Toward Versatile Structural Modification for Bayesian Nonparametric Time Series Models

Thomas Stepleton

CMU-RI-TR-10-16

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Robotics

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

May 2010

Thesis Committee:
Tai Sing Lee, *Chair*
Alexei Efros
Zoubin Ghahramani
Geoffrey Gordon
Shimon Ullman, *Weizmann Institute of Science*

Abstract

**Toward Versatile Structural Modification
for Bayesian Nonparametric Time Series Models**

Unsupervised learning techniques discover organizational structure in data, but to do so they must approach the problem with *a priori* assumptions. A fundamental trend in the development of these techniques has been the relaxation or elimination of the unwanted or arbitrary structural assumptions they impose. For systems that derive hidden Markov models (HMMs) from time series data, state-of-the-art techniques now assume only that the number of hidden states will be relatively small, a useful, flexible, and usually correct hypothesis.

With unwanted structural constraints mitigated, we investigate a flexible means of *introducing* new, useful structural assumptions into an advanced HMM learning technique, assumptions that reflect details of our prior understanding of the problem. Our investigation, motivated by the unsupervised learning of view-based object models from video data, adapts a Bayesian nonparametric approach to inferring HMMs from data [1] to exhibit biases for nearly block diagonal transition dynamics, as well as for transitions between hidden states with similar emission models. We introduce aggressive Markov chain Monte Carlo sampling techniques for posterior inference in our generalized models, and demonstrate the technique in a collection of artificial and natural data settings, including the motivating object model learning problem.

Contents

List of Figures

List of Tables

To Cathy Beck and Don Eckert

Summary of Contributions

This dissertation presents a number of original contributions to machine learning and vision research, including the following:

¶ — We introduce a very broad, very flexible class of Bayesian nonparametric realizations of hidden Markov models. This class is distinguished from prior work because it allows users to specify a wide range of structural characteristics that might be present in hidden Markov models that are inferred from data. The variety of structure that can be accommodated by our model family is unmatched by existing Bayesian nonparametric hidden Markov model methods. We focus on a particular type of structure: "sub-behavior" structure, or the presence of distinct behavioral regimes in the data. Besides learning dynamic models of the data, our technique segments the data by inferring which sub-behavior is active at any given time.

¶ — We introduce a new collection of efficient sampling techniques for the difficult inference challenges posed by our model family. Some of these may be used for other classes of Bayesian nonparametric hidden Markov models, while others are unique to the sub-behavior structure explored in this dissertation. The discrete optimization problem entailed by sub-behavior identification, for example, frustrates ordinary Bayesian nonparametric sampling-based approaches—our methods overcome its hurdles.

¶ — We anticipate a research problem in the study of biological and computational vision: namely, how a vision system can derive an indefinite number of distinct object models from the visual similarity and temporal persistence cues present in videos of objects. This problem attracts minimal study today, but will become increasingly pressing as we attempt to design and understand systems that can form rich, coherent ontologies of visual phenomena in an unsupervised way. We show how our framework can be adapted to incorporate fundamental cues that characterize distinct, individual objects in videos, cues that allow the successful unsupervised derivation of view-based object models from these videos.

More fundamentally, this dissertation contributes to the ongoing discussion, in machine learning and other fields, on how to design unsupervised learning methods to identify and exploit increasingly intricate structure in diverse data sets. The techniques described here are new, complementary tools in the expanding toolbox for creating and applying such methods.

Chapter 1

Concepts and Context, and a Guide

Imagine for a moment a healthy infant boy of about four and a half months. This time in his life is characterized by some of the most explosive developmental growth he will ever know. The early rush of post-natal weight gain is just beginning to taper below a pound per month; new experiences for him include laughter, rolling, and maybe even teeth. The sensory world is growing for him as well: basic language sounds are explored through imitation, and with new motor abilities, arms and hands reach, touch, and grasp. We might assume that the visual experience of our baby boy is also increasing in richness: after all, by now he can reliably distinguish colors and finer details in what he sees [3].

In fact, this is almost surely an understatement, since the fourth month marks not only an achievement in visual acuity but also the opening of a new frontier in visual understanding. Now, for the first time, our infant is beginning to see the world not as a mixture of color and light but as a society of distinct, discrete objects. He notices basic cues that indicate how his visual perceptions can be divided up into separate things. Coherent motion is among the first: parts of an object unify into a single whole if they are perceived to move together [4, 5]. In ensuing months, he will develop the ability to delineate objects based on other information, such as color and texture [6, 7]. Well within eight more months—that is, within the infant's first year of life—he is in possession of an essentially mature object perception apparatus: one capable of recognizing many familiar objects and learning many more [7].

Nearly all aspects of this accomplishment are awe-inspiring, but perhaps one of the most amazing individual feats is the creation and curation of visual object representations. At four and a half months, the taps open on a novel stream of information—observations of individual objects—for which the infant must derive a coherent and useful semantics. On a moment-to-moment basis, we might imagine then that the infant visual system determines the answers to questions like the following:

- Is the object seen at this moment novel or familiar?
- If it is a familiar object, how should this visual information be used to improve or change the visual representation that describes this object?
- If it is a novel object, how should the visual information be used to create a new object representation?

These questions about the organization of visual object information are formidable for a number of reasons. First of all, this problem is for the most part an unsupervised learning problem—while there are non-visual cues that accompany many objects, there is no way for a baby to tell for certain which object they are seeing at any given moment. Babies do not have access to object labels—when they see a toy giraffe, say, there is no side information to indicate that this visual experience should be used specifically to improve their conception of what a toy giraffe looks like and not (for example) a toy horse. The infant visual system must do data association on its own, at the scale of hundreds or thousands of objects. Furthermore, the number of objects that must be learned—a very useful tidbit of prior knowledge that is commonly guessed or conceded *a priori* to machine learning systems in a wide variety of unsupervised settings—is not known. Finally, the visual system must make the most of the information it receives, since many of our encounters with objects are chance events yielding incomplete information about visual appearance. A baby may see a car from a variety of angles, but he can hardly pick one up and examine it to get a good idea of what it looks like from all vantages.

Infants face all of these challenges when they organize their experiences of objects into an understanding of what objects look like, and yet they meet them, expertly and without apparent conscious effort, earning a mature object recognition system within months, after which they go on to learn more novel objects for the rest of their lives. This breathtaking ability is as intriguing as it is impressive. What is required of a system that can organize information in this way? How can a learning mechanism provide useful answers to questions like the three listed above without requiring an unrealistic amount of prior guidance? Can a mechanism that has abilities like these be realized on a computer? These are the questions that led to the investigations presented in this thesis.

1.1 What this thesis is actually about

This thesis explores a probabilistic model for time series. It is, in other words, mostly a machine learning thesis, an exploration of a general purpose method for finding latent structure in data. After such an enthusiastic characterization of infant visual learning, it may be surprising to discover that infants or infant vision or vision generally are not a key topic of this dissertation. Nevertheless, the remarkable visual object learning ability of human beings has remained a distant guiding light toward which we have steered the design of the techniques described in this document. The novel features of the methods we introduce were all conceived through meditations on what capabilities might be necessary for a system that handles the three questions about visual object information organization listed above.

Put briefly, we present a fairly general, fairly flexible elaboration on the hidden Markov model (HMM) [8]. HMMs have been used for decades to find structure in sequence data. They model the underlying cause of the data as a random walk among a discrete collection of noisy-data-emitting states, governed by a table of the probabilities of transitioning from one state to another. Within this framework, HMMs can indicate which parts of the sequence are caused by the same thing. Speech understanding is often cited as an application of HMMs: here, hidden states are presumed to emit the sounds of specific components of speech (usually phonemes). By inferring a random walk trajectory through the states that matches a recording of speech, it is possible to identify the sequence of phonemes in the audio [8]. HMMs can also be applied in situations where the semantics and dynamics of the hidden states are not known beforehand—instead, an inference algorithm (often the Baum-Welch algorithm [9]) is applied to find good ways to associate a fixed set hidden states with observations, as well as the transition probabilities between these hidden states. These latter applications fall squarely under the rubric of unsupervised learning: the objective is even more to discover the structural underpinnings of not just a single sequence (e.g. a speech sample) but of a data generating process (e.g. speech).

No unsupervised learning method starts completely from scratch. A system that attempts to find structure in data cannot consider all possible types of structure (there are far too many to enumerate, let alone explore). Instead, it examines the data with a particular class or classes of structure assumed *a priori*, and the work of learning involves determining good ways to fit the *a priori* structure to the data. Let us consider data clustering and the K-means algorithm. It is easy to recognize that the K parameter—the number of clusters to find in the data—is an important *a priori* structural stipulation. Additional structural assumptions are bound up in the within-cluster sum of squares objective function, which one could argue is best suited to circumstances where the data in clusters are assumed to be distributed according to normal distributions with spherical (i.e. isotropic) covariance.

The progress of unsupervised learning research is marked by a change in the nature and severity of structural assumptions that are built into the learning techniques. K-means was invented as far back as the 1950s [10] (and on numerous subsequent occasions); in the following decade, techniques arose that assumed that data within partitions could be distributed as something other than spherical normal distributions ([11], as identified by the now-famous Expectation-Maximization paper [12]). Meanwhile, several complementary efforts now give us options more flexible than direct specification of the number of clusters K—we can deduce this now through methods including various information criteria (e.g. AIC [13], BIC [14]) and, most importantly to this dissertation, Bayesian nonparametric approaches (e.g. [15]), about which much more will be said later. An optimistic view of these developments and others suggests that researchers are learning how to strip away extraneous structural assumptions in techniques for clustering, leaving in the end something close to the essence of the problem: that data tend to come in a relatively small number of groups, and that within these groups, the data tend to have something important in common.

A similar progression of improvement has taken place in time series modeling, with comparable developments. Most relevant to this thesis is a Bayesian nonparametric realization of hidden Markov models: the Infinite Hidden Markov Model (IHMM) [1, 16], which the work presented here generalizes. In the most immediately practical terms, the IHMM is like an ordinary HMM, but the procedure that "fits" it to data allows it to allocate a variable (but not overindulgent) number of states to describe the data. By contrast, the Baum-Welch algorithm requires you to specify this quantity *a priori*, which can be a difficult judgment for many datasets.

Within the progression of these developments, the contribution of this thesis is to provide a partial answer to the following question: given the progress in distilling the spurious structure out of Markov time series models, how can we go about putting some *desirable* structure back in? That is, we presume to know something more about the dynamic behavior of our data beyond the fact that it can be modeled by a Markov process. How can we take advantage of this knowledge?

As an example, consider an ordinary Western pop song. This time series seems suited to modeling with a hidden Markov model: combinations of notes are discrete entities yielding noisy observations (audio data), and the melody and harmony derives from a trajectory through these discrete "states." Very good, but we know more about pop music than that. Most songs in this category will feature a structure of distinctive, repeated sections—choruses, verses, and so on. Can we somehow incorporate this additional structure into our model of the song? There could be benefits: depending on how this knowledge is incorporated, inferring model parameters for a new song could automatically "segment" the song according to its phrasal structure, which is a useful kind of analysis. More subtly, a correct but not overly restrictive structural assumption ("pop songs come in sections") could help the inference system make the most out of the data it has. There are many different HMMs that could be used to describe a pop song, but one that does it well likely has an internal organization that reflects the song's phrasal structure. By avoiding interpretations of the song that lack this characteristic, an HMM learning system that is biased to seek out this structure may do a better job at coming up with a good model.

The approach taken by this dissertation is thoroughly Bayesian in nature. We will present a technique that imbues Bayesian nonparametric realizations of hidden Markov models with flexible, descriptive, structured priors, starting with the kind of "sectioned" arrangement we see in our pop song example and moving on to more elaborate structure. If the data are ambiguous, these priors can hint at the "right" kind of structure for our unsupervised learning system to discover. If there is enough data to disprove the initial structural assumptions, on the other hand, the prior will be more or less disregarded. This flexibility helps prevent our method from getting caught up in its own built-in misconceptions, which is perhaps a quaint way of characterizing an application of K-means with the "wrong" K, for example.

1.2 The epic backstory: motivations and concepts

So far we have expressed an abiding enthusiasm for infant object learning and offered a high-level characterization of the machine learning-related contributions of this dissertation. What on earth do these topics have to do with each other?

Object representations are the connection. Trivially, the human visual system must be using some kind of representation of the objects it learns to see. If we subscribe to the classic analytical program for understanding vision set forth by David Marr, then examining the nature of these representations is an important part of understanding human vision as a whole—specifically, for understanding the algorithmic basis by which the visual system solves some of the problems that it solves (like the three questions mentioned at the beginning of the chapter). Investigations along these lines take many forms, but one form involves proposing a particular kind of representation or algorithm for a vision problem and comparing the structure, performance, behavior, shortcomings or other attributes of the proposal with analogous properties of human vision systems. Good proposals are those motivating new and interesting comparisons that extend our insight into how human vision works.

We will see examples of this kind of investigation shortly, but before moving on we should remind readers that this thesis is not one of them! Instead, it examines a machine learning technique devised as a part of such an undertaking, one that more or less took on a life of its own. Nevertheless, we hope it might be of mild interest to vision researchers in addition to the machine learning community. Our motivating story continues...

¶ **View-based object models** — In computer vision, there are numerous approaches to representing the visual appearance of objects, and all but the simplest of these try to account for the many ways an object can look under different viewing conditions. These techniques range widely in complexity and are designed with different applications in mind. Some of the simpler ones attempt to isolate a few robust statistical measurements that have a good chance of being present whenever the object appears in visual data—a very desirable property for object detection (e.g. [17, 18]. These approaches often aren't very good for describing what objects look like, however, because the statistics themselves abbreviate or omit many details about object appearance, and "inverting" their construction from data can be difficult or impossible. By contrast, other methods attempt to construct full 3-D models of entire objects or even entire scenes. These elaborate models can be useful for detection, but they can also be used to synthesize the appearance of objects under new conditions for visual presentation [19, 20].

Falling somewhere between these extremes of complexity are what we shall call view-based object models. These approaches attempt to represent the appearance of an object with an ensemble of simpler representations, each dedicated to capturing a more limited range of possible visual phenomena associated with the object. These simpler representations are called views. View-based approaches are useful for dealing with variation in object appearance due to pose: since an object seldom looks the same from all angles, different views can capture the appearance of an object within specific, limited angular viewpoint ranges—a model of how a car looks might therefore have a view to capture its appearance from the front, a view for the rear, several for various oblique viewing angles, and so on.

¶ **View-based object models in computer vision** — View-based object models have been visited again and again in computer vision research. Early approaches constructed models called aspect graphs, where each view subtends a range of 3-D viewpoints (aspects) that are somehow deemed to be similar (see [21] for an early reference and [22] for a review). For the edge-based geometric models popular in early vision, "similarity" often had to do with the visibility and topological arrangement of edges and vertices. As edge-based view models fell out of favor, new view representation strategies took their place, along with new criteria for determining which views were needed in a model. We present some examples here to give a sense of the breadth of the development. In [23] and [24], views are represented by shape statistics like silhouette skeletons, and allocation of views determined by a similarity-based clustering. Active appearance models, which are linear models of appearance variation, make up the view representations in [25], though the view angles are specified *a priori*. The same is true of [26] and [27], though here the views are characterized by collections of image features in the first case and separate, view-specific, discriminative classifier-based object detectors in the second.

A recent effort of note is the work of Fei-Fei, Savarese, et al. on models of rigid objects [28, 29], where views comprise arrangements of object parts (which are themselves configurations of image patches). View angles are specified beforehand, though in [28], nearly all of the (many) views do not have training data during model learning, and so their contents must be inferred. This is possible thanks to the known geometric relationship between views, which allows both models to share information about the locations of parts and features.

A few systems employ temporal information to create object representations. In [30], features are tracked through image sequences; views are collections of features and the geometric relationships between features in rotationally adjacent views (which are spaced at prespecified view angles) are known. Much earlier, Seibert and Waxman explore constructing HMM-based view-based object representations in an unsupervised manner [31], an goal similar to the one that motivates the machine learning techniques explored in this dissertation. We will comment on this model in greater detail in the next subsection.

¶ **Connections to visual neuroscience** — The ventral visual stream in primate brains is a series of cortical areas centrally involved in visual object recognition. Roughly speaking, each of these areas is assumed to build increasingly high-level, abstract representations of the contents of visual scenes as information propagates between low-level areas at the rear of the brain (e.g. primary visual cortex) and more anterior high-level areas. We expect this information propagation to be bidirectional: low-level areas feed information forward into high-level areas, which in turn supply biases based on their interpretations of the data in the opposite direction, helping lower-level areas sharpen their representations (c.f. [32, 33]).

In macaque monkeys, the anterior end of the ventral stream is inferotemporal cortex (IT), a region of the temporal lobe. It is here (and in a few adjacent structures) that neural responses correlate with image structures complex enough to be considered object parts or even views of whole objects [34–36]. Some cells are tuned especially for faces [37, 38]; recent fMRI studies report regions of temporal cortex selectively activated by object categories (e.g. faces, body parts, inanimate objects) [39]. One study reports the presence of some IT neurons selective for 3-D structures as opposed to 2-D patterns [36].

Neurons in IT are often reported as relatively insensitive to the position and scale of stimuli but selective for limited ranges of angular viewpoint (e.g. [40, 41]). Booth and Rolls report the presence of viewpoint-invariant cells but acknowledge a majority of view-selective ones [42], while the aforementioned 3-D study reports that even cells sensitive to 3-D structure are specific to particular viewing angles. This selectivity has encouraged some authors to propose that IT employs a view-based approach to represent objects and object parts [40, 43–45].

A few papers suggest predictive behavior on the part of IT neurons. After several presentations of stimuli pairing visual patterns to which two IT cells are sensitive, either at the same time [46] or in sequence [47], these studies found that some cells would show activity above baseline when their "partner" cell's stimulus was shown—and then continue elevated activity for a stimulus-free delay interval before their preferred stimulus appeared. Although the investigators do not speculate on the relevance of their findings to object modeling, the ability to predict the future appearance of an object is a desirable property for a vision system that can track an object over time despite appearance changes.

¶ **So, why hidden Markov models?** — All of the view-based object models described in the cited computer vision articles express some kind of relationship between the views in the model. Almost always, this relationship is geometric: views are typically associated with viewing angles or angular ranges. Some models go even further: if a view consists of spatially arranged features, for example, then the model defines or implies some transformation between the features in one view and some of the same features (those that are still visible) in other views.

This is a logical approach to take, but it rests on the critical assumption that the object itself is rigid or near-rigid. [25] is perhaps an exception—the active appearance models that make up the object views permit some learned variation in the appearance of the object corresponding to shape changes like facial expressions. There are limits to this variation, however—a considerably articulated object would exhibit changes in appearance beyond the slight deformations that active appearance models are intended to handle.

A simpler view-based object model approach might express a more basic relationship between object views: probability of succession over time. Under the most straightforward realization of this strategy, object views are more or less independent entities, except the model is supplemented with some understanding of how views might appear in sequence in a video of the corresponding object. Although simpler, this representation still codes the kind of information that is useful for tracking an object in video data: information that allows specific predictions about how the object might look in the near future, so that the tracker won't waste time or risk errors searching incoming video frames for views of the object it is unlikely to see. Models that code 3-D structure allow for these predictions via imposing a range of probable future geometric transformations on the model during tracking; the simpler approach that we are elaborating now would simply assign probabilities to certain object views that might appear soon given views observed in the recent past.

As the reader has likely surmised, this thesis uses hidden Markov models as the basis for the simpler view-based object representation just proposed. Individual object views are expressed as hidden states in hidden Markov models. Information about the succession of object views manifests itself as the Markov transition matrix, which simply codes the probability of seeing a particular view of an object given the view seen immediately prior. The relationships between hidden states/object views and the observations or statistics actually derived from the video data are expressed in terms of the HMM's observation models: each view has one, and each one describes what statistics might be observed given the presence of the object in a configuration that corresponds to its view.

Held up against models like [28], which make extensive use of 3-D relationships between views to build models, this HMM-based approach has theoretical advantages and disadvantages:

1. The stricter near-rigid-body structural assumptions of models leveraging 3-D structure probably permit through geometric calculations better prediction and synthesis of what an object might look like from a novel or intermediate viewpoint; basic HMM-based models have only their constituent views and no intrinsic means of interpolating between or extrapolating from them.

2. These assumptions also permit the 3-D leveraging models to easily share statistical information between object views—this typically takes the form of shared features between the views, as in [26, 28, 29] and others. A basic HMM-based approach must learn each view more or less anew, and thus the representation contains some redundant information.

3. HMM-based approaches have greater flexibility (no pun intended) to learn models of objects that violate the structural assumptions used by 3-D leveraging approaches, including highly deformable objects (e.g. a Transformers robot toy). Nearly any kind of appearance change whatsoever[1] can be accommodated in an HMM approach—all that's needed is a state or states in the HMM that describe the appearance.

4. HMM-based approaches can readily learn and model idiosyncratic temporal relationships between object views that reflect the typical visual behavior of objects. A car, for example, is much more likely to rotate 10° about its vertical axis than 10° about its longitudinal axis, though from an *a priori* geometric perspective these rotations are equally significant. Better modeling of idiosyncrasies like these might yield better object appearance predictions while tracking.

These contrasts hold strictly only for the most straightforward 3-D structured and HMM-based modeling approaches—clearly, a little ingenuity in either camp could help make up for comparative shortcomings, or better yet, bring both kinds of model together somehow. We also remind the reader that this thesis will not examine the relative practical advantages and disadvantages of the two modeling strategies. We present this comparison simply to bring the nature of HMM-based approaches to view-based object models into clearer relief.

¶ **More motivation** — Now that we've summarized some of the background and traits of various view-based object modeling strategies, we can recount our interest in HMM-structured view-based object models a little more completely. Our original aim was to tackle a problem similar to the unsupervised object model learning problem solved by infants discovering the world of visual objects. We intended to develop a system capable

[1] Invisibility doesn't count.

of analyzing video data of natural scenes, extracting independently moving objects inside the data, and constructing view-based object models from these moving objects. Given the complexity of this task, we concluded that a basic but flexible object representation was necessary. The simplicity of a view-based representation seemed appropriate, particularly considering its long history in computer vision as well as its relevance to our understanding of biological vision systems. Given that natural movie objects cannot be relied upon to be rigid, that natural movies contain great volumes of data, and that these data were expected to be particularly noisy, the more elaborate, computationally demanding 3-D object models seemed a poor choice. We chose HMM-based approaches as a middle ground between model simplicity and expressiveness.

Unsupervised object extraction from videos is a topic that continues to attract some research, such as [48–50]. These investigations are limited for now, likely because the problem is difficult and even simple approaches are complex to design, time-consuming to implement, and computationally taxing. In what research exists, the criteria and methods for organizing visual data into object representations are usually provisional—based on a variety of indices and invented measures of uniqueness—and often limited to only a few objects. Object representations themselves are in most cases simple single-view models (as in [48]) or if they are more complex, they too are applied to learning only a few object models [49, 50]. With this in mind, we set out to build an important component of a system for unsupervised object extraction from videos, one that could construct HMM-structured view-based object models of multiple objects from segmented, unstructured video clips of the objects, one that reasoned in a principled way about how to organize the data into objects based on the evidence it was shown. The investigations that departed from this starting point are presented in this thesis.

