

[214] T. B. Moeslund and E. Granum. A survey of computer vision–based human motion capture. *Computer Vision and Image Understanding*, 81(3):231–268, 2001.

[215] B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19 (7):696–710, July 1997.

[216] P. Müller and F. A. Quintana. Nonparametric Bayesian data analysis. *Statistical Science*, 19(1):95–110, 2004.

[217] K. Murphy, A. Torralba, and W. T. Freeman. Using the forest to see the trees: A graphical model relating features, objects, and scenes. In *Neural Information Processing Systems 16*. MIT Press, 2004.

[218] K. Murphy and Y. Weiss. The factored frontier algorithm for approximate inference in DBNs. In *Uncertainty in Artificial Intelligence 17*, pages 378–385. Morgan Kaufmann, 2001.

[219] K. P. Murphy, Y. Weiss, and M. I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Uncertainty in Artificial Intelligence 15*, pages 467–475. Morgan Kaufmann, 1999.

[220] C. Musso, N. Oudjane, and F. Le Gland. Improving regularized particle filters. In A. Doucet, N. de Freitas, and N. Gordon, editors, *Sequential Monte Carlo Methods in Practice*, pages 247–271. Springer-Verlag, 2001.

[221] R. M. Neal. Bayesian mixture modeling. In *Maximum Entropy and Bayesian Methods 11*. Kluwer Academic, 1992.

[222] R. M. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, 2000.

[223] R. M. Neal. Density modeling and clustering using Dirichlet diffusion trees. In *Bayesian Statistics 7*, pages 619–629, 2003.

[224] R. M. Neal, M. J. Beal, and S. T. Roweis. Inferring state sequences for non-linear systems with embedded Hidden Markov Models. In *Neural Information Processing Systems 16*. MIT Press, 2004.

[225] R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. MIT Press, 1999.

[226] S. M. Omohundro. Five balltree construction algorithms. ICSI Technical Report TR-89-063, U.C. Berkeley, 1989.

[227] J. A. O'Sullivan. Alternating minimization algorithms: From Blahut-Arimoto to Expectation–Maximization. In A. Vardy, editor, *Codes, Curves, and Signals: Common Threads in Communications*, pages 173–192. Kluwer Academic, 1998.

[228] S. E. Palmer. *Vision Science: Photons to Phenomenology*. MIT Press, Cambridge, 1999.

[229] A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, New York, 1991.

[230] E. Parzen. On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076, 1962.

[231] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufman, San Mateo, 1988.

[232] S. Petrone and A. E. Raftery. A note on the Dirichlet process prior in Bayesian nonparametric inference with partial exchangeability. *Statistics & Probability Letters*, 36:69–83, 1997.

[233] J. Pitman. Combinatorial stochastic processes. Technical Report 621, U.C. Berkeley Department of Statistics, August 2002.

[234] J. Pitman and M. Yor. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 25(2):855–900, 1997.

[235] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[236] D. Ramanan and D. A. Forsyth. Finding and tracking people from the bottom up. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 467–474, 2003.

[237] C. E. Rasmussen. The infinite Gaussian mixture model. In *Neural Information Processing Systems 12*. MIT Press, 2000.

[238] C. E. Rasmussen and Z. Ghahramani. Occam's razor. In *Neural Information Processing Systems 13*, pages 294–300. MIT Press, 2001.

[239] R. A. Redner and H. F. Walker. Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, 26(2):195–239, April 1984.

[240] J. M. Rehg and T. Kanade. DigitEyes: Vision–based hand tracking for human–computer interaction. In *Proc. IEEE Workshop on Non–Rigid and Articulated Objects*, 1994.

[241] X. Ren and J. Malik. Learning a classification model for segmentation. In *International Conference on Computer Vision*, volume 1, pages 10–17, 2003.

[242] J. A. Rice. *Mathematical Statistics and Data Analysis*. Duxbury Press, Belmont, California, 1995.

[243] S. Richardson and P. J. Green. On Bayesian analysis of mixtures with an unknown number of components. *Journal of the Royal Statistical Society, Series B*, 59(4): 731–758, 1997.