1.2.1 Gestalt laws and HMM learning for view-based object models

Now that our object model learning task has been established as a problem of learning hidden Markov models, we can muse still further about the structure of our data—video clips of objects—and of the structure we hope for our learned object models to acquire. By integrating this kind of structural information as best we can into the learning mechanism, we can hope to do a better job of deriving object models than we might have done otherwise.

The first and most elementary cue is the temporal persistence and continuity of objects in our visual experience. Typically, objects do not rapidly flash in and out of existence—as long as we are looking toward them, they tend to remain present in our vision, although their appearance may change as time goes by.

A second important cue is the continuity of object appearance over time. Most changes in object appearance happen smoothly, particularly those due to movement or deformation: as a turning car rotates, for example, its appearance varies only slightly from one fraction of a second to the next. Exceptions to this rule might include lighting changes: turning off a light in a room changes the appearances of objects inside very rapidly. The majority of our visual experiences of objects, however, do not exhibit this kind of involuntary, abrupt change.

A third cue involves the similarity of object appearances. Although objects do change in appearance over time, in many cases, the possible appearances of an object have some common traits. Color is an example: most views of a can of Coca-Cola have at least some red in them; often, red predominates.

Not coincidentally, these three cues are rather direct realizations of Gestalt "laws" for visual perception [51]. The first cue relates most closely to the Gestalt principle of *proximity*, which dictates that perceptual data are more likely to be grouped into a whole if they occur near each other. In this case the proximity is temporal instead of spatial. The second cue, as hinted, reflects the law of *good continuation*: that we are more likely to group visual data if they appear in alignment. The kind of alignment most appropriate to our problem is the similarity of temporally adjacent visual measurements mentioned above. Finally, the third cue is effectively the Gestalt principle of *similarity*.

These cues can help an object model learning system determine which video data belong to which objects, and in what way. How can they be incorporated into techniques from inferring hidden Markov models from data? At last we come to questions this thesis addresses directly. Recall from the beginning of this section that unsupervised learning methods for HMMs have shown a steady decrease in the amount of *a priori* structure that must be specified by the user, and that having dispensed with these strictures it would be worthwhile to investigate putting useful structure back in. The object cues just enumerated are good examples of the kind of structure we would like to be able to impose.

In the end, using the techniques of this dissertation, we specified a prior probability distribution over hidden Markov models where HMMs with the following characteristics were most probable:

- A limited (but not fixed) number of hidden states almost surely accounts for most of the data drawn from the HMM.
- Hidden states are partitioned into a limited (but not fixed) number of exclusive groupings (or sub-behaviors) where
  - transitions between states in the same group are more probable than transitions between states in different groups,
  - transitions between states in the same group are additionally more probable when the observation models for the states (in other words, the object views) are similar, and
  - observation models for states in the same group are correlated; specifically, they have common traits corresponding to the global appearance characteristics of an object.

The first characteristic corresponds to the minimally structured prior entailed by models like the IHMM mentioned at the start of this chapter. The other two characteristics capture the three Gestalt principles cited above using new methods introduced here.

When data comprising segmented, unstructured video clips of objects are combined with this prior using Bayes' rule, the structure intrinsic to the prior results in a posterior distribution over HMMs that favor the following traits:

- Hidden states correspond to distinct, sensible object views for the various objects.
- Hidden states with observation models showing the same object are typically grouped into a sub-behavior that corresponds to that object.
- Observation models for states in a sub-behavior can be explained by a single shared "generating distribution" that codes common appearance traits of the corresponding object.
- Transitions between object views reflect the statistics of object appearance change observed in the data, or, where sparse or absent, the view similarity-based assumption coded in the prior.

We are not the first to propose view-based object models based on hidden Markov models, nor a system for learning them in an unsupervised way. As mentioned earlier, in [31], Seibert and Waxman considered the same problem in 1992. Their efforts were ambitious for the time—they, like us, attempt to derive the number of necessary hidden states and number of objects from the data. Nevertheless, there are fundamental differences between their methods and ours. Seibert and Waxman allocate hidden states/views without using temporal information—instead, they employ appearance-based clustering on the input data. Most of the states are shared between the object models, and the distinction between objects is derived almost exclusively from the dynamics of the object video (that is, the states traversed as the object's appearance changes over time) and seldom from the object appearance directly. There is no provision for incorporating prior assumptions about the transition relationships between object views. Finally, the mechanism for determining whether to create a new hidden state or object model is based on simple thresholding.

Eighteen years later, the method we describe jointly infers which hidden states to allocate and how to arrange these into separate object models, instead of making final decisions about the former before embarking on the latter. This inference is not based on thresholding but instead derives from a flexible, structured prior that can incorporate assumptions about dynamics based on appearance. Our approach can use both the appearance of individual views and the dynamic characteristics of view sequences to allocate views exclusively to individual object models. We consider approaches of comparable complexity to [31] in §6.4.9 and find that the advanced features of our model confer worthwhile advantages.

1.2.2 The specificity/invariance tradeoff

We now relate our motivating object model learning goal to an additional, important context: that of learning effective representations for visual object recognition. We do not employ our learned object models to recognition tasks in this thesis, which instead focuses on unsupervised means for organizing video data into an indefinite number of distinct object models, a complementary and perhaps overlooked task. Nevertheless, the considerations that motivate both research areas are very similar.

Most object recognition representation learning research recognizes a fundamental tradeoff between *specificity* and *invariance*. Object detection systems engineered for invariance

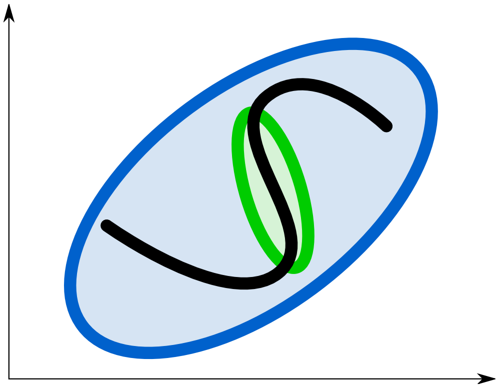


FIGURE 1.1: A cartoon depiction of the specificity/invariance tradeoff (§1.2.2) for a 2-D recognition problem. Let the black line be the set of valid appearances of an "object" (if you like, imagine that our camera takes a 2 × 1 pixel image). If an appearance measurement falls within the green ellipse, the first object recognition system will register a detection. It is overly specific, however, and fails to "go off" for most object appearances. The second system, identified by the blue ellipse, achieves total invariance at the expense of false positives—while it will register a detection for all possible measurements of the object, it will also register a detection for many other measurements besides. A view-based approach would attempt to build a good recognition system by using multiple high-specificity detectors to "cover" most valid object appearances (not shown).

are made to be sensitive to their target objects under a variety of viewing conditions. Unfortunately, the same sensitivity that makes their detections robust to changes in pose, illumination, and other factors also increases the rate of false positives by making it more likely for things that are not the object to "fool" the detector. Conversely, systems designed for specificity are sensitive to a smaller range of appearances, making false positives less likely but increasing the chance of false negatives. This tradeoff is portrayed graphically in Figure 1.1.

Systems that learn representations for visual object recognition are designed to explore or mitigate this tradeoff as best as possible, and hopefully to "carve out" a portion of appearance space that corresponds well to the target object. All of these systems do so by looking at visual training data that show an object or objects, but the ways they use this data differ. Stringer and Rolls [52] identify two learning strategies: *trace learning* and *continuous transformation learning*:

- Trace learning systems monitor changes in visual appearance within videos, building representations of the appearance variation of objects as they changes over time. Intuitively, if two visual appearance measurements happen in short succession, they probably came from the same object, even if they look fairly different.

- Continuous transformation learning systems, in contrast, use appearance similarity to "grow" object models: intuitively, if two visual appearance measurements look similar enough, they probably came from the same object. Applying this visual similarity judgment across a dataset allows a representation learning system to unite similar observations of the same object and "carve out" a range of visual appearances.

Trace learning systems require video data input, while continuous transformation learning systems do not. The learning approach related in this thesis uses trace learning as its primary cue for constructing view-based object models, although its incorporation of "generating distributions" to describe shared traits of object views gives it some ability to cluster views based on visual similarities. This capability is more akin to continuous transformation learning. In any case, a brief survey of several methods applying either strategy can help establish our approach in context.

The first widely-cited continuous transformation learning system is Fukushima's Neocognitron [53], a multi-layer neural network that learns to recognize line-drawn letters in spite of scaling and translation on the 2-D plane. These letters are learned in an unsupervised way, in the sense that there is no signal during training to indicate which letter is which—instead, carefully engineered competitive interaction between network units result in output layer units acting as "grandmother cells" uniquely associated with each letter. The basic architectural principles of the Neocognitron—layers in which units receive input from a spatially-localized collection of units in the previous layer, and local lateral inhibition to promote competitive interactions—are common to several subsequent approaches. Notably, Riesenhuber and Poggio study recognition of 3-D objects resembling bent wires [54], using a winner-take-all operation in certain layers to effect competition, while Stringer and Rolls employ a more conventional arrangement of inhibitory units to simultaneously learn multiple models of rotating polyhedra [52].

Neural networks are not the only option for continuous transformation learning. A number of papers attempt to identify useful representations of visual features (e.g. edges and corners, not whole objects) by proposing a bipartite interpretation of the image data. Visual measurements owe their variation to two factors: which feature is present, and the parameters of certain types of variation in that feature's appearance. Instantiations of this approach include bilinear models [55, 56] and more elaborate methods involving Lie operators [57]. Occasionally these approaches is extended to whole objects, but almost always on a very provisional basis. Finally, a few investigations simply consider numerous single-view recognition system "building blocks" (such as input features [58] and discriminative classifier kernels [59]) and employ supervised learning approaches that attempt to identify the best components to use for particular recognition tasks.

Neural network approaches to trace learning typically include variations on decay dynamics in unit activation or delay line-based network architectures so that activation from earlier stimuli can continue to have an effect on learning effects caused by subsequent data. Ideally, this leads to association of both stimuli (and their temporal neighbors in the data) with a singular object. Here, too, competitive interactions between units is necessary for the networks to evolve the capability to distinguish objects. Földiák presents this early on with a network that learns simple line features [60], with later efforts by Becker [61], Einhäuser [62], and Stringer et al. [63] directed toward more complex objects. Closely related to these are the recent efforts of Mobahi et al. [64], which combine deep belief networks with a training rule that attempts to build similar representations of temporally-adjacent inputs and distinct representations of non-adjacent ones, a slightly different realization of the competition inherent in the cited neural network approaches.

As with continuous transformation learning, efforts directed at image features rather than whole objects exist. Wiskott and Sejnowski build a learning framework that distinguishes between characteristics of image sequences that change slowly over time from those that change quickly—the idea being that the former are more likely to correspond to object features, which are presumably present for extended periods, while the latter correspond to transient changes in viewing conditions [65]. Cadieu and Olshausen consider a similar decomposition [66], but after observing that fast-changing image measurements are often oscillatory in nature, elect to use complex basis functions in their model.

Finally, there are a few object recognition representation learning systems that use HMMs to model certain aspects of temporal variation in object appearance. Both Dean [67] and George and Hawkins [68] employ multi-layer representations with units in each layer using HMMs to model the temporal behavior of groups of units in previous layers. These intricate representations do not learn view-based object models explicitly, and often parameters like the number of HMM states in each unit are specified beforehand. Most applications to moving images involve learning translation, scale, and slight deformation invariance for a set of stick-figure drawings introduced by George and Hawkins in an earlier paper—the purpose of HMMs in these efforts is to model the dynamics of these variations within the range of stick-figure data for which a unit is sensitive. It may be most apt, then, to regard the use of HMMs in these efforts as a bid to more explicitly determine the kinds of invariance to which individual units are tuned, and not as an overarching strategy for visual object modeling.

All of the techniques mentioned here use some of the same intuitions that the approaches in this thesis use to distill object models from video. Most of them are applied to actual object recognition; our system is not. Some employ neurally plausible computations; we do not investigate how our approach might be realized in cortical circuits. Nevertheless, the mechanisms we will describe have important new capabilities. Of the systems cited above that confront the problem of building representations of multiple objects at once, none have a really satisfactory means of explicitly learning *distinct* object models in an unsupervised way. Several employ a supervised learning paradigm [58, 59, 64, 67, 68]; nearly all of the rest either give the learning systems the number of objects beforehand [60, 61] or specify a large number of output layer units with the expectation that at least one unit (and often more) will reliably "fire" if and only if a particular object is present [52, 62, 63]. Note that identifying *which* units are the good ones requires comparisons between unit responses and "ground truth" object labels for a presentation of target objects—this judgment is arguably not quite the same thing as unsupervised learning of individual object models.[2] A standout is the Neocognitron [53], which appears to expect one and only one active output unit per object; that said, the paper gives the impression that this behavior requires very careful tuning of the network parameters.

By contrast, our methods are designed to learn an explicit collection of object models. With data from seven objects in the training data, we intend our system to learn seven object models; with ten, ten; and so on. This is a problem that our brains solve: we have hypotheses about which visual objects are distinct, and we recognize objects for the most part as discrete, semantically distinct entities. We do not guess beforehand how many objects it is necessary to know, nor do we require ground truth to vet candidate object representations. We expect that solving the problem of constructing clearly delineated object representations will be of increased relevance to building and understanding vision systems in the future, particularly those that "mine" objects from real-world experience.

Several approaches cited above also rely heavily on competition for useful organization of object models. It is worth noting that our approach employs similar dynamics as it learns: object views and even entire object models come from the system navigating the

[2] A controversial statement, to be sure. We don't mean to imply that these approaches are not learning anything *per se*, but perhaps instead that the completing steps of proposing the distinct objects that make up the world, and associating data with those objects, is important—and missing.

tradeoff between describing input data accurately and describing it parsimoniously. Competition also promotes parsimony by preventing multiple neural network units (or sparse belief network representations, or conceptually similar computational entities) from representing the same thing. That said, the connection between competition and parsimony is sometimes not made explicit in the object recognition representation learning literature.

1.3 Bayesian nonparametric models

The technique described in this dissertation derives hidden Markov models from data by exploiting a probabilistic framework based on Bayesian nonparametric models. Here we attempt to clarify what that means and compare our efforts to contemporary related work.

Traditionally, nonparametric statistical models have been those where the properties of the model originate in a rather immediate-seeming way from data associated with the model, rather than from parameters derived from the data or specified *a priori*. Regrettably, few descriptions dare to get much more specific than this (note similar caution in e.g. [69]), since use of the term has historically been somewhat cavalier. Regardless, an illustrative example of a nonparametric model is the familiar kernel density estimator of probability distribution functions [70], where the mean, variance, and other properties of the estimated PDF are functions of all the data samples associated with the estimate.

In light of nonparametric models' close relationship to the data, it is common to observe that "nonparametric" is a misnomer and that the data are themselves the (many) model parameters. This leads to the approach of Bayesian nonparametric models, which are explicitly characterized as having an infinite number of parameters. In this setting, the finite data used with the model provide evidence for what these parameters might be. What makes these models practical are clever formulations that do not require the actual representation or storage of infinite collections of parameters in applied settings—usually the amount of information that must be stored is of the same order as the amount of available data, and in some cases far less is necessary. In a Bayesian framework, once the data contributes the evidence it can, a prior expressed over the parameters also pitches in to help answer the questions about the parameter values that must be answered in the course of everyday operations like inference.

Several Bayesian nonparametric models are related to familiar finite-dimensional parametric models. The Dirichlet process is a countably infinite dimensional generalization of the

Dirichlet distribution where finite marginal distributions are Dirichlet distributions [71]; likewise, in a Gaussian process, finite marginal distributions are Gaussian distributions [72]. Indeed, an exciting new paper on extending parametric models to fully Bayesian nonparametric models has recently appeared [73]. As this paper notes, however, other models have appeared where the connection to finite, parametric distributions is obscure or absent.

Our technique is derived from the hierarchical Dirichlet process [1], a nonparametric Bayesian model we describe in considerable detail in Chapter 2. The same paper that introduced the hierarchical Dirichlet process also introduced the popular formulation of the infinite hidden Markov model that we mentioned at the start of §1.1—referred to in the paper as the HDP-HMM. This is the model we extend to achieve the data organization goals we set out in this chapter.

Our model is not the first to extend the IHMM, although to our knowledge there have only been a very few, specialized efforts. Fox et al. introduced a model where transitions between a hidden state and itself are more likely than they are in the original IHMM, a useful characteristic for dynamic processes that remain in the same state for extended periods [74]. This same group then uses this model as a basis for learning switching linear dynamical systems from data: each state in the HMM is associated with a linear model that can describe a limited degree of temporal variation in the data [75]. This approach is capable of describing a wide range of dynamic behavior in the data: effectively, the linear models describe small changes in the behavior while transitions between states capture broader variation. This idea of switching between dynamic models could also be related to the model we introduce in Chapter 3, since both could be said to isolate distinct dynamic behaviors in the data. However, the goals and methods for doing so are very different for both models: our model, by contrast, imposes a sub-behavior structure on the discrete states themselves.

An important area of related work is the effort to develop models that induce useful dependence among multiple random measures. For the purposes of this discussion, the random measures of interest are probability distributions over a countably infinite set of outcomes—as Chapter 2 will demonstrate, these measures are what one draws from a Dirichlet process. These measures are useful for mixture models, for example, where the outcomes are parameters associated with the mixture components, and the probability distribution specifies the components' mixing proportions. It can be useful for these measures to vary over time: when finding topic clusters in a document database, for example, the importance (i.e. probabilities) of various topics (mixing components) can wax and wane to match various trends. This temporal dependence is achieved in various ways in a number of papers, including [76–80]. Additional research now pursues dependence structure between random measures that varies with spatial location or other kinds of covariates, including [81–85].

The means by which these models and others achieve their desired dependences are varied, intricate, and interesting in their own right, but beyond the scope (or even the already taxed capacity) of this document. Whether any of these new approaches may be adapted to devising novel, flexible techniques for learning hidden Markov models from data is an open and exciting research question. That said, the means by which we modify the IHMM in this dissertation is itself a means of generating random measures with useful dependence, and it may be interesting to determine whether it, too, could be applied to problems like those in the papers cited above.

One final comment completes our overview of related work in Bayesian nonparametric models. We note that Dunson and colleagues have devised Bayesian nonparametric schemes for clustering time series [86] and even identifying sub-behavior structure in individual time series [79, 87]. Their approach differs considerably from the one we will espouse here, however. For the latter task, it is necessary to divide the time series data into short clips for which HMMs with a preset number of states are fit; a time-dependent Dirichlet process model expresses a discrete prior over the space of such HMMs, and thus those clips that are adjacent may be expected to draw the same HMM from the prior (and thus deemed to engage in the same sub-behavior). Our approach does not require breaking data into clips for sub-behavior identification, a problematic proposition both for short sequences and sequences that change slowly. In these cases, the prespecified duration of clips and the locations of clip boundaries would likely have an impact on the inference outcome. Furthermore, our method permits the specification of additional structure beyond the presence of sub-behaviors, and it does not mandate a preset number of states for each sub-behavior.

1.4 Goals, and a roadmap for readers

The goals of this dissertation document are as follows:

1. Introduce a new unsupervised learning scheme for hidden Markov models that allows the user to introduce useful structural assumptions, and that flexibly merges these assumptions with evidence from the data to construct learned models.

2. Describe effective inference methods that allow these techniques to be applied to real-world problems, and that do not require end-users to specify burdensome amounts of information about their problem (e.g. derivatives of probability distribution functions).

3. Provide sufficient detail on these methods to allow readers to implement them themselves.

4. Provide a tutorial introduction to Bayesian nonparametric probabilistic models based on the Dirichlet process.

5. Demonstrate these techniques on a selection of unsupervised learning problems, including an object model learning task that captures the motivating challenges behind this thesis (§1.2).

This thesis was written for an imaginary reader—a very patient second-year student at the Robotics Institute at Carnegie Mellon University with a machine learning research focus. It is assumed that this reader has a working familiarity with Bayesian probabilistic methods and graphical models, as well as hidden Markov models and basic concepts in unsupervised learning. Some understanding of sampling and why it is useful for inference would be most helpful. Beyond this background, adequate explanations for remaining terms left unexplained in the text could at the time of this writing be supplied by Wikipedia.

The tutorial, theoretical, and practical information in this dissertation is mixed together in a narrative exposition that very roughly mirrors the path followed by the development of the research itself. Although we feel that this narrative provides a convenient framework for introducing and describing all of the concepts and methods encompassed by the work, readers may appreciate being able to approach the material in a way that suits their interests and time. Accordingly, we offer the following focus areas for those wish to pick and choose from the contents of the following chapters.

- A tutorial introduction to Bayesian nonparametric methods derived from the Dirichlet process, their application to time series modeling, an overview of our generalizations of existing time series models, and some theoretical considerations: Chapter 2, Chapter 3 introduction, §3.1, §3.2, §3.4.9, Chapter 5 introduction, §5.1, §5.2, §5.3.1.

- A look at the theoretical and practical contributions of this thesis for readers already familiar with Dirichlet process-derived time series models: Chapter 3 introduction, §3.1, §3.2, §3.4.5, §3.4.6, Chapter 4, Chapter 5 (especially §5.1 and §5.3.7).

- For those who wish to implement the methods of this dissertation: §3.4, §4.5 (but useful information in §4.3, §4.4), §5.3 (especially §5.3.7).

We note that in service of the second goal listed above, there are several portions of this document that delve into considerable technical detail. Readers who are not interested in the considerations necessary for implementation but who wish to get a glimpse of the clockwork may enjoy reading the opening portions of the sections and subsections listed in the following "gist tour": §2.1, §2.2 *but skip* §2.2.1, Figure 2.5, §2.4, §3.1, §3.2, §3.4.5, §3.4.6, §3.4.3, §5.3.7, §4.1, §4.2, Chapter 5 introduction, §5.1, §5.2, §5.3.1, and of course the results in Chapter 6.

Finally, we humbly submit that this document contains the most evocative characterization of the famous Chinese Restaurant Process ever written. This appears at the end of §2.2.5.