[244] T. J. Richardson and R. L. Urbanke. The capacity of low-density parity-check codes under message-passing decoding. *IEEE Transactions on Information Theory*, 47(2):599–618, February 2001.

[245] I. Rish. Distributed systems diagnosis using belief propagation. In *Allerton Conference on Communication, Control, and Computing*, October 2005.

[246] G. O. Roberts and S. K. Sahu. Updating schemes, correlation structure, blocking and parameterization for the Gibbs sampler. *Journal of the Royal Statistical Society, Series B*, 59(2):291–317, 1997.

[247] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and documents. In *Uncertainty in Artificial Intelligence 20*, pages 487–494. AUAI Press, 2004.

[248] G-C. Rota. On the foundations of combinatorial theory: I. Theory of Möbius functions. *Z. Wahrscheinlichkeitstheorie*, 2:340–368, 1964.

[249] S. Roweis and Z. Ghahramani. A unifying review of linear Gaussian models. *Neural Computation*, 11:305–345, 1999.

[250] P. Rusmevichientong and B. Van Roy. An analysis of belief propagation on the turbo decoding graph with Gaussian densities. *IEEE Transactions on Information Theory*, 47(2):745–765, February 2001.

[251] D. Saad and M. Opper, editors. *Advanced Mean Field Methods*. MIT Press, Cambridge, 2001.

[252] L. K. Saul and M. I. Jordan. Exploiting tractable substructures in intractable networks. In *Neural Information Processing Systems 8*, pages 486–492. MIT Press, 1996.

[253] M. Seeger. Gaussian processes for machine learning. *International Journal of Neural Systems*, 14(2):1–38, 2004.

[254] J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4: 639–650, 1994.

[255] G. R. Shafer and P. P. Shenoy. Probability propagation. *Annals of Mathematics and Artificial Intelligence*, 2:327–351, 1990.

[256] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter sensitive hashing. In *International Conference on Computer Vision*, pages 750–757, 2003.

[257] R. N. Shepard. Multidimensional scaling, tree-fitting, and clustering. *Science*, 210:390–398, October 1980.

[258] S. E. Shimony. Finding MAPs for belief networks is NP-hard. *Artificial Intelligence*, 68(2):399–410, August 1994.

[259] K. Shoemake. Animating rotation with quaternion curves. In *Proc. SIGGRAPH*, pages 245–254, 1985.

[260] H. Sidenbladh and M. J. Black. Learning the statistics of people in images and video. *International Journal of Computer Vision*, 54:183–209, 2003.

[261] L. Sigal, S. Bhatia, S. Roth, M. J. Black, and M. Isard. Tracking loose–limbed people. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 421–428, 2004.

[262] L. Sigal, M. Isard, B. H. Sigelman, and M. J. Black. Attractive people: Assembling loose–limbed models using nonparametric belief propagation. In *Neural Information Processing Systems*, 2003.

[263] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall, London, 1986.

[264] E. P. Simoncelli, W. T. Freeman, E. H. Adelson, and D. J. Heeger. Shiftable multi–scale transforms. *IEEE Transactions on Information Theory*, 38(2):587–607, March 1992.

[265] J. M. Siskind, J. Sherman, I. Pollak, M. P. Harper, and C. A. Bouman. Spatial random tree grammars for modeling hierarchal structure in images. Submitted to *IEEE Transactions on Pattern Analysis and Machine Intelligence*, May 2004.

[266] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering objects and their location in images. In *International Conference on Computer Vision*, volume 1, pages 370–377, 2005.

[267] H. W. Sorenson and D. L. Alspach. Recursive Bayesian estimation using Gaussian sums. *Automatica*, 7:465–479, 1971.

[268] T. P. Speed and H. T. Kiiveri. Gaussian Markov distributions over finite graphs. *Annals of Statistics*, 14(1):138–150, March 1986.

[269] A. Srivastava, A. B. Lee, E. P. Simoncelli, and S. C. Zhu. On advances in statistical modeling of natural images. *Journal of Mathematical Imaging and Vision*, 18:17–33, 2003.

[270] B. Stenger, P. R. S. Mendonca, and R. Cipolla. Model–based 3D tracking of an articulated hand. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 310–315, 2001.