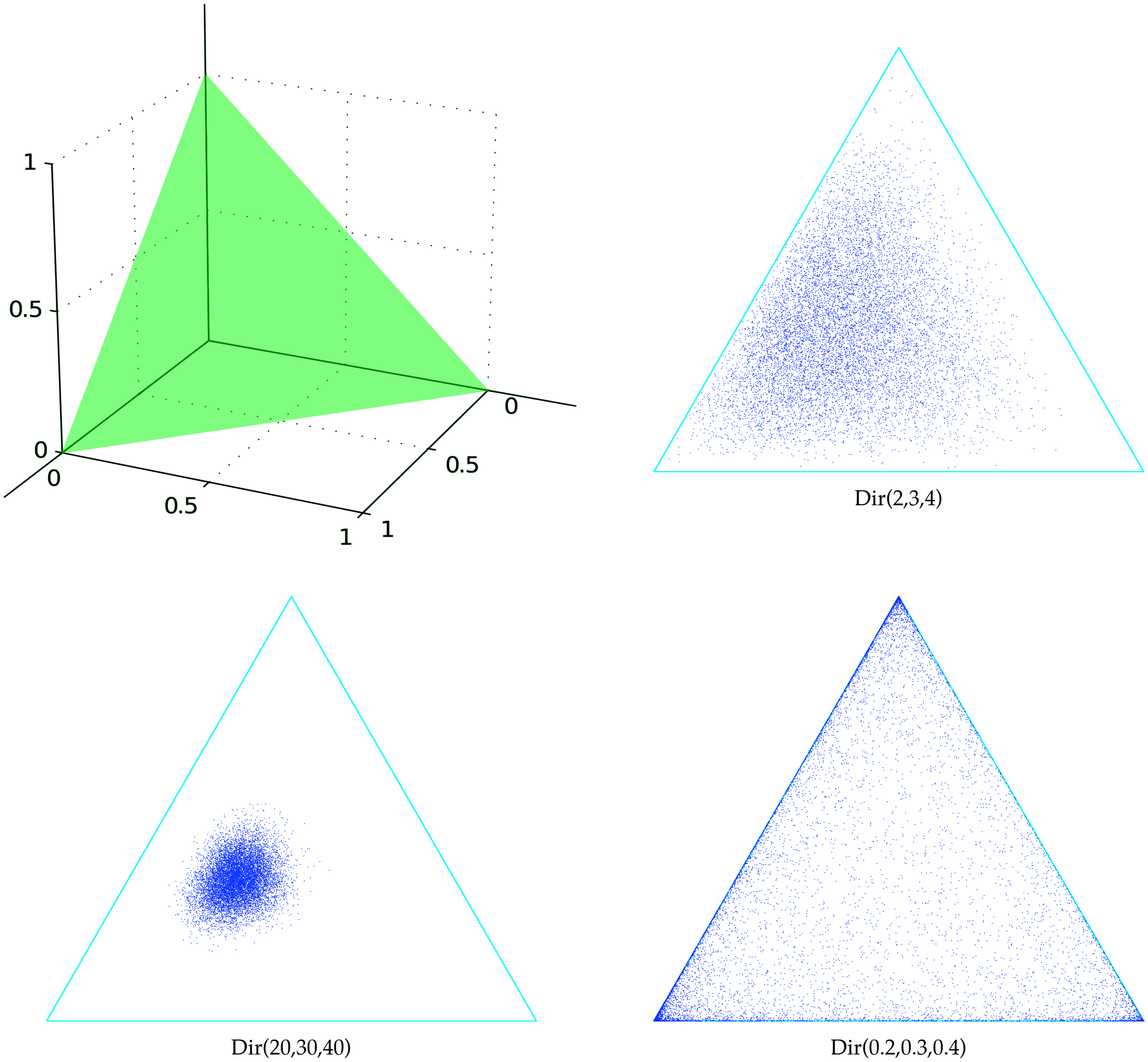
Chapter 2

Mathematical Background

The probabilistic models described in this thesis build on developments in Bayesian nonparametric statistics dating back to the early 1970s. These developments have exploited the properties of the Dirichlet process, a countably infinite-dimensional generalization of the Dirichlet distribution, for a wide range of statistical modeling and machine learning problems. More than many mathematical concepts, the Dirichlet process and inference in derived and related models has inspired a cottage industry in explanatory metaphors; this chapter honors this tradition by presenting our preferred conceptual tools for thinking about these stochastic entities, as well as more formal mathematical foundations. Finally, it presents the Infinite Hidden Markov Model (IHMM), an existing Dirichlet process-based time series model that ultimately inspired the novel techniques presented in this thesis. This introduction to the IHMM also establishes some useful idioms employed in later chapters.

2.1 The Dirichlet distribution

We begin with the ordinary, finite-dimensional Dirichlet distribution, a probability distribution over proportions. As such, the domain of the order $K+1$ Dirichlet probability distribution function (PDF) is the set of $(K+1)$-dimensional vectors $\boldsymbol{v} = [v_1, v_2, \ldots, v_{K+1}]$, where $0 < v_k < 1$ and $\sum_{k=1}^{K+1} v_k = 1$. This set an open analog to the closed set known as the **standard** K**-simplex**, which allows v_k to take the values 0 and 1. In this thesis, terms of the form "K-simplex" without further qualification will refer specifically to the open



Dir(2,3,4)

Dir(20,30,40)

Dir(0.2,0.3,0.4)

FIGURE 2.1: The 2-simplex (**top left**, shaded green), a bounded, convex subset of $\mathbb{R}^3$, is the domain of the order 3 Dirichlet distribution. Collections of samples from three separate Dirichlet distributions, displayed by rotating the 2-simplex into the paper plane, show that large parameters concentrate probability mass around a single mode (**bottom left**), while small parameters push probability mass toward the edges of the simplex (**bottom right**).

set just described. A point in the 2-simplex like $v = [.6, .35, .05]$ might describe your relative preferences for three types of food, or the chances of the three different outcomes of a particular experiment. A Dirichlet distribution can model uncertainty in proportional quantities like these.

The PDF for the Dirichlet distribution on the K-simplex is

$$\mathrm{Dir}(v_1, \ldots, v_{K+1}; \alpha_1, \ldots, \alpha_{K+1}) = \frac{\Gamma(\alpha_1 + \ldots + \alpha_{K+1})}{\Gamma(\alpha_1) \cdot \ldots \cdot \Gamma(\alpha_{K+1})} \, v_1^{\alpha_1 - 1} \cdot \ldots \cdot v_{K+1}^{\alpha_{K+1} - 1}, \tag{2.1}$$

where the α_k parameters are positive real numbers. Because the v_k must sum to 1, some texts replace v_{K+1} with $(1 - v_1 - \ldots - v_K)$. This substitution makes it clear that the Dirichlet distribution is a multivariate generalization of the beta distribution. Figure 2.1 shows how different α_k values modulate the density of the Dirichlet distribution on the 2-simplex.

2.1.1 The Pólya urn interpretation

One convenient interpretation of the Dirichlet distribution involves the Pólya urn, a probabilistic ball-and-urn scheme where balls of $K + 1$ different colors are drawn from an urn. With each draw, the selected ball is returned to the urn with another ball of the same color. Assuming that the urn initially contained α_1 balls of color 1, α_2 balls of color 2, and so on, then as the number of draws approaches infinity, the eventual proportions of the different ball colors will be distributed as $\mathrm{Dir}(\alpha_1, \ldots, \alpha_{K+1})$.

The following proof of this interpretation comes from [88]. Replicating it here introduces a useful concept that we will revisit later. Let X_1, X_2, ..., $X_N = \langle X_N \rangle$ be a sequence of N colors sampled by draws from a Pólya urn whose initial ball counts are α_1, ..., α_{K+1} as described above. For generality we will allow these initial ball counts to be fractional, though new balls placed in the urn after each draw will always be "whole" balls of mass 1.

Theorem 2.1. *(From [88]) Given the above definition of $\langle X_N \rangle$,*

1. *the proportion of balls in the urn allocated to any color k after N draws, abbreviated $\pi_k^{(N)}$, converges with probability 1 to a fixed proportion π_k^* as $N \to \infty$,*
2. *the proportions $\pi^* = [\pi_1^*, \ldots, \pi_{K+1}^*]$ are distributed as $\mathrm{Dir}(\alpha_1, \ldots, \alpha_{K+1})$, and*
3. *given π^*, the individual sequence colors $X_1, X_2, \ldots$ are independent, and each is distributed as π^*.*

Proof. (From [88]) We attack statement 1 first. If we assume that statements 2 and 3 are true, then we can achieve results identical to the Pólya urn scheme by adding new balls to the Pólya urn in a different way: first drawing $\pi^* \sim \mathrm{Dir}(\alpha_1, \ldots, \alpha_{K+1})$, then drawing

$X_n \sim \pi^*$ independently N times, all while adding a new ball of color X_n to the urn each time. By the strong law of large numbers, the proportions of ball colors in the sampled sequence $\langle X_N \rangle$ approaches π^* with probability 1 as $N \to \infty$. As for the contents of the urn, after adding N new balls,

$$\pi_k^{(N)} = \frac{c_k + \alpha_k}{N + \sum_j \alpha_j}, \tag{2.2}$$

where c_k is the number of X_n samples with the color value k. As N grows large, the c_k and N terms dominate the fraction, and thus $\pi_k^{(N)}$ converges on the proportion of color k in $\langle X_N \rangle$, which converges on π_k^*.

We now establish the equivalence of $\langle X_N \rangle$ as drawn from the Pólya urn and from π^*. Let $c_{k,n}$ denote the number of balls of color k that have been *added* to the urn with the $n-1$ previous samples. The probability of $\langle X_N \rangle$ as sampled from the Pólya urn is

$$\begin{aligned}
\pi_{X_1}^{(1)} \cdot \pi_{X_2}^{(2)} \cdot \pi_{X_3}^{(3)} \cdot \pi_{X_4}^{(4)} \cdot \ldots &= \frac{c_{X_1,1} + \alpha_{X_1}}{0 + \sum_j \alpha_j} \cdot \frac{c_{X_2,2} + \alpha_{X_2}}{1 + \sum_j \alpha_j} \cdot \frac{c_{X_3,3} + \alpha_{X_3}}{2 + \sum_j \alpha_j} \cdot \frac{c_{X_4,4} + \alpha_{X_4}}{3 + \sum_j \alpha_j} \cdot \ldots \\
&= \frac{\Gamma(\sum_j \alpha_j)}{\Gamma(N + \sum_j \alpha_j)} \prod_{j=1}^{K+1} \frac{\Gamma(c_j + \alpha_j)}{\Gamma(\alpha_j)} \\
&= \frac{\Gamma(\alpha_1 + \ldots + \alpha_{K+1})}{\Gamma(\alpha_j) \cdot \ldots \cdot \Gamma(\alpha_{K+1})} \cdot \frac{\Gamma(c_1 + \alpha_1) \cdot \ldots \cdot \Gamma(c_{K+1} + \alpha_{K+1})}{\Gamma(c_1 + \alpha_1 + \ldots + c_{K+1} + \alpha_{K+1})}.
\end{aligned}$$

Meanwhile, $P(\langle X_N \rangle)$ according to the π^* sampling scheme can be computed by integrating out π^* in $P(\langle X_N \rangle \mid \pi^*) P(\pi^*)$:

$$\begin{aligned}
P(\langle X_N \rangle) &= \int_{K\triangle} P(\langle X_N \rangle \mid \pi^*) P(\pi^*) \, d\pi^* \\
&= \int_{K\triangle} \pi_1^{*\,c_1} \cdot \ldots \cdot \pi_{K+1}^{*\,c_{K+1}} \cdot \frac{\Gamma(\alpha_1 + \ldots + \alpha_{K+1})}{\Gamma(\alpha_j) \cdot \ldots \cdot \Gamma(\alpha_{K+1})} \, \pi_1^{*\,\alpha_1 - 1} \cdot \ldots \cdot \pi_{K+1}^{*\,\alpha_{K+1} - 1} \, d\pi^* \\
&= \frac{\Gamma(\alpha_1 + \ldots + \alpha_{K+1})}{\Gamma(\alpha_j) \cdot \ldots \cdot \Gamma(\alpha_{K+1})} \int_{K\triangle} \pi_1^{*\,c_1 + \alpha_1 - 1} \cdot \ldots \cdot \pi_{K+1}^{*\,c_{K+1} + \alpha_{K+1} - 1} \, d\pi^* \\
&= \frac{\Gamma(\alpha_1 + \ldots + \alpha_{K+1})}{\Gamma(\alpha_j) \cdot \ldots \cdot \Gamma(\alpha_{K+1})} \cdot \frac{\Gamma(c_1 + \alpha_1) \cdot \ldots \cdot \Gamma(c_{K+1} + \alpha_{K+1})}{\Gamma(c_1 + \alpha_1 + \ldots + c_{K+1} + \alpha_{K+1})},
\end{aligned}$$

where $\int_{K\triangle}$ denotes integration over the K-simplex. The integration may be verified by reflecting on the definition of the Dirichlet distribution (2.1), which would divide the integrand above by the integration result to achieve a distribution function that sums to 1.

This derivation shows that $\langle X_N \rangle$ has the same probability under both sampling schemes. Guarantees about convergence under one scheme therefore transfer to the other; here, this verifies that the Pólya scheme yields $\langle X_N \rangle$ samples that converge on a Dirichlet distributed π^* as $N \to \infty$. □

2.1.2 Properties and thought experiments

The final derivation in the proof reveals one of the Dirichlet distribution's most important properties: its role as the conjugate prior of the multinomial distribution. Consider a multinomial distribution over $K+1$ possible outcomes whose probabilities are represented by a hidden stochastic vector π. After a finite number of draws from π, let c_k indicate how many such trials yielded outcome k. An integration nearly identical to the above derivation shows that if we impose a Dirichlet prior $\text{Dir}(\alpha_1, \ldots, \alpha_{K+1})$ on π, its posterior density after the trials is the Dirichlet distribution $\text{Dir}(\alpha_1 + c_1, \ldots, \alpha_{K+1} + c_{K+1})$.

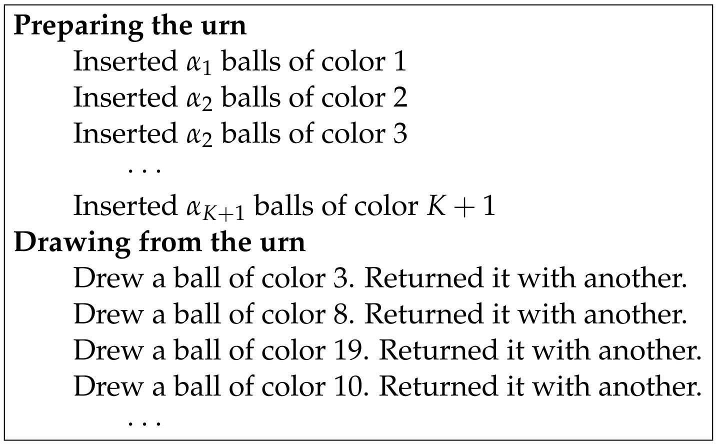
That aside, the Pólya urn view of the Dirichlet distribution, particularly the idea of generating proportional vectors via infinite draws from the urn, makes certain useful properties of the Dirichlet distribution easier to visualize:

¶ **Variance** — Larger α parameters yield Dirichlet distributions that concentrate mass at a small region of the simplex—a region centered at the proportions achieved by rescaling the α parameters to sum to 1. The Pólya urn metaphor permits a simple thought experiment that affirms this inverse relationship between α mass and variance.

Note from Equation 2.2 that as the number of balls $N + \sum_j \alpha_j$ within the urn increases, the effect on the proportions $\pi^{(N)}$ of depositing a new ball diminishes; this is essentially a restatement of the convergence property in Theorem 2.1. As long as the number of balls originally placed in the urn is much larger than the number of new balls, the color proportions will not deviate much from the original arrangement; or, as long as $\alpha_k \gg c_k$, the α terms dominate Equation 2.2. Larger α terms cause this dominance to persist through more draws—long enough that by the time the new ball counts c_k become significant, the effects of adding new balls are minimal, and the proportions $\pi^{(N)}$ within the urn have nearly converged on their limiting proportions π^*.

¶ **Aggregation** — Assume $\pi^* = [\pi_1^*, \ldots, \pi_{K+1}^*] \sim \text{Dir}(\alpha_1, \ldots, \alpha_{K+1})$. Then

$$\pi^{*'} = [\pi_1^*, \ldots, \pi_m^* + \pi_{m+1}^*, \ldots, \pi_{K+1}^*] \sim \text{Dir}(\alpha_1, \ldots, \alpha_m + \alpha_{m+1}, \ldots, \alpha_{K+1}).$$



Preparing the urn

Inserted α_1 balls of color 1
Inserted α_2 balls of color 2
Inserted α_2 balls of color 3
. . .
Inserted α_{K+1} balls of color $K+1$

Drawing from the urn

Drew a ball of color 3. Returned it with another.
Drew a ball of color 8. Returned it with another.
Drew a ball of color 19. Returned it with another.
Drew a ball of color 10. Returned it with another.
. . .

FIGURE 2.2: An excerpt from an imaginary "experimenter's logbook" narrating the progression of a Pólya urn experiment. To achieve a collection of balls in the urn whose color proportions are Dirichlet distributed, the experimenter must continue drawing forever, so this excerpt is relatively short.

More generally, any vector made by replacing any subset of the components of a Dirichlet distributed vector with the sum of those components is also Dirichlet distributed, and the parameter for this replacement component is the sum of the parameters for the subset.

One way to visualize this outcome through the Pólya urn interpretation is to imagine an experimenter keeping a detailed log of their Pólya urn experiment with balls of $K+1$ colors, as in Figure 2.2. If the experimenter later edited the transcript to replace all occurrences of the number 19 (for example) with the number 8, the resulting notes would be indistinguishable from the log of a Pólya urn experiment without balls of color 19, where color 8 was initially allocated $\alpha_8 + \alpha_{19}$ balls, and eventually wound up with $\alpha_8 + \alpha_{19} + c_8 + c_{19}$ balls at the experiment's conclusion. It follows that the proportion of color 8 balls in the "edited" experiment would be the sum of the proportions of color 8 and color 19 balls in the original experiment.

¶ **Subset proportions are Dirichlet** — Assume $\pi^* \sim \text{Dir}(\alpha_1, \ldots, \alpha_{K+1})$. Given any subset of the components of π^* (e.g. $\{\pi^*_s : s \in S\}$ and $S = \{3, 5, 8\}$) the relative proportions of these components (π^*_3/A, π^*_5/A, π^*_8/A, where $A = \sum_{s \in S} \pi^*_s$) are independent of the sum of these components (A) and the relative proportions or sums of any other non-overlapping subset of components of π^*. Moreover, these relative proportions are Dirichlet distributed, with the parameters for this distribution identical to the parameters for the corresponding components in the original Dirichlet distribution (α_3, α_5, α_8).

To visualize this property of Dirichlet distributions, consider again the experimental log in Figure 2.2. Now imagine the experimenter crossing out all lines referring to colors outside of some chosen subset S. The remaining lines would be indistinguishable from a log describing a Pólya urn experiment on just the colors within S—regardless of whether the preserved lines made up 9% or 99.99% of the original text, and regardless of any of the color proportions in the crossed-out lines. This independence is analogous to the independence properties mentioned above.

The two elegant properties of Dirichlet distributions just mentioned are established more rigorously in the following proof, which contains a derivation that we will revisit later:

Theorem 2.2. *Dirichlet distributed random vectors exhibit the aggregation property, and the relative proportions of subsets of components of these vectors are also Dirichlet distributed.*

We begin with the aggregation property. To prove this, we consider without loss of generality the aggregation of the first two dimensions in an Dirichlet distribution on the K-simplex. Stipulating the first two dimensions does not restrict generality because the dimensions of a Dirichlet distribution may be permuted freely, and an arbitrary number of dimensions may be aggregated by applying the two-state aggregation recursively.

Proof. Let $\pi_1^*, \pi_2^*, \ldots, \pi_{K+1}^* \sim \text{Dir}(\alpha_1, \alpha_2, \ldots, \alpha_{K+1})$. Now consider the transformation $\boldsymbol{g} = \Phi(\boldsymbol{\pi}^*)$ where $g_1 = \frac{\pi_1^*}{\pi_1^*+\pi_2^*}$, $g_2 = \pi_1^* + \pi_2^*$, and $g_m = \pi_m^*$ for all $m \geq 3$. The determinant of the Jacobian for the inverse transformation $\Phi^{-1}(\boldsymbol{g})$ is

$$\begin{vmatrix} g_2 & g_1 & 0 & \cdots \\ -g_2 & (1-g_1) & 0 & \cdots \\ 0 & 0 & 1 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{vmatrix} = g_2.$$

Thus

$$\begin{aligned} p(g_1, g_2, \ldots, g_{K+1}) &= \text{Dir}(\Phi^{-1}(\boldsymbol{g})) \cdot g_1 \\ &= \frac{\Gamma(\alpha_1 + \alpha_2 + \ldots + \alpha_{K+1})}{\Gamma(\alpha_1)\Gamma(\alpha_2) \cdot \ldots \cdot \Gamma(\alpha_{K+1})} g_2^{\alpha_1+\alpha_2-1} g_1^{\alpha_1-1} (1-g_1)^{\alpha_2-1} g_3^{\alpha_3-1} \cdot \ldots \cdot g_{K+1}^{\alpha_{K+1}-1}. \end{aligned}$$

Integrating away g_1 leaves g_2, the aggregation of π_1^* and π_2^*, and the rest of the proportions from g_3 on. Since

$$\int_0^1 g_1^{\alpha_1-1}(1-g_1)^{\alpha_2-1}\, dg_1 = \frac{\Gamma(\alpha_1)\Gamma(\alpha_2)}{\Gamma(\alpha_1+\alpha_2)},$$

the gamma functions cancel in the following:

$$\begin{aligned} p(g_2, g_3, \ldots, g_{K+1}) &= \int_0^1 p(g_1, g_2, g_3 \ldots, g_{K+1})\, dg_1 \\ &= \frac{\Gamma(\alpha_1 + \alpha_2 + \ldots + \alpha_{K+1})}{\Gamma(\alpha_1 + \alpha_2)\Gamma(\alpha_3) \cdot \ldots \cdot \Gamma(\alpha_{K+1})} g_2^{\alpha_1+\alpha_2-1} g_3^{\alpha_3-1} \cdot \ldots \cdot g_{K+1}^{\alpha_{K+1}-1} \\ &= \mathrm{Dir}(\alpha_1 + \alpha_2,\ \alpha_3,\ \ldots\ ,\ \alpha_{K+1}). \end{aligned}$$

□

Now for the property that the relative proportions of a subset of the dimensions of a Dirichlet distributed random variable are also Dirichlet distributed, with the Dirichlet parameters identical to the corresponding parameters in the original distribution. To prove this, we consider without loss of generality the subset containing all but the last dimension of a Dirichlet distribution on the K-simplex. As before, permuting the dimensions and applying this proof recursively establishes its validity for all possible dimension subsets.