[271] B. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla. Filtering using a tree–based estimator. In *International Conference on Computer Vision*, pages 1063–1070, 2003.

[272] M. Stephens. Bayesian analysis of mixture models with an unknown number of components: An alternative to reversible jump methods. *Annals of Statistics*, 28 (1):40–74, 2000.

[273] M. A. Stephens. Techniques for directional data. Technical Report 150, Stanford Department of Statistics, November 1969.

[274] A. J. Storkey and C. K. I. Williams. Image modeling with position-encoding dynamic trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7):859–871, July 2003.

[275] J. Strain. The fast Gauss transform with variable scales. *SIAM J. SSC*, 12(5): 1131–1139, 1991.

[276] E. B. Sudderth. Embedded trees: Estimation of Gaussian processes on graphs with cycles. Master's thesis, Massachusetts Institute of Technology, February 2002.

[277] E. B. Sudderth, A. T. Ihler, W. T. Freeman, and A. S. Willsky. Nonparametric belief propagation. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 605–612, 2003.

[278] E. B. Sudderth, M. I. Mandel, W. T. Freeman, and A. S. Willsky. Visual hand tracking using nonparametric belief propagation. In *CVPR Workshop on Generative Model Based Vision*, June 2004.

[279] E. B. Sudderth, M. I. Mandel, W. T. Freeman, and A. S. Willsky. Distributed occlusion reasoning for tracking with nonparametric belief propagation. In *Neural Information Processing Systems 17*, pages 1369–1376. MIT Press, 2005.

[280] E. B. Sudderth, A. Torralba, W. T. Freeman, and A. S. Willsky. Learning hierarchical models of scenes, objects, and parts. In *International Conference on Computer Vision*, volume 2, pages 1331–1338, 2005.

[281] E. B. Sudderth, A. Torralba, W. T. Freeman, and A. S. Willsky. Depth from familiar objects: A hierarchical model for 3D scenes. To appear at the *IEEE Conf. on Computer Vision and Pattern Recognition*, June 2006.

[282] E. B. Sudderth, A. Torralba, W. T. Freeman, and A. S. Willsky. Describing visual scenes using transformed Dirichlet processes. In *Neural Information Processing Systems 18*. MIT Press, 2006.

[283] J. Sun, H. Shum, and N. Zheng. Stereo matching using belief propagation. In *European Conference on Computer Vision*, pages 510–524, 2002.

[284] W. Sun. *Learning the Dynamics of Deformable Objects and Recursive Boundary Estimation Using Curve Evolution Techniques*. PhD thesis, Massachusetts Institute of Technology, September 2005.

[285] R. Szeliski. Bayesian modeling of uncertainty in low–level vision. *International Journal of Computer Vision*, 5(3):271–301, 1990.

[286] S. C. Tatikonda and M. I. Jordan. Loopy belief propagation and Gibbs measures. In *Uncertainty in Artificial Intelligence 18*, pages 493–500. Morgan Kaufmann, 2002.

[287] Y. W. Teh. A Bayesian interpretation of interpolated Kneser-Ney. Technical Report TRA2/06, National University of Singapore School of Computing, February 2006.

[288] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. In *Neural Information Processing Systems 17*, pages 1385–1392. MIT Press, 2005.

[289] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. To appear in *Journal of the American Statistical Association*, 2006.

[290] Y. W. Teh and M. Welling. The unified propagation and scaling algorithm. In *Neural Information Processing Systems 14*, pages 953–960. MIT Press, 2002.

[291] J. B. Tenenbaum and W. T. Freeman. Separating style and content with bilinear models. *Neural Computation*, 12:1247–1283, 2000.

[292] J. M. Tenenbaum and H. G. Barrow. Experiments in interpretation-guided segmentation. *Artificial Intelligence*, 8:241–274, 1977.

[293] A. Thayananthan, B. Stenger, P. H. S. Torr, and R. Cipolla. Learning a kinematic prior for tree–based filtering. In *British Machine Vision Conf.*, 2003.

[294] S. Thrun, J. C. Langford, and D. Fox. Monte Carlo hidden Markov models. In *International Conference on Machine Learning*, pages 415–424, 1999.