Proof. For this proof we recall that $\pi^*_{K+1} = 1 - \sum_{k=1}^{K} \pi^*_k$. Thus it is valid to say: let $\pi^*_1, \pi^*_2, \ldots, \pi^*_K \sim \mathrm{Dir}(\alpha_1, \alpha_2, \ldots, \alpha_{K+1})$, since the missing π^*_{K+1} is completely determined by the values of the other K dimensions. Now consider the transformation $\boldsymbol{g} = \Phi(\boldsymbol{\pi}^*)$ where $g_K = \sum_{k=1}^{K} \pi^*_k$ and $g_m = \pi^*_m / g_K$ for all $m < K$. The determinant of the Jacobian for the inverse transformation $\boldsymbol{\phi}^{-1}(\boldsymbol{g})$ is

$$\begin{vmatrix} g_K & 0 & \cdots & 0 & g_1 \\ 0 & g_K & \cdots & 0 & g_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & g_K & g_{K-1} \\ -g_K & -g_K & \ldots & -g_K & (1 - \sum_{m=1}^{K} g_m) \end{vmatrix} = g_K^{K-1}.$$

Thus

$$\begin{aligned} p(g_1, \ldots, g_K) &= \mathrm{Dir}(\Phi^{-1}(\boldsymbol{g})) \cdot g_K^{K-1} \\ &= \frac{\Gamma(\alpha_1 + \alpha_2 + \ldots + \alpha_{K+1})}{\Gamma(\alpha_1)\Gamma(\alpha_2) \cdot \ldots \cdot \Gamma(\alpha_{K+1})} g_K^{-1+\sum_{m=1}^{K} \alpha_m} (1 - g_K)^{\alpha_{K+1}-1} \\ &\qquad \cdot g_1^{\alpha_1-1} g_2^{\alpha_2-1} \cdot \ldots \cdot g_{K-1}^{\alpha_{K-1}-1} \left(1 - \sum_{n=1}^{K-1} g_n\right)^{\alpha_K-1}. \end{aligned}$$

Here, g_K represents the aggregate mass of the first K dimensions of the Dirichlet random variable $\boldsymbol{\pi}$—that is, all dimensions but the very last one. The remaining quantities $g_1, \ldots, g_{K-1}$ determine the ways in which the first K dimensions of $\boldsymbol{\pi}$ divide up the mass in g_K. We will integrate away g_K to attain only these relative proportions. Since

$$\int_0^1 g_K^{-1+\sum_{m=1}^K \alpha_m}(1-g_K)^{\alpha_{K+1}-1} = \frac{\Gamma(\alpha_1+\alpha_2+\ldots+\alpha_K)\Gamma(\alpha_{K+1})}{\Gamma(\alpha_1+\alpha_2+\ldots+\alpha_{K+1})},$$

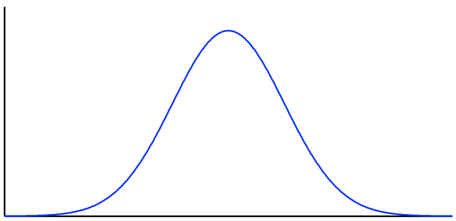
the gamma functions cancel in the following:

$$\begin{aligned} p(g_1, g_2, \ldots, g_{K-1}) &= \int_0^1 p(g_1, g_2, \ldots, g_{K-1}, g_K)\, dg_K \\ &= \frac{\Gamma(\alpha_1+\alpha_2+\ldots+\alpha_K)}{\Gamma(\alpha_1)\Gamma(\alpha_2)\cdot\ldots\cdot\Gamma(\alpha_K)} g_1^{\alpha_1-1} g_2^{\alpha_2-1}\cdot\ldots\cdot g_{K-1}^{\alpha_{K-1}-1}\left(1-\textstyle\sum_{m=1}^{K-1} g_m\right)^{\alpha_K-1} \\ &= \mathrm{Dir}(\alpha_1, \alpha_2, \ldots, \alpha_K). \end{aligned}$$

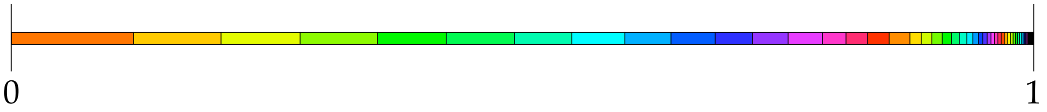
□

2.2 The Dirichlet process

We now confront an infinite generalization of the Dirichlet distribution, albeit one that is infinite in a particular and useful way. The Dirichlet process is a discrete distribution over a countably infinite set of *atoms*, which are drawn independently from some finite *base measure*. To build an intuition for what kind of uncertainty is modeled by the Dirichlet process, which may lead to an understanding of what it is good for, we start with a commonly-presented imaginary mechanism that generates Dirichlet process samples. This mechanism has two core components. The first is a *base measure*, which can be any finite measure (for now, a non-negative function with a finite integral), like this 1-D normal density on the reals.

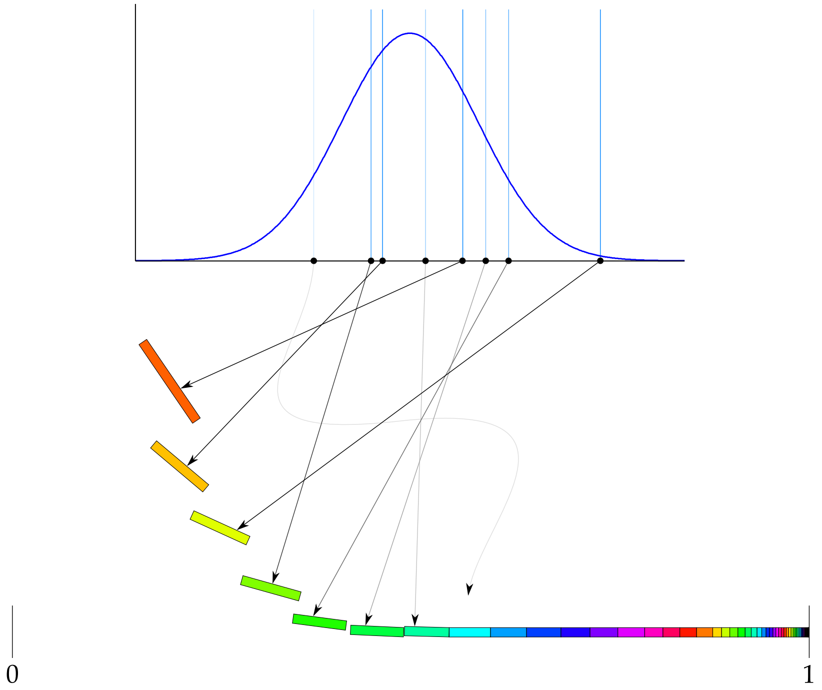


The second component is some mechanism capable of dividing the unit interval $[0, 1]$ into a countably infinite number of pieces. The sizes of these pieces, or in other words the proportions of the interval taken up by the pieces, should be Dirichlet distributed—or something akin to that, since it's impossible to write an expression for an order-∞ Dirichlet density. Critically, this arrangement allows for each proportion to take on a unique size, and thus not all of the proportions have to be infinitesimally small: some can be much larger than that, occupying sizable fractions of the interval. If we sorted such a set of proportions by size and concatenated them, they might look like this.



In this image, the horizontal widths of the rectangles indicate the sizes of the proportions; height and color are added only to make the intervals easier to see. Note how the sorted proportions become smaller as we near the 1; depending on the capabilities of your printer, you may not be able to make out the infinite number of very tiny intervals crowding there.

Once we've generated this infinite collection of proportions, we assign each proportion to a random draw from the base measure—an *atom*. We depict this pairing in progress fancifully below.



This pairing of proportions and atoms is a sample from a Dirichlet process, and by itself it represents a discrete distribution over the atoms, where each proportion indicates the likelihood of drawing its paired atom from the entire set of atoms. Since there are a countably infinite number of proportions, there are as many outcomes of this distribution.

The following make-believe example application of the Dirichlet process occupies the vast sub-genre of foodservice-related metaphors. Consider a diner with no menus: the customers order whatever comes to mind. The universe of possible meals is enormous and, depending on ratios of ingredients and so forth, continuous in at least some dimensions. In contrast, the ordering behavior of the customers appears discrete, chiefly because in some cases more than one customer orders the same thing: a hamburger, scrambled eggs, and so on; some items are well-known and popular. Nevertheless, at any moment a new patron could arrive and demand a unique dish of their own imagining.

A restaurant manager wishing to model the customers' relative preference for meals could find Dirichlet processes useful. Having served only a finite number of customers, the manager can't know her clientele's exact tastes; instead, she must incorporate her observations into a distribution that reflects her uncertainty. Using a Dirichlet process, the atoms are dishes covered by some base measure over all possible meal items (we'll just assume that she has such a thing), and the associated proportions reflect customers' relative preferences for the dishes. Through the properties of the Dirichlet process, this model can account for the infinity of possible dishes, the repeated ordering of popular meal choices, and of course the lack of certainty in the exact arrangement of the proportions. Naturally, the task of inferring the parameters for—or even using—such an infinite model seems daunting at first, but we will soon introduce techniques that make it possible. First, a more rigorous presentation of the Dirichlet process is necessary.

2.2.1 A more formal characterization

We call the tuple $\mathfrak{F} = (Y, \Sigma)$ a *measurable space* if Y is any set (e.g. the natural numbers), and Σ is any collection of subsets of Y that is closed under complementation and countable unions (e.g. the power set of the natural numbers). The tuple $\mathfrak{F}$ is sometimes called a σ-field, while Σ is known as a σ-algebra. A *measure* μ is a function from the members of σ to the non-negative real numbers such that the measure of the empty set $\mu(\varnothing) = 0$, and the measure of the union of a countable set of pairwise disjoint members of Σ is the sum of the measures of those members; thus, in our natural number example above, $\mu(\{2, 3, 7, 9\}) =$

$\mu(\{2,3\}) + \mu(\{7,9\})$. Finally, a measure on $\mathfrak{F}$ is finite if $\mu(Y)$ is finite, and a *probability measure* if $\mu(Y) = 1$.

These concepts allow us to characterize the Dirichlet process as follows: given a measurable space $\mathfrak{F} = (Y, \Sigma)$ and some finite *base measure* μ on $\mathfrak{F}$, then for any partition of Y into pairwise disjoint subsets Z_1, …, Z_{K+1}, where $Z_k \in \Sigma$ and $\bigcup_{k=1}^{K+1} Z_k = Y$, the probabilities $\pi_{Z_1}, \ldots, \pi_{Z_{K+1}}$ assigned to these subsets are distributed as $\mathrm{Dir}(\mu(Z_1), \ldots, \mu(Z_{K+1}))$. In simpler terms, if you break a measurable space into any finite arrangement of pieces, the probabilities assigned to each piece are distributed according to an ordinary, finite Dirichlet distribution, with the parameters computed by evaluating μ on each piece.

Glimmering faintly in the haze of these abstractions is the aggregation property of ordinary Dirichlet distributions. Suppose we are able to consider all the singleton members of some discrete Y—call them z_1, z_2, and so on—and suppose that these are in Σ. By the characterization above, the probabilities assigned to these singletons are distributed as $[\pi_{z_1}, \pi_{z_2}, \ldots] \sim \mathrm{Dir}(\mu(z_1), \mu(z_2), \ldots)$. If we now call $Z_1 = z_1 \cup z_2$, we would expect from the aggregation property that $[\pi_{Z_1}, \pi_{z_3}, \ldots] \sim \mathrm{Dir}(\mu(z_1) + \mu(z_2), \mu(z_3), \ldots)$, which is consistent with the properties of the Dirichlet processes and finite measures. This kind of aggregation—and its integral analog for continuous $\mathfrak{F}$—allows us to grapple with inference problems involving Dirichlet processes by mapping them onto problems involving finite Dirichlet distributions on partitions.

2.2.2 Return of the Pólya urn

Just as we can use the Pólya urn to generate Dirichlet distributed vectors, Blackwell and Ferguson show that we can use a generalization of the Pólya urn to realize Dirichlet processes [88]. This generalization involves extending the urn to arbitrary measurable spaces with finite (i.e. normalizable) base measures, and it will eventually serve as the foundation for our Dirichlet process inference techniques. To begin, let μ be a finite base measure; we characterize our urn scheme as follows. For the first draw, we sample directly from a probability measure made by normalizing the base measure to sum to 1:

$$X_1 \sim \frac{\mu}{\mu(Y)}. \tag{2.3}$$

For the next sample we draw from a modified measure with an additional unit mass at the last sample—the additional mass is analogous to an additional ball of color X_1 placed into the urn:

$$X_2 \sim \frac{\delta(X_1) + \mu}{1 + \mu(Y)}. \tag{2.4}$$

And so on. For completeness, after N draws, we sample from this probability measure:

$$X_{N+1} \sim \frac{\mu + \sum_{i=1}^{N} \delta(X_i)}{N + \mu(Y)}, \tag{2.5}$$

which we will call $\mu^{(N)}$. If we consider the circumstance where Y is the set of real numbers and μ is a continuous density function over Y, like the normal distribution used in our example, we can see that $\mu^{(N)}$ nevertheless behaves a lot like a discrete probability measure, in that for any single real number x, $\mu^{(N)}(\{x\})$—that is, $\mu^{(N)}$ evaluated on the set containing only x—is *not* infinitesimally small if (and only if) x equals one of the sampled X_k. In contrast, $\mu(\{x\})$ is infinitesimal for every real number x. Using a generalization of Theorem 2.1, the proof in [88] shows that as $N \to \infty$, $\mu^{(N)}$ approaches μ^*, a discrete probability measure over Y with countably infinite outcomes—a sample from a Dirichlet process over Y with base measure μ.

That said, in the grand literary tradition of foreshadowing, let's consider the dynamics of our generalized Pólya urn still further. Taking inspiration from aggregation, let us designate by x_k, $k \in \{1, \ldots, K\}$ the K unique elements of Y drawn from the urn after N samples. Also, let $Y' = Y - x_1 - \ldots - x_K$. Together, this collection forms a partition of Y. Based on Equation 2.5, we can express the probability of X_{N+1} being one of the x_k as

$$P(X_{N+1} = x_k) = \frac{\mu(x_k) + c_k}{N + \mu(Y)}, \tag{2.6}$$

or, the probability of drawing anything novel this time around as

$$P(X_{N+1} \in Y') = \frac{\mu(Y')}{N + \mu(Y)}. \tag{2.7}$$

Here, as with the original Pólya urn explication, c_k is the number of draws in X_1, ..., X_N that selected element x_k. Now, if μ is a continuous measure, then $\mu(x_k)$ is infinitesimal, and effectively $\mu(Y') = \mu(Y)$. We simplify for this case accordingly:

$$P(X_{N+1} = x_k) = \frac{c_k}{N + \mu(Y)}, \qquad P(X_{N+1} \in Y') = \frac{\mu(Y)}{N + \mu(Y)}. \tag{2.8}$$

Note that each draw has a chance of increasing the number of unique elements by selecting one of the members of Y'. This chance is influenced by the magnitude of $\mu(Y')$, and it diminishes over time as N increases. Nevertheless, as $N \to \infty$ and $\mu^{(N)}$ converges on μ^*, $K \to \infty$ as well.

2.2.3 The stick-breaking interpretation

The beginning of this subsection presented an imaginary Dirichlet process sampling mechanism with two key components: a base measure and a means of creating a countably infinite set of proportions. It may be slightly mystifying, then, that both our formal definition of the Dirichlet process and our Pólya urn characterization seem to refer only to the base measure. In fact the two-component approach to understanding the Dirichlet process comes from factoring the base measure into its two most salient properties—shape and size—and considering their effects separately, an exercise that will help illuminate aspects of Dirichlet process inference further on. The seminal way to realize this separation is the *stick-breaking* interpretation, introduced by Sethuraman in [89]. We shall now derive this interpretation from the generalized Pólya urn described previously.

Consider a base measure μ, and for simplicity let us assume that μ is a continuous density. We introduce the new shorthand $\alpha = \mu(Y)$ for the integral of μ over its domain—the "size" referred to earlier—but we can borrow established notation for its shape, the normalized base measure $\mu^{(0)} = \mu/\alpha$. We "initialize" the urn with μ, which is to say that the first draw will be from $\mu^{(0)}$ just as described in Equation 2.3. For the sake of easy analogy, let us also refer to X_1, the result of the first draw, as a *color*, specifically as the color C_1, which denotes the first *unique* color we drew.[1] After the draw, an extra unit mass associated with C_1 is returned to the urn, in keeping with the numerator of Equation 2.4.

Now let us divide the current contents of the urn into two portions: the mass associated with the color C_1, and the mass associated with all other possible colors. Some contemplation will reveal that since μ is continuous, the first portion has mass 1, while the second has mass α. By the formal definition of the Dirichlet process, we know that after infinite draws from the urn, the proportions of the mass allocated to color C_1 and to all other colors combined respectively will be distributed as $\text{Dir}(1, \alpha) = \text{Beta}(1, \alpha)$.

Before making infinite draws from the urn, though, let's just draw a few more times and stop when we get a second unique color C_2. We may have drawn C_1 a few times again, so the masses associated with C_1, C_2, and all other colors are some count c_1, 1, and α respectively. Using the same logic as before, we know that after infinite draws of the

[1] *Atom* is an equivalent term to *color* here, but *color* emphasizes the Pólya urn metaphor.

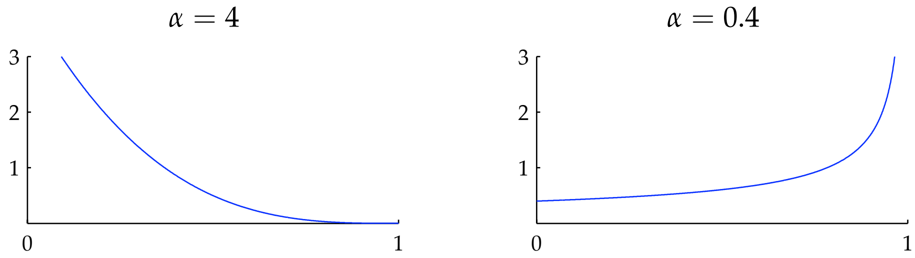
urn, the proportions of mass allocated to these colors will now be distributed according to the order-3 Dirichlet distribution $\mathrm{Dir}(c_1, 1, \alpha)$. This time, though, we invoke the subset-proportions-are-Dirichlet property to focus strictly on C_2 and the leftover colors. How are their proportions distributed within whatever mass in the urn is not allocated to C_1? Via direct application of this property, the answer again is $\mathrm{Beta}(1, \alpha)$.

This procedure can be applied "recursively" to attain a new proportional sample for a succession of colors C_1, C_2, C_3, ... sampled from the base measure. Starting with a unit interval-sized "stick", break off a fraction of the stick drawn from $\mathrm{Beta}(1, \alpha)$ and assign it to C_1. Next, break off *from the remaining stick* a fraction drawn from $\mathrm{Beta}(1, \alpha)$ and assign it to C_2. Continue breaking off pieces of the stick in this way for subsequent colors. Because it's impossible to draw 1 from a beta distribution, we are assured of never using up the stick; however, since we continue to take fractions of smaller and smaller pieces, the proportions allocated to successive colors tend to decrease in size.

The stick breaking interpretation is our first practical realization of the Dirichlet process: rather than requiring infinite draws from an urn, or an order-∞ Dirichlet distribution, this arrangement needs only samples from a beta distribution, which can be generated until the investigator tires of it. Significantly, since the first proportions broken from the stick are usually the largest, samples from the base measure neglected by stopping the stick breaking at some point are likely to be associated with negligible probabilities.

That said, there are also attractive conceptual properties of this construction, which are chief motivation for exploring it in depth here. First, the assignment of a countably infinite collection of proportions broken off of the stick to atoms drawn from the base measure makes it clear that a sample from a Dirichlet process is a discrete measure. More striking, perhaps, is the segregation of the roles of the base measure's size and shape. Note that to sample $C_1, C_2, C_3, \ldots,$ only the shape of the base measure is relevant. Meanwhile, the size of the measure, α, only turns up in the creation of the proportions.

Before moving on, some further contemplation of this last fact is worthwhile. Consider the following plots of the density of $\mathrm{Beta}(1, \alpha)$ for large and small α.



The larger α value yields smaller samples from the beta distribution, and hence smaller stick breaks, than the smaller α value does: conceptually, it will take longer with the large α value for the pieces to dwindle into insignificance than with the smaller α. Moreover, the diversity of the sticks will be greater with the larger α: there will be a few sticks at any given scale and a horde of sticks at smaller scales. In contrast, as α increases, the sticks tend more to resemble short, uniform-length pieces. A Dirichlet process employing these short sticks will seem more like a discrete reproduction of the base measure from a collection of uniformly-weighted atoms. Recall that with the original Dirichlet distribution, the variance was inversely proportional to the magnitude of the parameters; by detailing the effect of differently sized α values, we have encountered the same property for Dirichlet processes.

2.2.4 Dirichlet process mixture models

Our prolonged explication of Dirichlet processes finally arrives at a practical application in the Dirichlet process mixture model (DPM), which sees use in a variety of data clustering applications. One attraction of DPMs for clustering is that by exploiting the mechanics of Dirichlet processes, it is possible to group data together without a rigid specification of how many groups there are beforehand.

As with some other clustering approaches (e.g. K-means), DPMs group data by inferring the parameters of a generative model from the data, then analyzing the inferred model to assign cluster labels to each data point. Generative accounts for DPMs usually take a form like the following:

Variables

Name	*Description*	*Hidden?*	*Example*
X_n	Data points n, $n \in \{1, \ldots, N\}$	No	A real number
θ_n	Parameters for the mixture component distributions $f(\theta_n)$ from which the X_n were sampled	Yes	The mean of a normal distribution with $\sigma = 1$
μ	Base measure on the space of θ_n parameters	No	A normal distribution with zero mean and $\sigma = 10$, multiplied by some scaling factor α
μ^*	A Dirichlet process distributed probability measure over the space of θ_n parameters	Yes	

Generative process

1. Draw μ^* from a Dirichlet process with base measure μ.

2. Draw N mixture component parameters θ_n from μ^*.

3. Draw each of the N data points as $X_n \sim f(\theta_n)$.

This generative procedure may be counterintuitive at first, since each data point X_n is assigned its own mixture component $f(\theta_n)$. Believe it or not, this is the way things are, and indeed "having about as many parameters as data points" is one of the vaguely defining characteristics of nonparametric statistical techniques. What makes DPMs useful for clustering is the fact that μ^* is a discrete distribution, and thus for well-tuned base measures, the θ_n parameters drawn by some sets of data points—those belonging to a cluster—*will tend to be the same*. Conceptually, the process of partitioning the data points into groups thus involves collecting points whose inferred θ_n parameters are identical.

Figure 2.3 shows a graphical model depiction of the Dirichlet process mixture model.

2.2.5 The Chinese restaurant process

An important fact is implicit in the preceding description of Dirichlet process mixture models applied to clustering: after inference is complete, the measure μ^* is not actually used to assemble the data points into clusters, only the θ_n parameters. The Chinese restaurant

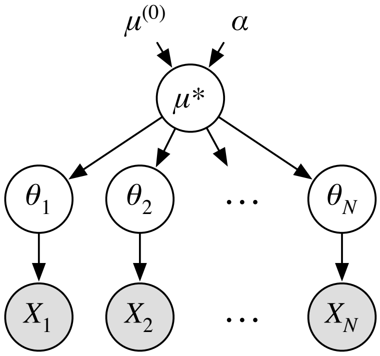


FIGURE 2.3: A graphical depiction of the Dirichlet process mixture model. Here, the base measure μ has been factored as $\mu = \alpha\mu^{(0)}$, where $\mu^{(0)}$ is a probability measure and α is a positive constant.

process, one of the most popular and straightforward approaches to Dirichlet process inference, takes advantage of this fact by marginalizing μ^* out of the model altogether. The following discussion assumes that $\mu^{(0)}$ is a continuous base density. Consider that with μ^* known, the probability of θ' being the mixture parameter used to generate X_n is conditionally independent of the other data and mixture component parameters:

$$\begin{aligned} P(\theta_n = \theta' \mid \boldsymbol{X}, \boldsymbol{\theta}_{\setminus n}, \mu^*, \mu^{(0)}, \alpha) &= P(\theta_n = \theta' \mid X_n, \mu^*) && (2.9) \\ &\propto P(\theta' \mid \mu^*)\, p(X_n \mid \theta_n), && (2.10) \end{aligned}$$

where $\boldsymbol{X}$ denotes the entire dataset and $\boldsymbol{\theta}_{\setminus n}$ denotes all mixture component parameters except for θ_n. This arrangement is typical to many mixture model techniques. After integrating away μ^*, however, some of this independence is lost:

$$\begin{aligned} P(\theta_n = \theta' \mid \boldsymbol{X}, \boldsymbol{\theta}_{\setminus n}, \mu^{(0)}, \alpha) &= P(\theta_n = \theta' \mid X_n, \boldsymbol{\theta}_{\setminus n}, \mu^{(0)}, \alpha) \\ &\propto P(\theta' \mid \boldsymbol{\theta}_{\setminus n}, \mu^{(0)}, \alpha)\, p(X_n \mid \theta_n). \end{aligned}$$

To understand the nature of this $P(\theta' \mid \boldsymbol{\theta}_{\setminus n}, \mu^{(0)}, \alpha)$ term and the integration that defines it, one relatively simple means is to invoke the Dirichlet aggregation property to create μ'. This is an ordinary, finite set of Dirichlet distributed proportions such that if $\phi_1, \ldots, \phi_K$ correspond to all K *unique* mixture parameters in the collection $\boldsymbol{\theta}_{\setminus n}$ (recall that some elements of $\boldsymbol{\theta}_{\setminus n}$ may be repeats, giving rise to clustering), then $\mu'_{\phi_k} = P(\phi_k \mid \mu^*)$, and one further element:

$$\mu'_{\text{new}} = \int_{Y-\phi_1-\ldots-\phi_K} P(\phi \mid \mu^*)\, d\phi = \mu^*(Y - \phi_1 - \ldots - \phi_K),$$

where Y is the space of possible mixture parameters. This integral is the invocation of the Dirichlet aggregation property and is summing over all of the elements of μ^* that correspond to mixture parameters that haven't been assigned to any of our data points.

With this, if θ' is identical to any of the mixture parameters in $\boldsymbol{\theta}_{\setminus n}$, we can compute the joint probability of all the mixture parameters, $P(\theta', \boldsymbol{\theta}_{\setminus n} \mid \mu')$, exactly as $\prod_{n=1}^{N} \mu'_{\theta_n}$. If we designate by c_{ϕ_k} the number of mixture parameters in $\theta_{\setminus n}$ that equal the unique mixture parameters ϕ_k, we can also compute this probability as $\mu_{\theta'} \prod_{k=1}^{K} {\mu'_{\phi_k}}^{c_{\phi_k}}$. Integrating away μ' in this second case as we condition on the $n-1$ other cluster parameter selections, we have the following:

$$
\begin{aligned}
P(\theta' \mid \boldsymbol{\theta}_{\setminus n}, \mu^{(0)}, \alpha) &= \frac{P(\theta', \boldsymbol{\theta}_{\setminus n} \mid \boldsymbol{X}, \mu^{(0)}, \alpha)}{P(\boldsymbol{\theta}_{\setminus n} \mid \boldsymbol{X}, \mu^{(0)}, \alpha)} \\
&= \frac{\int_{K\triangle} P(\mu' \mid \mu^{(0)}, \alpha)\, P(\theta', \boldsymbol{\theta}_{\setminus n} \mid \mu')\ d\mu'}{\int_{K\triangle} P(\mu' \mid \mu^{(0)}, \alpha) \quad P(\boldsymbol{\theta}_{\setminus n} \mid \mu')\ d\mu'} \\
&= \frac{\int_{K\triangle} \mathrm{Dir}\left(\mu'_{\phi_1}, \ldots, \mu'_{\phi_K}, \mu'_{\text{new}}\,;\, a_{\phi_1}, \ldots, a_{\phi_K}, a_{\text{new}}\right) \cdot \prod_{k=1}^{K} {\mu'_{\phi_k}}^{c_{\phi_k} + \delta(\theta' = \phi_k)}\ d\mu'}{\int_{K\triangle} \mathrm{Dir}\left(\mu'_{\phi_1}, \ldots, \mu'_{\phi_K}, \mu'_{\text{new}}\,;\, a_{\phi_1}, \ldots, a_{\phi_K}, a_{\text{new}}\right) \cdot \prod_{k=1}^{K} {\mu'_{\phi_k}}^{c_{\phi_k}} \qquad d\mu'},
\end{aligned}
$$

where we have used the following shorthand:

$$
a_{\phi_k} = \alpha \mu^{(0)}(\phi_k) \qquad \text{and} \qquad a_{\text{new}} = \alpha \mu^{(0)}(Y),
$$

and where numerator and denominator are identical except for the one added delta function in the exponent in the numerator, representing the extra multinomial draw of ϕ_k for θ_n. This simplifies to something more compact:

$$
= \frac{\frac{1}{Z} \int_{K\triangle} {\mu'_{\text{new}}}^{a_{\text{new}} - 1} \prod_{k=1}^{K} {\mu'_{\phi_k}}^{a_{\phi_k} + c_{\phi_k} + \delta(\theta' = \phi_k) - 1}\ d\mu'}{\frac{1}{Z} \int_{K\triangle} {\mu'_{\text{new}}}^{a_{\text{new}} - 1} \prod_{k=1}^{K} {\mu'_{\phi_k}}^{a_{\phi_k} + c_{\phi_k} - 1} \qquad d\mu'},
$$

where Z is the normalization constant from the $P(\mu' \mid \mu^{(0)}, \alpha)$ Dirichlet distribution. The integrands, being sums of proportions raised to powers greater than -1, are themselves unnormalized Dirichlet densities, and as such we can compute the integrals "for free" as

Dirichlet normalizers, as we did in Theorem 2.1. Our probability is equal to:

$$= \frac{\dfrac{\Gamma(a_{\text{new}})\prod_{k=1}^{K}\Gamma\left(a_{\phi_k}+c_{\phi_k}+\delta(\theta'=\phi_k)\right)}{\Gamma\left(a_{\text{new}}+\sum_{k=1}^{K}a_{\phi_k}+c_{\phi_k}+\delta(\theta'=\phi_k)\right)}}{\dfrac{\Gamma(a_{\text{new}})\prod_{k=1}^{K}\Gamma\left(a_{\phi_k}+c_{\phi_k}\right)}{\Gamma\left(a_{\text{new}}+\sum_{k=1}^{K}a_{\phi_k}+c_{\phi_k}\right)}}.$$

A flurry of cancellation happens next after invoking the property $\Gamma(x+1) = x\,\Gamma(x)$, yielding at last

$$= \frac{a_{\theta'}+c_{\theta'}}{a_{\text{new}}+\sum_{k=1}^{K}a_{\phi_k}+c_{\phi_k}},$$

where $a_{\theta'}$ and $c_{\theta'}$ denote the a_{ϕ_k} and c_{ϕ_k} such that $\theta' = \phi_k$. This rather simple expression, mainly a ratio of counts, was seen before in Equation 2.6, a part of our demonstration of how to create a Dirichlet process sample with a generalized Pólya urn, and it differs only in notation. A very similar derivation yields the probability of θ' taking on a value that isn't any of the other cluster parameters, namely

$$P(\theta' \in \Upsilon' \mid \boldsymbol{\theta}_{\backslash n}, \mu^{(0)}, \alpha) = \frac{a_{\text{new}}}{a_{\text{new}}+\sum_{k=1}^{K}a_{\phi_k}+c_{\phi_k}},$$

or, a reproduction of Equation 2.7. Here, as there, we designate Υ' to be $\Upsilon - \theta_1 - \ldots - \theta_{n-1}$, and since we are again stipulating a continuous base density $\mu^{(0)}$, $\mu^{(0)}(\Upsilon') = \mu^{(0)}(\Upsilon) = a_{\text{new}}$. Indeed, for the same reason, $\mu^{(0)}(\theta)$ is infinitesimal for any specific θ, and so we can remove the corresponding parameters in the previous two expressions to get analogs to the final generalized Pólya urn expressions in (2.8):

$$P(\theta' \mid \boldsymbol{\theta}_{\backslash n}, \mu^{(0)}, \alpha) = \begin{cases} \dfrac{c_{\theta'}}{a_{\text{new}}+N-1} & \theta' \in \{\theta_1, \ldots, \theta_{n-1}\}, \\ \dfrac{a_{\text{new}}}{a_{\text{new}}+N-1} & \text{otherwise.} \end{cases}$$

Incidentally, it is not difficult to re-express this conditional probability—or the intuition presented in this subsection—for non-continuous base measures. For these, individual atoms have non-infinitesimal mass associated with them, a situation not unlike the continuous base measure case *after* a number of draws have already been made from the Pólya urn. The precise expression for this conditional probability is left as an exercise, although some of the discussion in §2.4 may offer useful hints.

The conditional probability for a mixture parameter presented above has an important property. Consider the probability of a set of four mixture parameters $\theta_1 = \phi_1$, $\theta_2 =$

ϕ_1, $\theta_3 = \phi_2$, $\theta_4 = \phi_1$. Ignoring the likelihoods of data generated from these parameters, we can compute their combined probabilities as

$$P(\theta_1 \mid \mu^{(0)}, \alpha) \;\cdot\; P(\theta_2 \mid \theta_1, \mu^{(0)}, \alpha) \;\cdot\; P(\theta_3 \mid \theta_1, \theta_2, \mu^{(0)}, \alpha) \;\cdot\; P(\theta_4 \mid \theta_1, \theta_2, \theta_3, \mu^{(0)}, \alpha),$$

expanding to

$$\propto \frac{a_{\text{new}}}{a_{\text{new}}}\mu^{(0)}(\phi_1) \;\cdot\; \frac{1}{a_{\text{new}}+1} \;\cdot\; \frac{a_{\text{new}}}{a_{\text{new}}+2}\mu^{(0)}(\phi_2) \;\cdot\; \frac{2}{a_{\text{new}}+3},$$

where the $\mu^{(0)}(\phi)$ terms reflect the specific, albeit infinitesimal, probabilities of drawing ϕ_1 and ϕ_2 from the base measure. The way the conditioned variables grow as the first expression is read from left to right might suggest that the order in which we consider the mixture parameters affects the outcome of the probability of the expression. As it turns out,

$$P(\theta_4 \mid \mu^{(0)}, \alpha) \;\cdot\; P(\theta_2 \mid \theta_4, \mu^{(0)}, \alpha) \;\cdot\; P(\theta_1 \mid \theta_4, \theta_2, \mu^{(0)}, \alpha) \;\cdot\; P(\theta_3 \mid \theta_4, \theta_2, \theta_1, \mu^{(0)}, \alpha)$$

expands to

$$\propto \frac{a_{\text{new}}}{a_{\text{new}}}\mu^{(0)}(\phi_1) \;\cdot\; \frac{1}{a_{\text{new}}+1} \;\cdot\; \frac{2}{a_{\text{new}}+2}, \;\cdot\; \frac{a_{\text{new}}}{a_{\text{new}}+3}\mu^{(0)}(\phi_2),$$

which is the same as the expanded expression as the other ordering, except with the terms reshuffled. Some meditation on the components of expressions like these may reveal that for any collection of N mixture components, the ordering of the components themselves *never* matters when calculating their conditional probability. The denominators of the multiplied fractions will always count up from a_{new} to $a_{\text{new}} + N - 1$; the first draws of atoms ϕ_k from the base measure will always produce terms $a_{\text{new}} \cdot \mu^{(0)}(\phi_k)$, and subsequent draws of those values will always count up to one minus the number of such draws somewhere in the numerators. Sequences of draws from Pólya processes are *exchangeable*, meaning that the probability of all the draws does not depend on the ordering of the elements in the sequence, only the counts of the different element types it contains.

Exchangeability is especially useful for fitting Dirichlet process mixtures to data, because it allows an inference algorithm using Gibbs sampling to reconsider which mixture parameters generated any particular data point at any time in the inference. The algorithm will typically have computed the unscaled conditional density of a set of mixture parameters $\boldsymbol{\theta}$ as

$$p(\boldsymbol{\theta} \mid \boldsymbol{X}, \mu^{(0)}, \alpha) \propto \prod_{n=1}^{N} P(\theta_n \mid \boldsymbol{\theta}_{\{1,\ldots,n-1\}}, \mu^{(0)}, \alpha)\; p(X_n \mid \theta_n).$$

This computation iterates through the parameters in some predetermined order. Nevertheless, since the ordering does not affect the final value of this expression, the algorithm can determine how changing one of the θ_n values affects the conditional density *by pretending that this θ_n was the last one in the ordering*. Thus, the probability after changing θ_n from θ' to θ'' is

$$p(\boldsymbol{\theta} \,|\, \boldsymbol{X}, \mu^{(0)}, \alpha) \frac{P(\theta_n = \theta'' \,|\, \boldsymbol{\theta}_{\setminus n}, \mu^{(0)}, \alpha) p(X_n \,|\, \theta'')}{P(\theta_n = \theta' \;\,|\, \boldsymbol{\theta}_{\setminus n}, \mu^{(0)}, \alpha) p(X_n \,|\, \theta'\;)},$$

no matter where θ_n is in the original order. The prior terms in the fraction are simple to compute, involving only the ratios of counts derived above. The algorithm can iterate this resampling of the mixture parameters each by each until it converges on a reasonable clustering of the entire data.

This specific approach of using Pólya urns to perform Gibbs sampling-based inference on Dirichlet process mixtures is known as the Chinese Restaurant Process, and it's worth explaining the visualization that gave rise to this term, since its set of characters and props are now an informal shorthand in the machine learning community for describing aspects of Bayesian nonparametric inference with Dirichlet processes and related models.

We imagine a restaurant with what is sometimes called "family style" dining: customers share tables even when they are not in the same party, and diners at the same table all share whatever food happens to be there. Different tables have different dishes, so when a new guest arrives, he usually sits at a table that matches his food preference. Occasionally, no meal at any table is to his taste, and the customer seats himself at a new table to request a more palatable dinner there.

We imagine further that each restaurant patron is gripped by a cruel and vacuous social neurosis. One's taste in food is everything here, a sign of education, class, and gentility. To venture out on one's own and order a novel dish is a bullheaded, precarious kind of maneuver, a peremptory prostration before the silent and ruthless court of social favor. Most patrons opt instead to accommodate their tastes as best they can at tables where others are seated already, and even then more popular tables can seem a safer bet than sparser ones. In any case, each guest must weigh the gustatory and constitutional benefits of captaining their own nourishment against the risk of an enduring social opprobrium.

Thank heavens, then, for the free drinks. For each guest, *gratis*, the restaurant bar offers some quantity of a liquor named *Alpha*. One assumes it must be Greek. Enterprise and bravery transfused from Mediterranean lands in anise and mastic spirits embolden the spineless guests to strike out for their own, and in this, the more the better. A sip of *Alpha* huddles all guests at one table; three fingers engenders an egoist revolution.

It is no matter either way. Whatever Aegean brio stirs in the diners' hearts meets a perfunctory Ligurian quashing at the hands of the restaurant bouncer, a man as massive and taciturn as a boulder. He is from Monte Carlo, and his employment comprises the brusque eviction of anyone putting utensil to food. One at a time he throws them out, and as soon as they march in again to private agony over table choice, another sportscoated missile hurtles past in the opposite direction. This process never ends. The hospitality is impeccable, indefatigably. The lights are always on. Table occupancies wax and wane like the tides of empires. The dishes are delicious and untouched, and the diners quest for them forever. Johann P.G.L. Dirichlet was a 19th-century German mathematician and not a French Existentialist, but one wonders about it sometimes.

As a concluding note, the restaurant is a Chinese restaurant. This ethnic distinction bears no particular relevance to our parable.

The mapping from the preceding tale to the model follows.

Story element	*Model component*
Individual diners	x_n, θ_n pairs
Individual table dishes	ϕ_k
Table popularity	c_{ϕ_k}
The kitchen	$\mu^{(0)}$
Alpha	α hyperparameter
The bouncer	Gibbs sampler

Besides presenting a philosophical object lesson, the Chinese Restaurant Process described mathematically on previous pages offers a clearer intuition for the claim that generalized Pólya urns can generate Dirichlet process samples—at least conceptually. As shown above, marginalizing away the Dirichlet process sample μ^* yields a generalized Pólya urn scheme—therefore we might suspect (and the proof cited earlier shows) that any given generalized Pólya urn scheme must be attainable by marginalizing away some Dirichlet process sample in the presence of some assumed arrangement of counts. Pólya urns converge toward a set of color proportions as the number of balls inside increases, and so we expect that after an infinite number of draws, the convergence selects a single, final Dirichlet process sample.

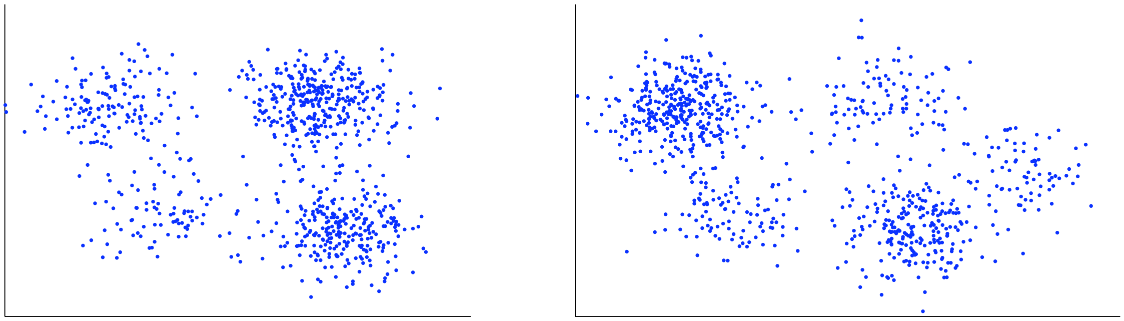


FIGURE 2.4: Example dataset illustrating a situation where a hierarchical Dirichlet process could be a useful model. The dataset contains two sets of 2-D data points, one for each plot above. In both, the four leftmost mixture components creating the data are identical, but the mixing proportions are different. Additionally, the second dataset has a fifth mixture component, which appears on the right side of the plot.

2.3 Hierarchical Dirichlet processes

An example dataset like the one shown in Figure 2.4 can motivate Hierarchical Dirichlet processes (HDPs), a class of nonparametric Bayesian models that can share the same atoms among multiple sets of proportions. Suppose that we have a data collection with two separate sets of points, as shown in the figure. The data in both sets appear to have been drawn from some mixture of distributions, probably Gaussians. Additionally, some of the mixture component distributions in both datasets seem to be the same. Their mixing proportions are different (indeed, one has an entire “extra” component), so it’s incorrect to say that the sets were drawn from the same mixtures, and fairly tenuous to try and combine the data into a single dataset for any serious analysis. Nevertheless, it *is* valid to wonder which mixture component created each data point, and to expect that some data points from both sets will have the same pedigree.

A hierarchical Dirichlet process mixture model can create data like what’s shown in the example. By inferring an HDP mixture model from a dataset, we can perform clustering across separate collections of data (e.g. the two sets of points in the figure) in a principled way. The power of the model to share components across the collections explain its utility in modeling text documents, for example, where each data collection is a document, each element of data is a word, and the mixture components are distributions of words corresponding to different topics [90]. Not counting stop words like “the” or “of” and simplifying things a bit, we might imagine that eighty percent of the words in a particular article in *The Lancet* are associated with a topic we could call *medicine*, with another twenty percent associated with *law*. An article on health issues in *The Journal of Public Policy* might reverse those proportions. By inferring an HDP mixture model in this setting, where the input is only a collection of documents, we could learn:

- Which words are associated with different topics (mixture components).
- How much is each topic represented in each document (per-collection mixture proportions).
- Which topic generated each word in the document (assignments of data points to mixture components).

To create a two-level hierarchical Dirichlet process, we begin with an ordinary Dirichlet process as described in §2.2, with any sort of finite base measure. A draw from this "first layer" Dirichlet process, a discrete distribution over countably infinite atoms drawn from the base measure, is scaled by some constant positive factor, *then used as the base measure for a number of "second layer" Dirichlet processes*. Further layering is possible by passing draws from the second layer DPs on to another level of DPs, but two layers often suffice for most applications. This procedure is visualized in Figure 2.5, which also depicts one important subtlety: since the base measure for the subordinate Dirichlet processes is discrete, more than one sampled proportion will be paired with the same atom, as the second and fifth proportions are in the image. This phenomenon requires special handling during inference.

The depiction of the hierarchical Dirichlet process in Figure 2.5 can clarify how it happens that separate data collections can use the same mixture components. The continuous density that supplies base measure parameters is only used to generate the discrete distribution sampled from the first layer DP; in that sampled distribution (depicted at top right), which becomes the base measure for subordinate DPs, any atom that is probable at all is associated with a non-infinitesimal probability. Consequentially, a collection of subordinate DPs has a greater-than-zero chance of assigning the same atom to a proportion in more than one subordinate DP. Put differently, because all subordinate DPs are drawing atoms from a discrete base measure, it is possible for a subordinate DP to at some point draw the same atom that some other subordinate DP has used as well—or that many other subordinate DPs have used. In this way, the subordinate DPs share atoms.

If these atoms are parameters for mixture components, and if each data collection is modeled by a separate subordinate DP, then because the subordinate DPs share atoms, the

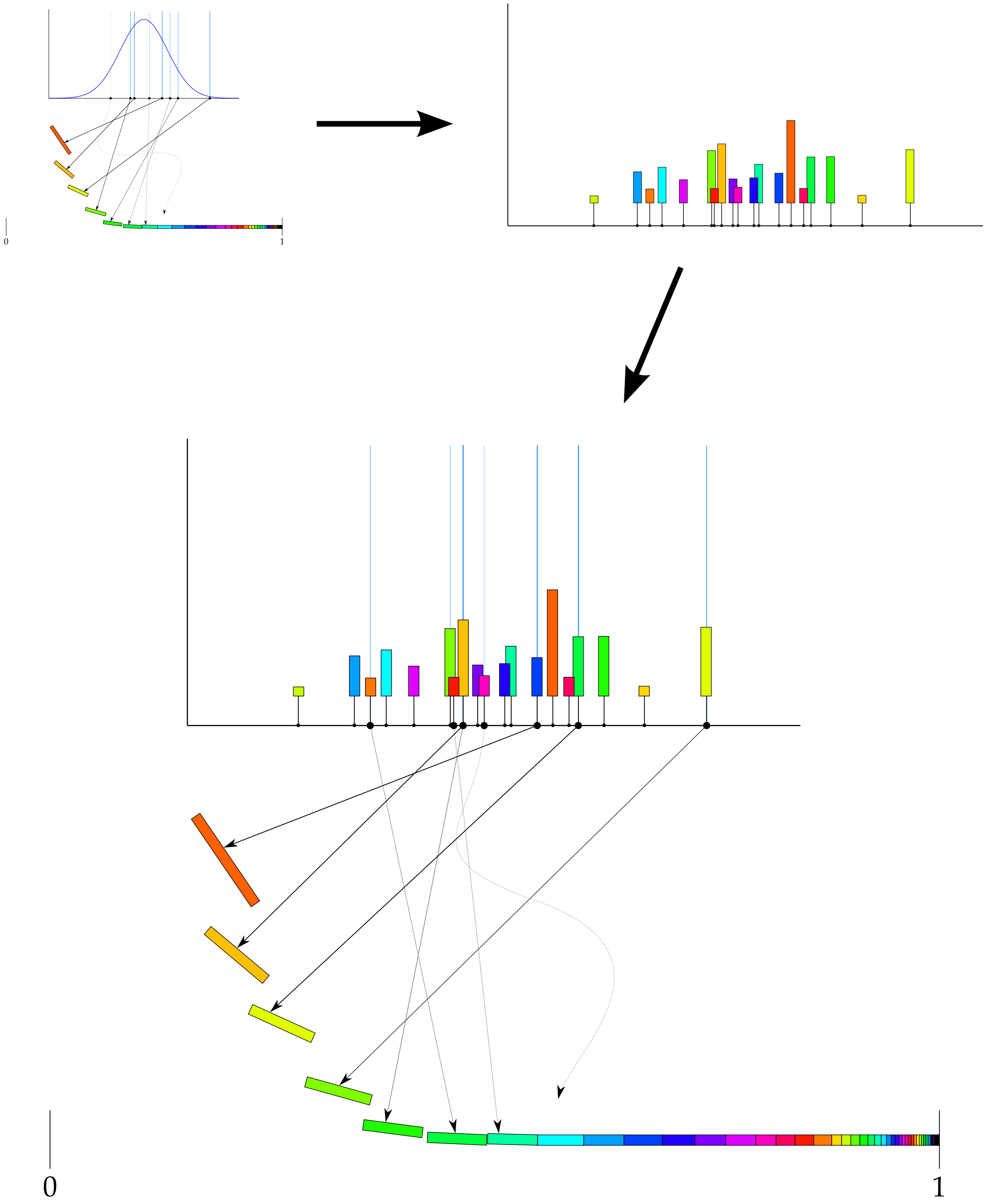


FIGURE 2.5: Visualization of a hierarchical Dirichlet process at work. A single discrete distribution (top right) is drawn from a Dirichlet process (§2.2); lengths of the colored bars indicate the probabilities of the associated points on the real line, and only a few of the countably infinite outcomes are shown. The discrete distribution becomes the base measure for *all* subordinate Dirichlet processes, including the one shown at center. Each subordinate process associates a *unique* set of sampled proportions with a *unique* set of draws from the *same* discrete base measure.