[295] C. Tomasi, S. Petrov, and A. Sastry. 3D Tracking = Classification + Interpolation. In *International Conference on Computer Vision*, pages 1441–1448, 2003.

[296] G. Tomlinson and M. Escobar. Analysis of densities. Technical report, University of Toronto, November 1999.

[297] G. Tomlinson and M. Escobar. Analysis of densities. Talk given at the Joint Statistical Meeting, 2003.

[298] A. Torralba. Contextual priming for object detection. *International Journal of Computer Vision*, 53(2):169–191, 2003.

[299] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing features: Efficient boosting procedures for multiclass object detection. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 762–769, 2004.

[300] A. Torralba, K. P. Murphy, and W. T. Freeman. Contextual models for object detection using boosted random fields. In *Neural Information Processing Systems 17*. MIT Press, 2005.

[301] Z. Tu, X. Chen, A. L. Yuille, and S. C. Zhu. Image parsing: Unifying segmentation, detection, and recognition. In *International Conference on Computer Vision*, volume 1, pages 18–25, 2003.

[302] S. Ullman, M. Vidal-Naquet, and E. Sali. Visual features of intermediate complexity and their use in classification. *Nature Neur.*, 5(7):682–687, July 2002.

[303] S. Verdú and H. Poor. Abstract dynamic programming models under commutativity conditions. *SIAM Journal on Control and Optimization*, 25(4):990–1006, July 1987.

[304] P. Viola and M. J. Jones. Robust real–time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[305] M. J. Wainwright. *Stochastic Processes on Graphs with Cycles: Geometric and Variational Approaches*. PhD thesis, Massachusetts Institute of Technology, January 2002.

[306] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree–based reparameterization framework for analysis of sum–product and related algorithms. *IEEE Transactions on Information Theory*, 49(5):1120–1146, May 2003.

[307] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree–reweighted belief propagation algorithms and approximate ML estimation by pseudo–moment matching. In *Artificial Intelligence and Statistics 9*, 2003.

[308] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree consistency and bounds on the performance of the max–product algorithm and its generalizations. *Statistics and Computing*, 14:143–166, 2004.

[309] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. MAP estimation via agreement on trees: Message–passing and linear programming. *IEEE Transactions on Information Theory*, 51(11):3697–3717, November 2005.

[310] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. A new class of upper bounds on the log partition function. *IEEE Transactions on Information Theory*, 51(7): 2313–2335, July 2005.

[311] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. Technical Report 649, Department of Statistics, UC Berkeley, September 2003.

[312] M. J. Wainwright and M. I. Jordan. Semidefinite relaxations for approximate inference on graphs with cycles. In *Neural Information Processing Systems 16*. MIT Press, 2004.

[313] S. G. Walker, P. Damien, P. W. Laud, and A. F. M. Smith. Bayesian nonparametric inference for random distributions and related functions. *Journal of the Royal Statistical Society, Series B*, 61(3):485–509, 1999.

[314] C. S. Wallace and D. L. Dowe. MML clustering of multi-state, Poisson, von Mises circular and Gaussian distributions. *Statistics and Computing*, 10:73–83, 2000.

[315] D. Walther, U. Rutishauser, C. Koch, and P. Perona. Selective visual attention enables learning and recognition of multiple objects in cluttered scenes. *Computer Vision and Image Understanding*, 100:41–63, 2005.

[316] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *IEEE Transactions on Image Processing*, 3(5):625–638, September 1994.

[317] L. Wasserman. Asymptotic properties of nonparametric Bayesian procedures. In D. Dey, P. Müller, and D. Sinha, editors, *Practical Nonparametric and Semiparametric Bayesian Statistics*. Springer-Verlag, 1998.

[318] M. Weber, M. Welling, and P. Perona. Unsupervised learning of models for recognition. In *European Conference on Computer Vision*, pages 18–32, 2000.

[319] Y. Weiss. Correctness of local probability propagation in graphical models with loops. *Neural Computation*, 12:1–41, 2000.