.

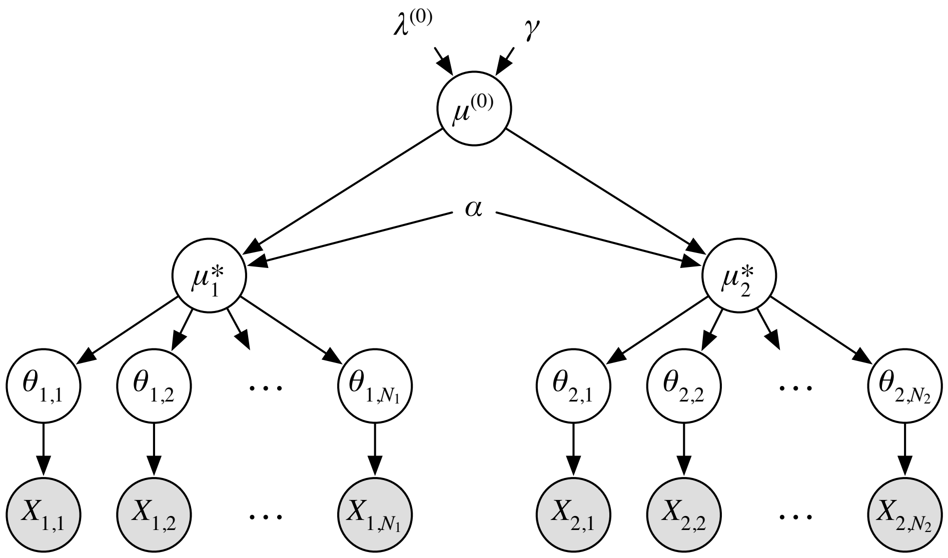


FIGURE 2.6: A graphical depiction of a two-layer hierarchical Dirichlet process mixture model with two collections of data. Here, the root base measure λ has been factored as $\lambda = \gamma\lambda^{(0)}$, where $\lambda^{(0)}$ is a probability measure and γ is a positive constant. Likewise, the probability mass function $\mu^{(0)}$ is scaled by the positive constant α to yield the base measure for the subordinate Dirichlet processes.

mixture components using the same mixture parameters can be inferred to have generated some of the data in both collections. In the original Dirichlet process mixture model of Figure 2.3, we are familiar with deeming two data points X_i and X_j as “clustered” together if their corresponding inferred mixture parameters θ_i and θ_j are the same. We can make a similar judgment for a hierarchical Dirichlet process mixture model, whose graphical representation appears in Figure 2.6. Certainly there will be cases where data points in the same collection $X_{m,i}$ and $X_{m,j}$ have the same generating parameters $\theta_{m,i}$ and $\theta_{m,j}$, and we would consider these to be in the same cluster. There will also be data points in other collections $X_{n,k}$ whose mixture parameters $\theta_{n,k}$ are also the same as $\theta_{m,i}$. It is in this way that the HDP mixture model clusters data across collections.

With this, we conclude the introduction to mathematical concepts that appear frequently in this document, and hopefully there is sufficient background here to support the discussion that follows. This introduction notably omits a treatment of inference techniques for hierarchical Dirichlet processes, but a detailed characterization of several sampling-based techniques appears in the HDP journal article [1]. These techniques, closely related to the inference methods we use for extensions to the infinite hidden Markov model, are described in detail below.

2.4 The infinite hidden Markov model

An advertisement for the infinite hidden Markov model (IHMM) might claim that it has something for everyone—or at least everyone who likes to think about Bayesian modeling of time series. For the consumer of analytical methods, the IHMM can "fit" a hidden Markov model to data without knowing beforehand how many states are necessary for a good representation. For the more theoretically minded, the IHMM describes a versatile, tractable probability distribution over hidden Markov models with a countably infinite number of hidden states. This is a noteworthy capability—in normal hidden Markov models, the number of parameters is proportional to the square of the number of states, and one might expect that countably infinite states would require storing and computing with impossibly large matrices. Thanks to the IHMM's clever use of Dirichlet processes in its prior, such heroic labors in bookkeeping are unnecessary. Because of its flexibility and elegance, the IHMM has seen applications as diverse as genetic analysis [91] and modeling of music [45].

The IHMM was first presented in [16]—a paper published before the formal introduction of hierarchical Dirichlet processes. As such, it employs a hierarchical Pólya urn scheme which, while similar to the arrangement one gets when applying the Chinese Restaurant Process to HDP inference, is nevertheless different in certain key respects. The revised IHMM appearing in the HDP paper [1] is a direct application of the HDP to time series modeling (it is also known as the HDP-HMM for this reason) and is usually considered to have superseded the older version. Only this newer, second IHMM will be discussed in this dissertation document.

One characterization of the IHMM begins with a somewhat unorthodox way to think about hidden Markov models in general [92]. Consider the collection of three mixture models in Figure 2.7. The models share the same three mixture components, but they differ in the proportional weights that determine how much each component contributes to the mixture. Samples drawn from the mixture reflect this variation.

We can imagine sampling a sequence from this collection of mixtures as follows. Starting at the leftmost mixture, we first select one of the mixture components according to the proportions, then sample a data point from the selected component. The mixture we will use for the next sample now depends on which component we just selected: if we used the bottom-left component, we remain in the first mixture; the top component takes us

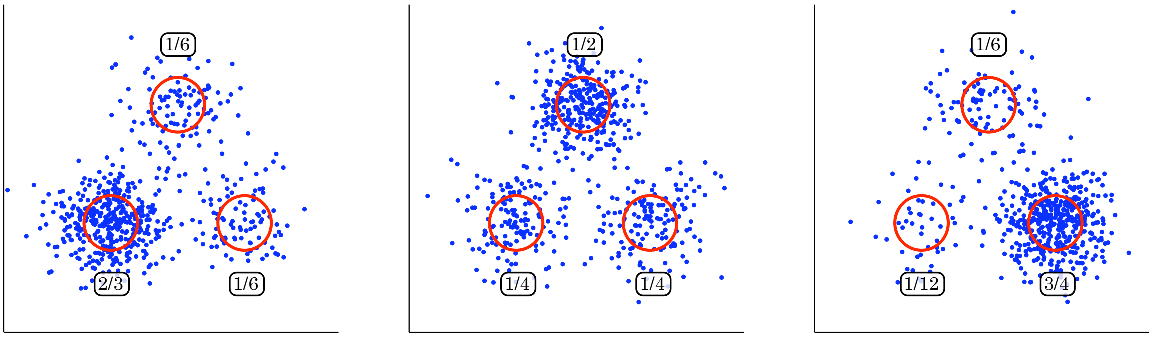


FIGURE 2.7: Three mixtures of three 2-D spherical Gaussian distributions. Covariance ellipses of the mixture components are shown in red, mixing proportions are the fractional quantities near the components, and blue dots show samples from the mixtures. The three models only differ in their mixing proportions; the mixture component parameters are the same for each. This figure and 2.8 help illustrate a conceptualization of hidden Markov models described in the text.

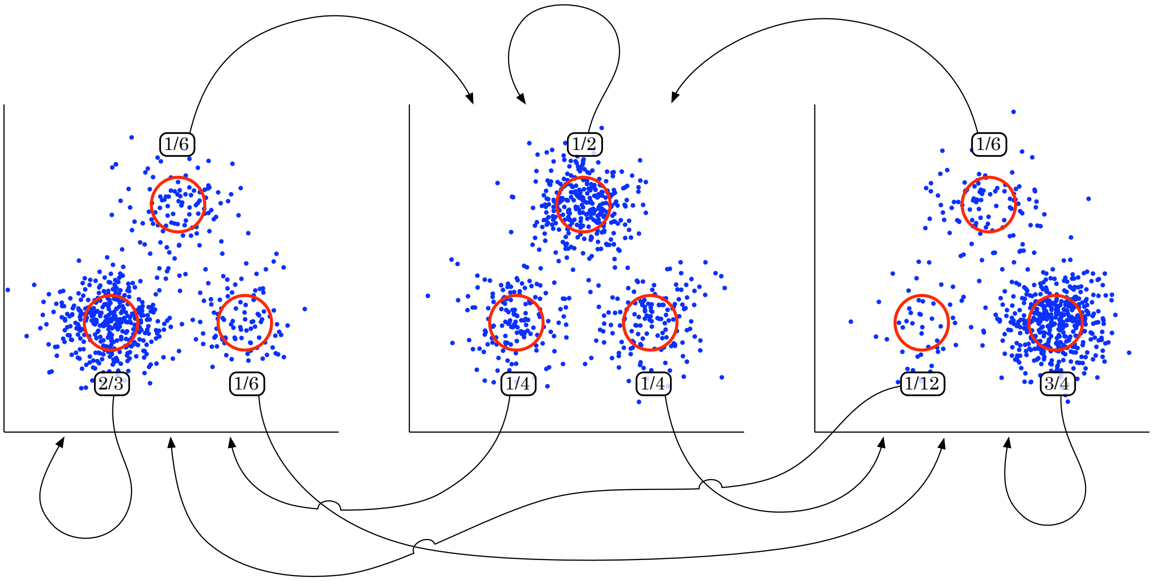


FIGURE 2.8: Transition relationships have been added to the mixture models in Figure 2.7 to portray the data generating procedure described in the text.

to the second mixture; the bottom right component sends us to the third mixture. These transition relationships among all the components and mixtures appears in Figure 2.8.

This arrangement of mixture models and transitions is slightly different from the one in most characterizations of hidden Markov models, which have an iterated informal "storyline" for generating each data point that goes something like this:

1. Conditioned on the current hidden state at time step t, sample a new hidden state for time step $t+1$.

2. "Jump to" the new hidden state.

3. Using the emission model belonging to this new hidden state, sample an observation for time step $t + 1$.

The new model has the following:

1. Conditioned on the current mixture model at time step t, sample a mixture component for drawing the observation at time step $t + 1$.

2. Using the selected mixture component, sample an observation for time step $t + 1$.

3. "Jump to" the mixture model corresponding to the selected mixture component.

No matter how different these narratives may seem, the mathematics behind both are identical. If we make the emission models in the first account the same as the mixture components in the second, and if we set the mixture weights to the same values as the transition probabilities in the "conventional" HMM, both entities will represent the same distribution over sequences. Selecting a new hidden state conditioned on the present state, or a mixture component conditioned on the current mixture model, involves drawing from the same categorical distribution. Sampling an observation from an emission model or a mixture component is likewise the same operation—as long as the mixture components and the emission models are identical. Finally, at the end of each iteration of the storylines, we are poised to sample the same distributions again in either case.

The conceptualization of HMMs just introduced, involving collections of mixture models with identical but differently weighted mixture components, makes it clear to see how hierarchical Dirichlet processes can be applied to time series modeling. Indeed, because the HDP mixture model described in §2.3 generates collections of mixture models with (infinitely many) identical but differently weighted mixture components, it seems nearly purpose-built for the job.

Because the mixture component weights in the IHMM have a new significance as transition parameters, it is convenient to use the stick-breaking construction as a tool for imagining how to generate a sample from the model. Compared to other ways of thinking about DP and HDP models, the stick-breaking construction very cleanly separates the generation of infinite vectors of proportions from the selection of atoms from the base measure. In mixture models, these vectors are the mixing proportions for mixture components, and in the IHMM they can be considered one of the countably infinity of countably infinitely long rows of the hidden state transition matrix. Explicit representation of this matrix in generative accounts of the IHMM is common, probably because it simplifies comparisons between the IHMM and other HMM techniques.

Before we present the stick-breaking generative account of the IHMM, we must introduce a special adaptation of the stick-breaking construction that simplifies the sampling of weights in the subordinate DPs. While the construction in §2.3 is correct, there is a more compact way to determine how much mass to associate with each subordinate DP atom, one that involves breaking off a single stick for each first-layer DP atom, rather than breaking off many sticks and sampling the atoms that get associated with them. As with our derivation of the stick-breaking interpretation in §2.2.3, we imagine sampling a subordinate DP by starting out with a generalized Pólya urn. This time, the normalized base measure $\mu^{(0)}$ is the discrete distribution sampled from the first-layer DP, though for simplicity we will continue to assume that the first-layer DP's base measure was continuous.

Let us denote the sizes of the broken sticks sampled in the first-layer DP as $\beta_1, \beta_2, \ldots$, and the atoms (or "colors") associated with those sticks as $\theta_1, \theta_2, \ldots$. Accordingly, before any draws from the subordinate DP's Pólya urn, the mass of balls there that are associated with θ_1 is proportional to β_1, and that the mass of the balls associated with other θ is proportional to $\beta_2 + \beta_3 + \cdots = 1 - \beta_1$.[2] We say "proportional to" because we can multiply all of the β masses by some positive constant α to modulate the variance of the draws from the subordinate DP (c.f. §2.1.2); therefore, the true base measure for the subordinate DP is $\mu = \alpha\mu^{(0)}$, and the true masses are $\alpha\beta_1, \alpha\beta_2$, and so on.

Invoking the Dirichlet aggregation property, we can sample the proportion of mass in the subordinate DP's Pólya urn that will be associated with θ_1 after infinite draws:

$$\pi_1 \sim \text{Beta}(\alpha\beta_1,\ \alpha(1-\beta_1)).$$

With this sampled proportion fixed, we know that the remaining $1-\pi_1$ proportion of the mass in the urn is associated with the other colors $\theta_1, \theta_2, \ldots$. We can invoke the Dirichlet aggregation property again, along with the subset-proportions-are-Dirichlet property, to sample the amount of *that* mass associated specifically with θ_2, as

$$\pi_2' \sim \text{Beta}(\alpha\beta_2,\ \alpha(1-\beta_1-\beta_2)).$$

[2] Note also by definition that $\mu^{(0)}(\theta_m) = \beta_m$ if (and only if) θ_m is an atom drawn in the first-level DP.

With this, the overall mass associated with θ_2 is $\pi_2 = (1 - \pi_1)\pi'_2$. We continue portioning out the remaining bits of mass in this way; for π_3, we would draw

$$\pi'_3 \sim \text{Beta}(\alpha\beta_3,\ \alpha(1 - \beta_1 - \beta_2 - \beta_3)),$$

then compute $\pi_3 = (1 - \pi_1 - \pi_2)\pi'_3$, and so on. For notational convenience, we will denote this method of sampling proportions conditioned on some prior set of proportions $\boldsymbol{\beta}$ and a constant positive scalar parameter α with $\text{SBP2}(\alpha, \boldsymbol{\beta})$. A general form for this procedure follows:

$$\text{SBP2}: \quad \begin{aligned} \pi'_m &\sim \text{Beta}(\alpha\beta_m, \alpha(1 - \textstyle\sum_{n=1}^{m} \beta_n)) \\ \pi_m &= \left(1 - \sum_{i=1}^{m-1} \pi_i\right) \pi'_m. \end{aligned}$$

Likewise, the original stick-breaking method of generating proportions (§2.2.3), conditioned only on the positive scalar parameter γ, is denoted by $\text{SBP1}(\gamma)$ from here on. For easy reference, we summarize this method again:

$$\text{SBP1}: \quad \begin{aligned} \beta'_m &\sim \text{Beta}(1, \gamma) \\ \beta_m &= \left(1 - \sum_{i=1}^{m-1} \beta_i\right) \beta'_m. \end{aligned}$$

We can now present a generative characterization of the IHMM that uses these stick-breaking processes together to create infinitely long rows $\boldsymbol{\pi}_m$ (note bold $\boldsymbol{\pi}$ symbol) of transition probabilities (or, equally, mixture proportions) associated with hidden state m—first by making the first-layer DP's mixing proportions ($\boldsymbol{\beta}$), which can be thought of as taking the role of a template for the proportions in subordinate DPs. Also generated are emission models for each state θ_m, and finally a sequence of hidden states $\boldsymbol{v}$ and the corresponding observations $\boldsymbol{y}$ that make up the data sequence. The generative process is:

$$\begin{aligned} \boldsymbol{\beta} \,|\, \gamma &\sim \text{SBP1}(\gamma) & & \\ \boldsymbol{\pi}_m \,|\, \alpha_0, \boldsymbol{\beta} &\sim \text{SBP2}(\alpha_0, \boldsymbol{\beta}) & \qquad v_t \,|\, v_{t-1}, \boldsymbol{\pi} &\sim \boldsymbol{\pi}_{v_{t-1}} \\ \theta_m \,|\, H &\sim H & y_t \,|\, v_t, \boldsymbol{\theta} &\sim g(\theta_{v_t}). \end{aligned}$$

A graphical model representation of this process appears in Figure 2.9. Figure 2.10 shows (truncated) transition matrices and hidden state trajectories drawn from the generative process just described. Because most of the mass in each row is assigned to transitions to the first several states in the model, hidden state trajectories will tend to dwell in these states rather than wandering off permanently into other regions of the countably infinite

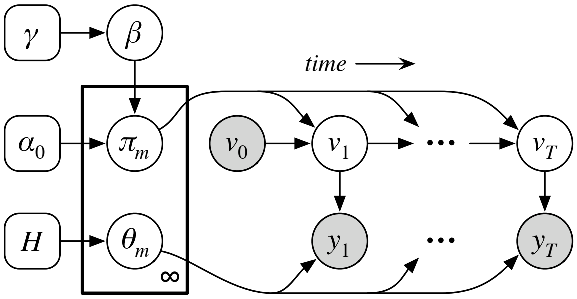


FIGURE 2.9: Graphical model depiction of the infinite hidden Markov model. Descriptions of the variables are furnished in the text.

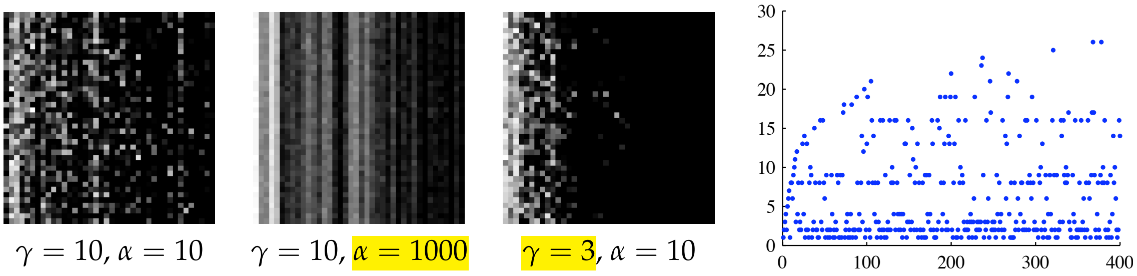

$\gamma = 10, \alpha = 10$ $\gamma = 10, \alpha = 1000$ $\gamma = 3, \alpha = 10$

FIGURE 2.10: At left and center, three truncated IHMM transition matrices drawn via the SBP1/SBP2 stick-breaking methods. Highlights show which parameters are mainly responsible for the two inner matrices' differences from the leftmost matrix; as can be seen, γ modulates how many states have high probability overall, while α affects how similar states' transition dynamics are. At right, a hidden state trajectory sampled from an IHMM transition matrix ($\gamma = 10, \alpha = 10$); x axis is time steps; y axis is numeric state IDs in the order first visited. The number of unique states visited in a state sequence is roughly logarithmic in the length of the sequence.

state space. At right in the figure, this behavior is shown in a plotted hidden state trajectory.

Critically, although the figure shows representative samples from the IHMM, *posterior* IHMM samples will exhibit more useful dynamics than the ones shown here, which come from what serves as the prior. Indeed, an IHMM posterior distribution will represent a classic Bayesian tradeoff between the likelihood function's pressure to describe the data well (usually by allocating many states to describe the sequence) with the prior's preference for parsimonious trajectories (i.e. hidden state sequences concentrated on a small set of significant states). The structure of the prior, demonstrated by the samples in Figure 2.10, induces a useful structural characteristic—conciseness—in inferred representations without imposing strict restrictions like a fixed number of states. This gentle guidance is essentially what makes the IHMM a compelling technique for data analysis.

We will omit a description of IHMM posterior inference from this section, since the inference procedure for the block-diagonal IHMM generalizes these techniques, and its specialization for the IHMM case is not difficult to envision or derive. For examples of the IHMM in action, we refer the reader to the application papers cited earlier [45, 91].

Chapter 3

The Block-Diagonal Infinite Hidden Markov Model

The previous chapter indicates that the structure of the infinite hidden Markov model promotes parsimonious structure in inferred representations of sequential data, a consequence of the way the hierarchical Dirichlet process generates the proportions that determine the IHMM's state-to-state transition behavior. We might therefore wonder whether an even more structured prior could induce additional useful structure in inferred representations.

A variety of sequential data exhibit what could be called *sub-behaviors*, in that they seem to alternate among a collection of distinct dynamic regimes. Examples of such data include musical recordings, which might switch between a number of musical themes, or video clips of a person communicating in sign language. A hidden Markov model designed to capture or generate the dynamics of processes like these would exhibit a nearly block-diagonal structure in its transition matrix, with individual blocks of relatively high-probability transitions corresponding to specific sub-behaviors, and less probable transitions between blocks supporting alternation between the sub-behaviors.

The block-diagonal infinite hidden Markov model (BD-IHMM) is an extension of the IHMM whose prior induces nearly block-diagonal structure in inferred hidden state transition dynamics. In doing so, it explicitly associates each state with a specific sub-behavior in an unsupervised way. Like the IHMM, the BD-IHMM employs mass allocation methods derivable from the Pólya urn to govern the arrangement of hidden states and blocks that describe the data, so similar Bayesian tradeoffs between efficient and descriptive models are present in the BD-IHMM as well.