[320] Y. Weiss. Comparing the mean field method and belief propagation for approximate inference in MRFs. In D. Saad and M. Opper, editors, *Advanced Mean Field Methods*. MIT Press, 2001.

[321] Y. Weiss and W. T. Freeman. Correctness of belief propagation in Gaussian graphical models of arbitrary topology. *Neural Computation*, 13:2173–2200, 2001.

[322] Y. Weiss and W. T. Freeman. On the optimality of solutions of the max–product belief–propagation algorithm in arbitrary graphs. *IEEE Transactions on Information Theory*, 47(2):736–744, February 2001.

[323] M. Welling, T. P. Minka, and Y. W. Teh. Structured region graphs: Morphing EP into GBP. In *Uncertainty in Artificial Intelligence 21*, 2005.

[324] M. Welling and Y. W. Teh. Linear response algorithms for approximate inference in graphical models. *Neural Computation*, 16:197–221, 2004.

[325] M. West. Mixture models, Monte Carlo, Bayesian updating and dynamic models. In *Computing Science and Statistics*, volume 24, pages 325–333, 1993.

[326] M. West, P. J. Harrison, and H. S. Migon. Dynamic generalized linear models and Bayesian forecasting. *Journal of the American Statistical Association*, 80(389): 73–83, March 1985.

[327] W. Wiegerinck. Variational approximations between mean field theory and the junction tree algorithm. In *Uncertainty in Artificial Intelligence 16*, pages 626–633. Morgan Kaufmann, 2000.

[328] W. Wiegerinck. Approximations with reweighted generalized belief propagation. In *Artificial Intelligence and Statistics 10*, 2005.

[329] W. Wiegerinck and T. Heskes. Fractional belief propagation. In *Neural Information Processing Systems 15*, pages 438–445. MIT Press, 2003.

[330] A. S. Willsky. Multiresolution Markov models for signal and image processing. *Proceedings of the IEEE*, 90(8):1396–1458, August 2002.

[331] J. Winn and C. M. Bishop. Variational message passing. *Journal of Machine Learning Research*, 6:661–694, 2005.

[332] Y. Wu, G. Hua, and T. Yu. Tracking articulated body by dynamic Markov network. In *International Conference on Computer Vision*, volume 2, pages 1094–1101, 2003.

[333] Y. Wu and T. S. Huang. Hand modeling, analysis, and recognition. *IEEE Signal Proc. Mag.*, pages 51–60, May 2001.

[334] Y. Wu, J. Y. Lin, and T. S. Huang. Capturing natural hand articulation. In *International Conference on Computer Vision*, 2001.

[335] E. P. Xing, M. I. Jordan, and S. Russell. A generalized mean field algorithm for variational inference in exponential families. In *Uncertainty in Artificial Intelligence 19*, pages 583–591. Morgan Kaufmann, 2003.

[336] C. Yanover and Y. Weiss. Approximate inference and protein–folding. In *Neural Information Processing Systems 16*, pages 1457–1464. MIT Press, 2003.

[337] J. S. Yedidia. An idiosyncratic journey beyond mean field theory. In D. Saad and M. Opper, editors, *Advanced Mean Field Methods*. MIT Press, 2001.

[338] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Generalized belief propagation. In *Neural Information Processing Systems 13*, pages 689–695. MIT Press, 2001.

[339] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millennium*. Morgan Kaufmann, 2002.

[340] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 51(7):2282–2312, July 2005.

[341] A. Yuille. CCCP algorithms to minimize the Bethe and Kikuchi free energies: Convergent alternatives to belief propagation. *Neural Computation*, 14:1691–1722, 2002.

[342] X. Zhu, Z. Ghahramani, and J. Lafferty. Time–sensitive Dirichlet process mixture models. Computer Science Technical Report CMU-CALD-05-104, Carnegie Mellon University, May 2005.

[343] A. Zisserman et al. Software for detection and description of affine covariant regions. Visual Geometry Group, Oxford University. Available from `http://www.robots.ox.ac.uk/~vgg/research/affine/`.

[344] O. Zoeter and T. Heskes. Gaussian quadrature based expectation propagation. In *Artificial Intelligence and Statistics 10*, pages 445–452, 2005.