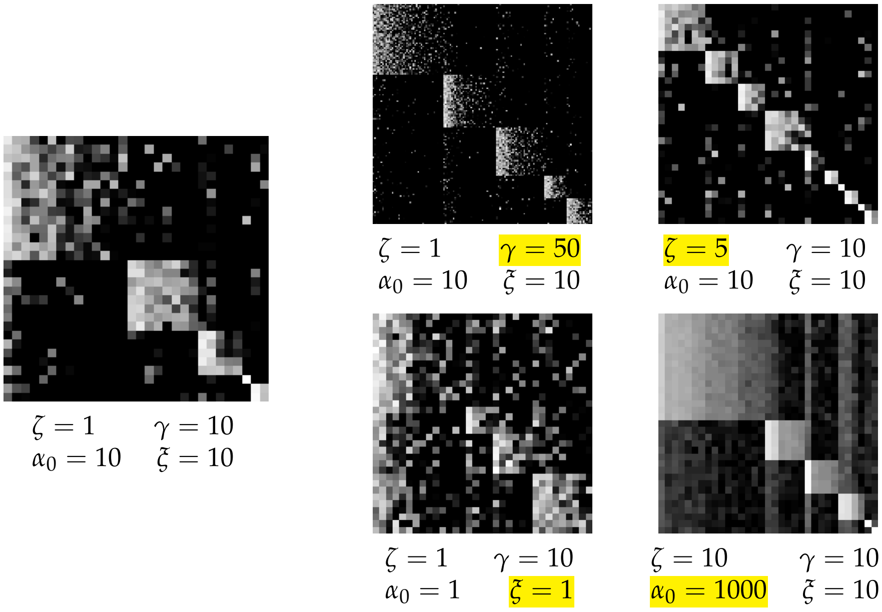


FIGURE 3.1: Truncated Markov transition matrices sampled from the BD-IHMM prior with various fixed hyperparameter values; highlighted hyperparameters yield the chief observable difference from the leftmost matrix. The upper left matrix has more states; the upper right more blocks; the lower left stronger transitions between blocks, and the lower right decreased variability in transition probabilities. States have been resorted by block label in all matrices.

3.1 Characterization of the model

The BD-IHMM imposes block structure on transition dynamics via a two-step process. First, it assigns each of the countably infinite hidden states to one of a countably infinite set of sub-behavior blocks. Next, for each state i, it selectively modifies the proportions $\boldsymbol{\beta}$ in the base measure such that in the sampled transition dynamics $\boldsymbol{\pi}_i$, transitions to states within the same block are emphasized relative to transitions to states in different blocks. These new behaviors are modulated by additional hyperparameters ζ and ξ, which affect the relative probabilities of different blocks and the relative amount of enhancement given to within-block transitions respectively. Figure 3.1 displays example truncated transition matrices drawn from the IHMM generative process, illustrating the block-diagonal structure imposed by the model, as well as representative effects of different hyperparameter values.

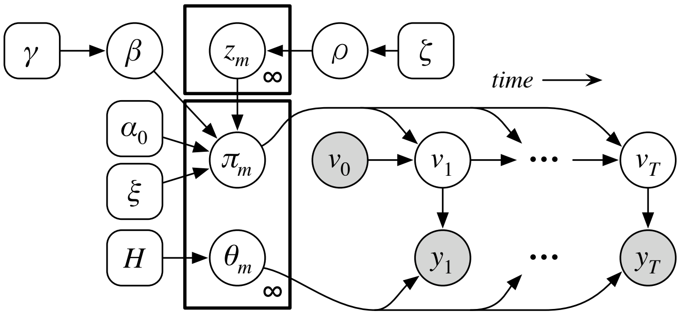


FIGURE 3.2: Graphical model depiction of the block-diagonal infinite hidden Markov model.

The generative process for the BD-IHMM is as follows:

$$\boldsymbol{\beta} \,|\, \gamma \sim \mathrm{SBP1}(\gamma)$$

$$\boldsymbol{\rho} \,|\, \zeta \sim \mathrm{SBP1}(\zeta) \qquad\qquad z_m \,|\, \boldsymbol{\rho} \sim \boldsymbol{\rho}$$

$$\beta^*_{mn} = \frac{\beta_n}{1+\xi}\left(1 + \frac{\xi \cdot \delta(z_m = z_n)}{\sum_k \beta_k \cdot \delta(z_m = z_k)}\right)$$

$$\boldsymbol{\pi}_m \,|\, \alpha_0, \boldsymbol{\beta}^*_m \sim \mathrm{SBP2}(\alpha_0, \boldsymbol{\beta}^*_m) \qquad\qquad v_t \,|\, v_{t-1}, \boldsymbol{\pi} \sim \boldsymbol{\pi}_{v_t - 1}$$

$$\theta_m \,|\, H \sim H \qquad\qquad y_t \,|\, v_t, \boldsymbol{\theta} \sim g(\theta_{v_t}).$$

The first line and the last two lines of this process description are nearly identical to those of the IHMM from the last chapter. What's new comes from the other two lines, the first of which uses the stick-breaking process to sample proportions $\boldsymbol{\rho}$ for a categorical distribution over block labels with countably infinite outcomes. It then samples block labels $\boldsymbol{z}$ for each of the countably-infinite hidden states. The next line describes how the subordinate Dirichlet process base measure masses $\boldsymbol{\beta}^*$ are computed for transitions out of each hidden state.

A graphical model depiction of this process appears in Figure 3.2. Before describing inference for the BD-IHMM, however, we shall elaborate on the base measure modification at the heart of this new model, offering some insight into what it does and what we need to know to perform effective inference.

3.2 Base measure probability mass modification illustrated, and considered

Within the subordinate Dirichlet processes employed by the BD-IHMM, the base measure probability mass associated with a transition from hidden state m to hidden state n is computed as

$$\beta^*_{mn} = \frac{\beta_n}{1+\xi}\left(1 + \frac{\xi \cdot \delta(z_m = z_n)}{\sum_k \beta_k \cdot \delta(z_m = z_k)}\right). \tag{3.1}$$

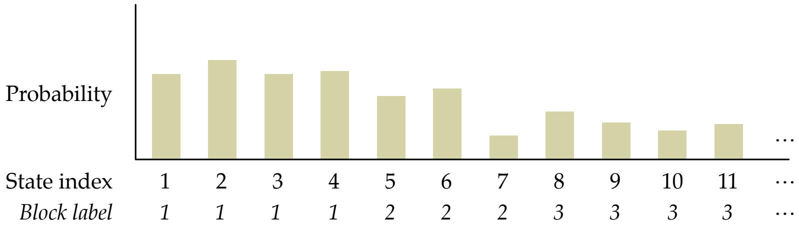
This modification of the mass β_n sampled from the top-layer Dirichlet process can be unpacked as follows. First, we have a normalization term $\frac{1}{1+\xi}$ so that the sum of the modified masses associated with all transitions out of state m is equal to one. This term, along with other structure in the modification equation, permits the continued clean factorization of the base measure used in subordinate DPs into shape (mean) $\boldsymbol{\beta}^*_m$ and strength (inverse variance) α_0. Although this factorization is not necessary to achieve a well-formed model or even convenient inference, we consider it worthwhile to preserve structure that might assist those who need to review and interpret posterior samples of BD-IHMM parameters inferred from their data.

Within the parentheses there are two terms: a 1, which ensures that a "baseline" mass of $\frac{\beta_n}{1+\xi}$ is associated with each transition, and what we will refer to as the *augmentation term*. The delta function in the augmentation term numerator, $\delta(z_m = z_n)$, ensures that additional mass is associated with the transition when and only when both the source and the destination hidden state are within the same block. When this occurs, the amount of extra mass added to the base measure probability for this transition is equal to

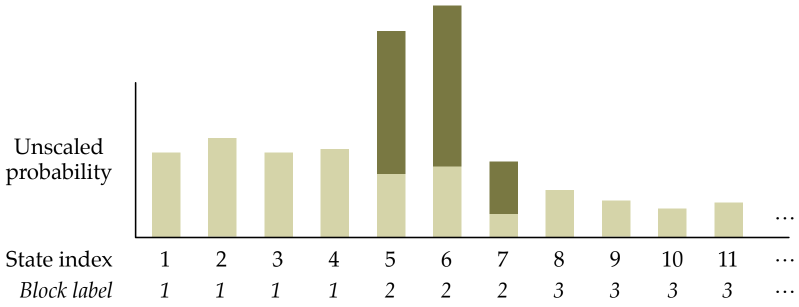
$$\frac{\beta_n}{1+\xi}\frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}.$$

We must observe before getting much further ahead that given any destination state n, β^*_{mn} will be the same for all states m in the same block as n. The per-transition notation is somewhat simpler, however, and it will be good to be familiar with this idiom when we present generalizations in Chapter 5.

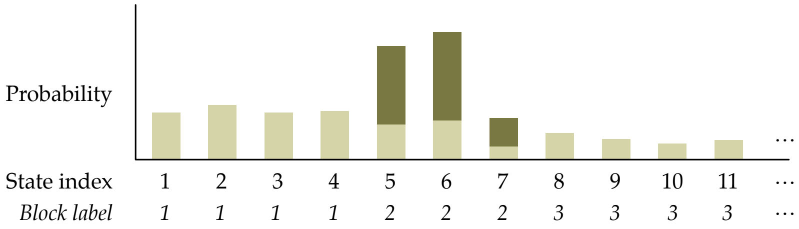
Luckily, there is some reasoning and intuition behind what we've just unpacked. Recall that within the hierarchical Dirichlet process and related models, the base measure proportions used by the subordinate DPs will behave as the mean proportions for the discrete distributions sampled from those subordinate DPs. In another manner of speaking, $\boldsymbol{\beta}^*_m$ serves as a template for the probabilities of transitions out of hidden state m. The modification performed by Equation 3.1 creates a template that expresses the model's block-diagonal structure. To do this, it takes the original set of proportions sampled from the top-layer DP—the template for the template, so to speak—



and then it scales the masses associated with all of the states in a particular block by a scalar factor until the total mass adds up to $1 + \xi$.



Finally, it renormalizes all of the masses so that they sum to 1 again.



Note that in the three preceding illustrations, only a finite number of hidden states appear to be allocated to each block. In the actual model, each block actually has a countably infinite number of hidden states.

The formal definition of the Dirichlet process (§2.2.1), particularly as it reflects the aggregation property of Dirichlet distributions (§2.1.2), give us further insight into how the modifications to $\boldsymbol{\beta}$ influence the dynamics of the BD-IHMM. Consider a circumstance where the original mass in $\boldsymbol{\beta}$ belonging to states in block z_m add to some value x, or, $\sum_k \beta_k \cdot \delta(z_m = z_k) = x$. The mass belonging to the remaining states must add up to $1 - x$. Once we modify $\boldsymbol{\beta}$ for transitions out of states in block z_m, however, the total mass for the block z_m states in $\boldsymbol{\beta}^*$ sums to $\frac{x+\xi}{1+\xi}$, while the remaining mass sums to $\frac{1-x}{1+\xi}$.

Since $\boldsymbol{\beta}^*$ is the template for sampled transitions out of states in block z_m, these ratios take on a special significance. Let us consider some numerical values to make the meaning easier to interpret. If $x = 0.5$ and $\xi = 1$, then the ratios equal 0.75 and 0.25 respectively. This means that on average, if we have a Markov chain traversing the states in a hypothetical BD-IHMM matching these numbers, and if we observe it at a time step t in a state within block z_m, there is a 75% chance that it will transition to another block z_m state at time step $t + 1$, and a 25% chance that it will transition to a state in another block. Regardless of the variations involved in specific probabilities sampled for individual transitions, the "block-level" view of the effects of the probability modification scheme are useful for understanding the model conceptually, and also for interpreting the meaning of a posterior ξ value inferred from data. It will also be useful for devising inference schemes for $\boldsymbol{\beta}$, as we will see soon enough.

Hopefully these considerations along with the preceding illustrations clarify the nature of the base measure probability mass modification employed in the BD-IHMM. It is worth noting before moving on some of the variations, extensions, and alternatives to this scheme that are possible. One might observe, for example, that the base measure probability mass modification just described is multiplicative, in that it alters $\boldsymbol{\beta}$ values by multiplying them by some computed value. To induce an extra bias for transitions from a state to itself, the authors of [74] consider a per-state *additive* modification to the single base measure proportion corresponding to the self-transition. In the BD-IHMM, the number of transitions that must be enhanced is countably infinite, not singular, so adding a constant value to the base measure mass at each of these transitions would result in an intermediate collection of values (c.f. the second graph above) that sums to infinity. One might then suggest implementing some kind of scheme that prevents the amount of total additive modification from exceeding some prescribed value—we might call this ξ—but somehow this value must be divided up unevenly among $\boldsymbol{\beta}$ values corresponding to transitions to states in the same block to keep it from being spread infinitesimally thinly. We suspect that any additive scheme that fulfills these requirements and serves some practical use will begin to look not very different from the multiplicative approach the BD-IHMM uses anyway.

The denominator sum $\sum_k \beta_k \cdot \delta(z_m = z_k)$ in Equation 3.1 requires special care, as the next section will discuss. We might therefore be tempted to consider the following multiplicative base measure modification:

$$\beta^*_{mn} = \beta_n \cdot \xi^{\delta(z_m = z_n)}.$$

The simplicity of this expression pays dividends when devising the sampling-based inference schemes common to most applications of the hierarchical Dirichlet process—indeed, the schemes devised for the HDP and IHMM can be applied to models using this modification with very little additional effort. The chief objection to this approach is that, without some kind of normalization in place, it lacks the decoupling between the shape and the strength of the base measure that exists in the BD-IHMM's approach.

An example may help clarify this point. Consider a circumstance where 50% of the mass of $\boldsymbol{\beta}$ belongs to states associated with block 1, 20% belongs to states associated with block 2, and the remaining mass allocated somehow to states in the remaining blocks. Assuming $\xi = 5$, $\boldsymbol{\beta}^*_m$ vectors for states in block 1 will sum to 3; those for states in block 2 will sum to 1.8; and those for states in other blocks will assume further different values. These will be multiplied in turn by α_0 to yield the final masses that are paired with the atoms sampled by the top-layer DP, which in turn serve as the base measures for the corresponding subordinate DPs.

Because the total base measure mass determines how closely the proportions in the subordinate DP samples match the shape of the base measure, these base measures can be said to have varying strengths: those with more mass in the original $\boldsymbol{\beta}$ will be more closely matched by sampled transition probabilities than those with less mass there. One might venture that this is a somewhat arbitrary basis for determining the strength of a base measure. Base measures are priors, after all, and the strength of a prior should reflect the certainty of the assumptions it encodes. In the BD-IHMM, this certainty is related entirely by the global hyperparameter α_0, which can either be set directly by the user or sampled itself.

For some applications, however, it may be overly restrictive to use single global hyperparameters α_0 and ξ to relate, respectively, certainty about base measures and the degree of base measure probability mass modification. It may be the case, for example, that we wish transitions from states in block 1 to be 99% likely to remain in block 1, while transitions from states in block 2 should only have a 30% chance of staying in that block. In these settings, it may be appropriate to have per-block or even per-state α_0 and ξ hyperparameters. We leave closer examination of these somewhat more flexible models to future research.

3.3 A necessary approximation

The base measure probability mass modification in Equation 3.1 incorporates a sum over all the $\boldsymbol{\beta}$ values corresponding to hidden states in a particular block: $\sum_k \beta_k \cdot \delta(\beta_m = \beta_k)$. Since there are a countably infinite number of such states in each block, the exact computation of this sum requires us to have sampled a countably infinite number of $\boldsymbol{\beta}$ values. Clearly, instead of doing that, we must approximate this sum instead. This section examines a simple, easy-to-compute approximation strategy used for all results presented in this thesis.

To give a better intuitive feel for the problem, we consider a numerical example whose values are reflective of a typical application of the BD-IHMM. Assume we have a data sequence and a particular posterior sample of the BD-IHMM parameters that associates the data with nine states in three blocks. The block labels and $\boldsymbol{\beta}$ values for these states are as follows:

State index	1	2	3	4	5	6	7	8	9
Block label	*1*	*1*	*1*	*2*	*2*	*2*	*3*	*3*	*3*
$\boldsymbol{\beta}$ value	0.2	0.2	0.2	0.1	0.1	0.1	0.02	0.02	0.01

which means that the sum of the $\boldsymbol{\beta}$ values for the infinity of “unused” states must be 0.05. Meanwhile, let us imagine that the inferred block label probabilities came out to be:

Block label	*1*	*2*	*3*
$\boldsymbol{\rho}$ value	0.3	0.3	0.3

leaving a combined probability of 0.1 for all “unused” block labels in the posterior BD-IHMM sample. Finally, let’s assume an inferred ξ value of 5.

Consider now the modified β value for the transition from state 1 to itself. We have

$$\beta^*_{11} = \frac{0.2}{1+5}\left(1 + \frac{5}{0.2+0.2+0.2+\cdots}\right).$$

Filling in values for the $\cdots$, we can set lower and upper bounds on β^*_{11}. There is a vanishing chance that nearly all of the remaining states will be allocated to block 1, in which case nearly all of the remaining beta mass of 0.05 will be represented here. Using this value, we get $\beta^*_{11} \approx 0.2897$. There is also a chance that nearly all remaining states will be allocated to different blocks; here, filling in 0 for the missing summands, we get $\beta^*_{11} \approx 0.3111$. The difference between these values, around 0.0214, may seem fairly insignificant, but the modified base measures, as the priors for transition probabilities, play an important role in determining the outcome of other portions of BD-IHMM inference, particularly if α_0 is large. It behooves us to select a value for these missing entries judiciously.

One simple albeit expensive solution would be to sample $\boldsymbol{\beta}$ values and block labels $\boldsymbol{z}$ for enough of the unused states that the difference between the upper and lower bounds for each of the $\boldsymbol{\beta}^*$ values falls beneath some predetermined threshold. It is reasonable to set this forth as a kind of "gold standard" among approximations of the sum, since computed values of $\boldsymbol{\beta}^*$ can be made to come arbitrarily close to true samples from their distribution.

Luckily for us, it is possible to compute the expected value of $\sum_k \beta_k \cdot \delta(z_m = z_k)$ with respect to the distribution over $\boldsymbol{\beta}$ and $\boldsymbol{z}$, using a collection of nested integrals:

$$\begin{aligned}
E\left[\sum_{k=1}^{\infty} \beta_k \cdot \delta(z_m = z_k)\right] = \\
&\int_{\beta_1} \sum_{z_k} P(z_1 = z_k) \cdot \beta_1 \cdot \delta(z_m = z_k) \\
&\quad + \int_{\beta_2} \sum_{z_k} P(z_2 = z_k) \cdot \beta_2 \cdot \delta(z_m = z_k) \\
&\qquad + \int_{\beta_3} \sum_{z_k} P(z_3 = z_k) \cdot \beta_3 \cdot \delta(z_m = z_k) + \ldots \\
&\qquad\qquad \ldots dP(\beta_3 \mid \beta_1, \beta_2)\, dP(\beta_2 \mid \beta_1)\, dP(\beta_1),
\end{aligned}$$

which simplifies conveniently thanks to the delta functions:

$$\begin{aligned}
E\left[\sum_{k=1}^{\infty} \beta_k \cdot \delta(z_m = z_k)\right] = \int_{\beta_1} \rho_{z_m}\beta_1 + \int_{\beta_2} \rho_{z_m}\beta_2 + \int_{\beta_3} \rho_{z_m}\beta_3 + \ldots \\
\ldots dP(\beta_3 \mid \beta_1, \beta_2)\, dP(\beta_2 \mid \beta_1)\, dP(\beta_1).
\end{aligned}$$

The ρ_{z_m} can be pulled out of the nested integrals, which evaluate together to $\sum_{k=1}^{\infty} \beta_k = 1$, yielding the expected sum as, simply, ρ_{z_m}. Under circumstances like our example (and, indeed, nearly all BD-IHMM samples) where some of the $\boldsymbol{\beta}$ values are actually known, a nearly identical derivation will demonstrate that the expected sum is $\rho_{z_m}\beta_{\text{new}}$, where β_{new} is the sum of the $\boldsymbol{\beta}$ values for all the unused hidden states (0.05 in our example). This, then, is the simple approximation we use to fill in missing summands in Equation 3.1, yielding $\beta^*_{11} \approx 0.3043$.

How far off will this approximation tend to be? There are two answers to this question, depending on whether we are interested in how far off the approximation will tend to be from the true infinite sum $\sum_k \beta_k \cdot \delta(z_m = z_k)$, or whether we are interested in how far off the $\boldsymbol{\beta}^*$ estimates will tend to be from their true values. The latter is what truly matters for the sake of good inference, but the answer to the first question can also offer some insight into the approximation. For this, we consider the expected squared error of the sum estimate,

$$E\left[\left(\rho_{z_m} - \sum_{k=1}^{\infty} \beta_k \cdot \delta(z_m = z_k)\right)^2\right],$$

with respect to the joint distribution of $\boldsymbol{\beta}$, $\boldsymbol{\rho}$, and $\boldsymbol{z}$. Note that this quantity will be dependent on the parameters of this joint distribution, namely γ and ζ.

Since we cannot compute this quantity analytically, we settle for a numeric approximation that repeatedly compares ρ_{z_m} with a sum over a long but still finite $\boldsymbol{\beta}$ vector. Each iteration uses newly sampled $\boldsymbol{\rho}$ and $\boldsymbol{\beta}$ vectors, which are drawn by the stick-breaking process. The z_m we consider correspond exclusively to the larger ρ_{z_m} values in the $\boldsymbol{\rho}$ vector—an apt choice since the ρ_{z_m} values encountered in practice, the ones corresponding to blocks whose hidden states are actually associated with data, are likely to be the larger ones. The $\boldsymbol{\beta}$ vectors we use, meanwhile, are long enough that their later entries require special care while programming to avoid numerical underflow problems—this is the "gold standard" alluded to previously.

For a particular γ and ζ value, then, we sample $\boldsymbol{\rho}$, $\boldsymbol{\beta}$, and $\boldsymbol{z}$ many times. Each time, we compute the squared difference within the expectation for a few z_m values with large corresponding ρ_{z_m}. These computed squared differences are averaged to yield the estimate of the expectation. Figure 3.3 shows how this expectation varies over a range of practical γ and ζ values. In general, these values are fairly small, particularly for the larger γ values that tend to arise when more than a few states are needed to describe a dataset. Furthermore, the medians of the sampled squared errors, also shown in Figure 3.3, are smaller still, reflecting the fact that the errors of our $\boldsymbol{\beta}$ sum estimates are highly peaked around 0.

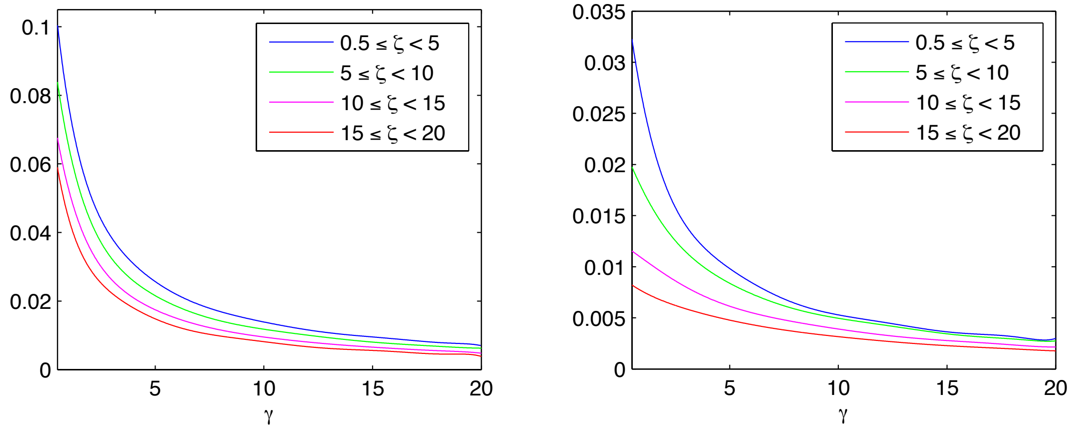


FIGURE 3.3: Left: curves showing the relationship between ζ, γ, and the expected squared error of the approximation of the infinite sum $\sum_k \beta_k \cdot \delta(z_m = z_k)$ used in this thesis. Right: here, the *median* squared error of the approximation. These curves are polynomial fits to error estimates generated by the sampling procedure described in the text.

We now explore how $\boldsymbol{\beta}^*$ approximations derived from these sum estimates differ from their true values. Although it is a matrix, it is instructive to consider a single value within $\boldsymbol{\beta}^*$—here, we scrutinize β^*_{j8} for any j such that $z_j = z_8$; note that β^*_{j8} will be the same value for all such j. We further assume that we maintain 15 values of $\boldsymbol{\beta}$, β_1 through β_{15}, as well as the sum of "leftover" $\boldsymbol{\beta}$ values β_{new}. This scenario matches well with typical BD-IHMM inference scenarios with a modest number of states (15 in this case), as the next subsection will show. We are interested in the difference between the estimate $\hat{\beta}^*_{j8}$:

$$\hat{\beta}^*_{j8} = \frac{\beta_8}{1+\xi}\left(1 + \frac{\xi}{\rho_{z_j}\beta_{\text{new}} + \sum_{k=1}^{15} \beta_k \cdot \delta(z_k = z_j)}\right),$$

and the true value β^*_{j8}:

$$\beta^*_{j8} = \frac{\beta_8}{1+\xi}\left(1 + \frac{\xi}{\sum_{k=1}^{\infty} \beta_k \cdot \delta(z_k = z_j)}\right).$$

Here again we take a numerical approach similar to the one we used to explore $\boldsymbol{\beta}$ summing. In this case, ξ is an additional parameter to consider, so we vary this as well within a sensible range.

Depending on the sampled $\boldsymbol{\beta}$ vector, the correct β^*_{j8} value can vary through a wide range of scales. For this reason, it may be more informative to consider the ratio of $\hat{\beta}^*_{j8}$ to β^*_{j8}, rather than their absolute or squared differences. Samples of this ratio showed very little

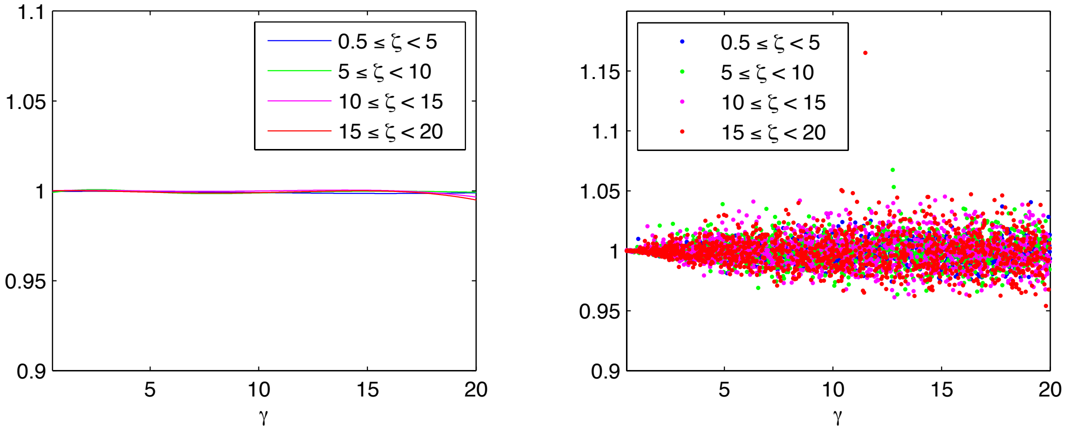


FIGURE 3.4: Left: curves showing the relationship between ζ, γ, and the estimated value of the ratio $\hat{\beta}^*_{j8}/\beta^*_{j8}$ discussed in the text. At right, the data from which the curves were derived.

dependence on the value of the ξ, so further discussion will consider the effects of varying only γ and ζ. Figure 3.4 makes a good case for the expected value of this ratio being equal to 1 within our parameter range.

To conclude this subsection, we feel that these investigations demonstrate that the expected sum approximation to the modification term denominator sum in Equation 3.1 is suitable for estimating the values of $\boldsymbol{\beta}^*$ terms for general inference purposes. Should higher fidelity be desired, however, we can recommend the "gold standard" sampling-based approach described at the beginning of this section, which may be brought arbitrarily close to true samples from the distribution over $\boldsymbol{\beta}^*$ values, and whose divergence from such samples may be estimated with well-defined error bounds. To date, we have not had the opportunity to compare our approximation with the "gold standard" in practical settings (that is, to determine what, if any, differences might arise in inference outcomes if both methods are used to apply the BD-IHMM to actual data), since the implementation of this second approach would require considerable reworking of our existing inference code. We leave this important effort to future work.

3.4 Inference

In a typical BD-IHMM application setting, the user will have one or more unlabeled time series datasets and little or no additional quantitative information about the generating process that created the data. He may suspect, however, that this process can be nearly decomposed into a collection of sub-behaviors, and finally he may believe that the process can be sufficiently characterized by a Markov chain operating on a set of discrete states (as opposed to e.g. a continuous hidden state space). Under these circumstances, the goal of BD-IHMM inference is to sample parameters of the model conditioned on the data. These posterior parameters comprise the hidden state trajectory of the generating Markov chain ($\boldsymbol{v}$), the hidden state observation models ($\boldsymbol{\theta}$), the association of hidden states with sub-behavior blocks ($\boldsymbol{z}$), the base probabilities of transitions to at least each of the hidden states visited by the Markov chain ($\boldsymbol{\beta}$), the probabilities of allocating a hidden state to at least each of the blocks occupied by visited states ($\boldsymbol{\rho}$), and finally the four hyperparameters (α_0, ξ, γ, ζ).

Under some circumstances, the user might choose to fix some of the hyperparameters or set tighter priors on their values to encode further information or assumptions about the data—to indicate that the number of states should be large relative to the size of the data, for example, a large γ hyperparameter may be used. Nevertheless, if the model itself is more or less appropriate for the data, it is intended that an inference technique should be able to generate good posterior samples with minimal intervention.

In this section we present a Gibbs sampling approach to drawing samples of BD-IHMM parameters conditioned on data. As such, it alternates between sampling new values for each of the quantities listed above, one by one, while holding all of the other quantities fixed. Like all Markov chain Monte Carlo (MCMC) sampling techniques, this procedure is iterated from some initial starting point until the sampler converges on what appears to be a high probability region. Once there, subsequent Gibbs sampling iterations are considered to produce samples from the posterior. Although in principle it is difficult to recognize when this regime of posterior sampling "officially" begins—a problem typical to most non-trivial applications of MCMC techniques—in practice, convergence is usually characterized by the BD-IHMM sampler failing to make additional, lasting "major" alterations to the parameters, such as creating or deleting states and blocks, by the sampler alternating between a small number of similar, repeated configurations, or by running several parallel samplers and detecting when their samples are similar to each other.

Finally, like many approaches to inference in hierarchical and ordinary Dirichlet processes, the sampler does not always represent the transition probabilities $\boldsymbol{\pi}$ explicitly (except in one specific circumstance) but instead integrates them out of the conditional probabilities that make up the inference. This marginalization step realizes the Chinese restaurant process described in §2.2.5. Most BD-IHMM applications do not require explicit samples of $\boldsymbol{\pi}$, but for those that do, drawing them *post facto* conditioned on the rest of a BD-IHMM parameter posterior sample is not particularly difficult (c.f. §3.4.5).

We now consider each of the Gibbs sampling steps individually before describing initialization and the overall sampling scheme.

3.4.1 Observation model parameters (θ)

As a warm-up, we consider one of the easier components of BD-IHMM inference, the sampling of parameters for observation models corresponding to each of the visited hidden states. This step is no different from corresponding steps in ordinary HMM or mixture model inference. For a particular hidden state m, we draw θ_m as

$$\theta_m \mid \boldsymbol{y}, \boldsymbol{v}, H \sim H(\theta_m) \prod_{\{t:\, v_t = m\}} p(y_t \mid \theta_m). \tag{3.2}$$

In some settings, it is not even necessary to represent observation model parameters explicitly, since these too can be marginalized out. This "collapsed" approach [93] has, in our experience, mixed success in the BD-IHMM and in the Dirichlet process mixture model setting for which it was originally devised.[1] We encourage users to evaluate both approaches where possible.

3.4.2 Block label probabilities (ρ)

Another warm-up inference problem is the block label probabilities $\boldsymbol{\rho}$, usually sampled on-demand whenever any of the other inference components require it. Typically, the only entries of $\boldsymbol{\rho}$ that are necessary to sample are those corresponding to block labels assigned to hidden states visited in the inferred trajectory $\boldsymbol{v}$, as well as one additional entry ρ_{new}, the sum of $\boldsymbol{\rho}$ values for all of the "unused" block labels. Via straightforward application of the aggregation property (c.f. §2.1.2), we draw this as

$$\{\rho_1, \rho_2, \ldots, \rho_K, \rho_{\text{new}}\} \sim \text{Dir}(Q_1, Q_2, \ldots, Q_K, \zeta), \tag{3.3}$$

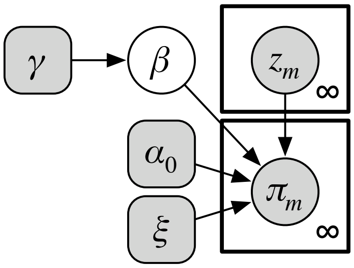
[1] These experiments are not reported in this document.

where Q_j is the number of "used" hidden states (i.e. those visited in the trajectory $\boldsymbol{v}$) with block label j, and K is the largest block label among those assigned to "used" hidden states. Equation 3.3 is presented without loss of generality as if used block labels ranged contiguously from 1 to K, since it is possible to permute block labels to whatever configuration is convenient.

3.4.3 Base measure probability masses ($\boldsymbol{\beta}$)

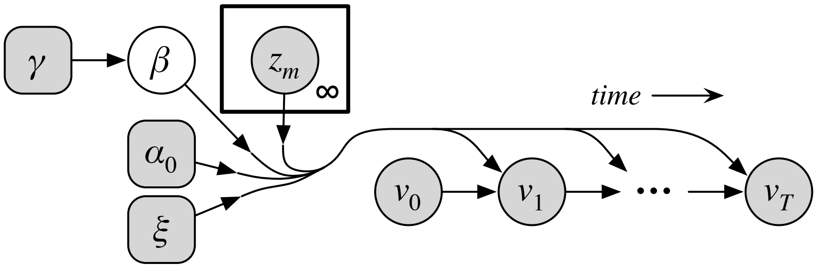
Here we confront a somewhat more challenging inference problem. Before beginning, it may be useful to revisit the bit of mathematical background about the Dirichlet distribution being the conjugate prior of the multinomial distribution (§2.1.1,2.1.2); that is, if c_m counts the number of times the outcome m was sampled from a multinomial distribution with M outcomes, $\text{Mult}(\pi_1, \pi_2, \ldots, \pi_M)$, and if the parameters π_m of this distribution are drawn from a Dirichlet distribution, $\text{Dir}(\alpha_1, \alpha_2, \ldots, \alpha_M)$, then the conditional distribution of the multinomial distribution's parameters is also Dirichlet distributed: $\text{Dir}(\alpha_1 + c_1, \alpha_2 + c_2, \ldots, \alpha_M + c_M)$.

In a straightforward application of Gibbs sampling, where we infer $\boldsymbol{\beta}$ while holding the remaining variables fixed at sampled values, we might depict the problem with the following graphical model:



Here, grayed-out nodes indicate that the corresponding values are "observed", at least in that they are given fixed values that we have drawn during other Gibbs sampling steps. The remaining variables in the model in Figure 3.2 are independent of $\boldsymbol{\beta}$ when conditioning on $\boldsymbol{\pi}$ and are thus omitted here.

As mentioned at the beginning of this section, however, the inference scheme does not explicitly represent $\boldsymbol{\pi}$ but instead marginalizes these quantities out of the model. This alters the "blanket" of variables we must consider when sampling $\boldsymbol{\beta}$ to match the following diagram:



This marginalization realizes the Chinese restaurant process discussed in §2.2.5. Each row m of $\boldsymbol{\pi}$ is replaced by a separate Chinese restaurant, with its base measure specified by the set of discrete atoms $\boldsymbol{\theta}$, the row-specific base measure probability masses, and the concentration parameter α_0. This mechanism implements the marginal probabilities of drawing a particular transition—conditioned on the base measures and on all of the other transitions sampled so far.

Unfortunately, the derivations in §2.2.5 concern circumstances where the base measure is continuous. Since we have a discrete set of atoms paired with probabilities, we need a discrete analog to the Chinese restaurant process for continuous densities. Rather than subject readers to another lengthy derivation very similar to ones seen in the past, we will just state the appropriate conditional probability expression, which, unsurprisingly, happens to implement a Pólya urn scheme. Ignoring the likelihood of any observed data associated with the transition (since we won't need it for $\boldsymbol{\beta}$ inference), the chance of choosing a transition from state m to state n in the BD-IHMM is

$$P(v_t \mid \boldsymbol{v}_{t-1}, \boldsymbol{\beta}, \boldsymbol{z}, \alpha_0, \xi) = \frac{\alpha_0 \beta^*_{mn} + c_{mn,t-1}}{\alpha_0 + c_{m\cdot,t-1}}, \tag{3.4}$$

where $c_{mn,t-1}$ is the count of the number of transitions from m to n in $\boldsymbol{v}_{t-1}$, the complete state trajectory up through time step $t-1$, and $c_{m\cdot,t-1}$ is a shorthand for $\sum_k c_{mk,t-1}$. The wonderful and convenient thing about this expression is that it is not necessary to worry about the order of the transitions in $\boldsymbol{v}_{t-1}$; it is sufficient to know only the number of each type of transition. It turns out that this reasoning allows us to express the probability of an entire hidden state trajectory $\boldsymbol{v}$, again ignoring observation likelihoods, in a very compact form:

$$P(\boldsymbol{v} \mid \boldsymbol{\beta}, \boldsymbol{z}, \alpha_0, \xi) = \prod_m^M \frac{\prod_n^M \dfrac{\Gamma(\alpha_0 \beta^*_{mn} + c_{mn})}{\Gamma(\alpha_0 \beta^*_{mn})}}{\dfrac{\Gamma(\alpha_0 + c_{m\cdot})}{\Gamma(\alpha_0)}}, \tag{3.5}$$

where M is the largest index of all of the states visited in the trajectory $\boldsymbol{v}$, and c_{mn} denotes the total tally of transitions from m to n throughout $\boldsymbol{v}$. We refer back to Equation 3.5 and its components frequently enough that it is worthwhile to illustrate how it comes about. Consider the following sequence of eight transitions:

$$\boldsymbol{v} : 1 \to 2 \to 3 \to 1 \to 2 \to 3 \to 3 \to 1 \to 2.$$

Applying Equation 3.4 for each transition, we get this likelihood for the sequence:

$$\frac{\alpha_0\beta_{12} + c_{12,0} = 0}{\alpha_0 + c_{1\cdot,0} = 0} \cdot \frac{\alpha_0\beta_{23} + c_{23,1} = 0}{\alpha_0 + c_{2\cdot,1} = 0} \cdot \frac{\alpha_0\beta_{31} + c_{31,2} = 0}{\alpha_0 + c_{3\cdot,2} = 0} \cdot \frac{\alpha_0\beta_{12} + c_{12,3} = 1}{\alpha_0 + c_{1\cdot,3} = 1}$$
$$\cdot \frac{\alpha_0\beta_{23} + c_{23,4} = 1}{\alpha_0 + c_{2\cdot,4} = 1} \cdot \frac{\alpha_0\beta_{33} + c_{33,5} = 0}{\alpha_0 + c_{3\cdot,5} = 1} \cdot \frac{\alpha_0\beta_{31} + c_{31,6} = 1}{\alpha_0 + c_{3\cdot,6} = 2} \cdot \frac{\alpha_0\beta_{12} + c_{12,7} = 2}{\alpha_0 + c_{1\cdot,7} = 2}.$$

We can reshuffle the numerator and denominator terms in the likelihood any way we like, and thus we see how the likelihood of the transitions does not depend on the order of the transitions—it is *exchangeable* in the order of the transitions. Consider this ordering, which sorts each of the terms by their c indices:

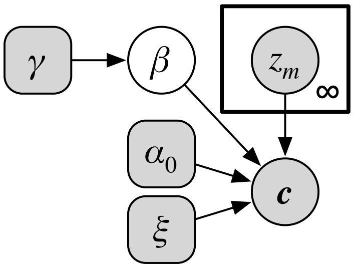
$$\frac{\alpha_0\beta_{12} + c_{12,0} = 0}{\alpha_0 + c_{1\cdot,0} = 0} \cdot \frac{\alpha_0\beta_{12} + c_{12,3} = 1}{\alpha_0 + c_{1\cdot,3} = 1} \cdot \frac{\alpha_0\beta_{12} + c_{12,7} = 2}{\alpha_0 + c_{1\cdot,7} = 2}$$
$$\cdot \frac{\alpha_0\beta_{23} + c_{23,1} = 0}{\alpha_0 + c_{2\cdot,1} = 0} \cdot \frac{\alpha_0\beta_{23} + c_{23,4} = 1}{\alpha_0 + c_{2\cdot,4} = 1}$$
$$\cdot \frac{\alpha_0\beta_{31} + c_{31,2} = 0}{\alpha_0 + c_{3\cdot,2} = 0} \cdot \frac{\alpha_0\beta_{31} + c_{31,6} = 1}{\alpha_0 + c_{3\cdot,5} = 1} \cdot \frac{\alpha_0\beta_{33} + c_{33,5} = 0}{\alpha_0 + c_{3\cdot,6} = 2}.$$

Note that some numerators are paired with different denominators in this expression, but its value is the same as before. If we observe that

$$\prod_{k=0}^{K-1} (k + c) = \frac{\Gamma(K + c)}{\Gamma(c)},$$

it becomes fairly clear how the likelihood of any sequence, through reorderings like the one above, can be expressed in the compact form of Equation 3.5.

To reflect the fact that transition counts are sufficient to characterize the hidden state trajectory as far as $\boldsymbol{\beta}$ inference is concerned, we present our third and final graphical depiction of the problem:



Let us consider the likelihood function of $\boldsymbol{\beta}$ again, using a form which is more verbose than Equation 3.5, explicitly computing the product of the numerators of the Equation 3.4 transition probabilities one at a time:

$$p(\boldsymbol{v} \,|\, \boldsymbol{\beta}, \boldsymbol{z}, \alpha_0, \xi) \propto \prod_{m=1}^{M} \prod_{n=1}^{M} \prod_{k=0}^{c_{mn}-1} (\alpha_0 \beta^*_{mn} + k). \tag{3.6}$$

The normalization constant for this expression, borrowing directly from Equation 3.5, is expressed compactly as $\prod_m^M \Gamma(\alpha_0 + c_{m\cdot})/\Gamma(\alpha_0)$, which incorporates no $\boldsymbol{\beta}^*$ terms. As a likelihood, this expression threatens to be somewhat cumbersome, since a product of sums is somewhat dissimilar to a Dirichlet density, and therefore unlikely to give us a convenient conditional density for $\boldsymbol{\beta}$. To remedy this, we introduce a collection of auxiliary variables $\boldsymbol{s}$ that each take on values in $\{0, 1\}$, and that integrate away to yield the likelihood in Equation 3.6. We have:

$$p(\boldsymbol{v}, \boldsymbol{s} \,|\, \boldsymbol{\beta}, \boldsymbol{z}, \alpha_0, \xi) \propto \prod_{m=1}^{M} \prod_{n=1}^{M} \prod_{k=0}^{c_{mn}-1} (\alpha_0 \beta^*_{mn})^{s_{mnk}} \cdot k^{1-s_{mnk}} \,. \tag{3.7}$$

Drawing these auxiliary variables conditioned on the rest of the information in the likelihood is easy enough:

$$s_{mnk} \,|\, \boldsymbol{v}, \boldsymbol{\beta}, \boldsymbol{z}, \alpha_0, \xi \sim \frac{\alpha_0 \beta^*_{mn}}{\alpha_0 \beta^*_{mn} + k} \,,$$

and note that s_{mn0} for any m, n is always 1. To introduce one final bit of notation, let us refer to the transition-wise sums of the sampled auxiliary variables as $q_{mn} = \sum_{k=0}^{c_{mn}-1} s_{mnk}$.

Where on earth are we going with all of this? Consider that when a particular transition destination is sampled using the marginal transition probability expression in Equation 3.4, the sampler has rolled the dice, spun the roulette wheel, thrown the dart, or engaged in any probabilistic metaphor of your preference and watched a particular outcome come up as a result. The probability mass associated with this outcome, the one described by the equation, can be further portioned into two parts—one due to the base measure mass associated with that transition ($\alpha_0 \beta^*_{mn}$) and one due to the accumulated mass from other times when this transition has been taken ($c_{mn,t-1}$).

When we isolate those times when the sampler "landed on" the first variety of mass, we recover a collection of draws from multinomial distributions parameterized by the proportions $\boldsymbol{\beta}^*_m$. Were a $\boldsymbol{\beta}^*_m$ a Dirichlet distributed multinomial distribution, then with sufficient statistics summarizing these draws, we would be set to draw a new $\boldsymbol{\beta}^*_m$.

Two problems arise from here. The first of these is that we don't keep track of when the sampler "lands on" base measure mass—indeed, drawing hidden state trajectories $\boldsymbol{v}$ has enough bookkeeping already, as readers will discover. We have solved this problem, though, by sampling which of the transitions arose in this way—in effect, guessing. The auxiliary variables s_{mnk} are these guesses, and the counts q_{mn} are sufficient for the $\boldsymbol{\beta}$ conditional density.

The second problem, meanwhile, is the form of this density. Thanks to our base measure modification, the $\boldsymbol{\beta}$ conditional no longer has all of the convenient marginal properties of the Dirichlet process. To see this, compare the $\boldsymbol{\beta}$ likelihood term for the BD-IHMM:

$$\begin{aligned}\prod_m^M \prod_n^M {\beta^*_{mn}}^{q_{mn}} &= \prod_m^M \prod_n^M \beta_n^{q_{mn}} \left(1 + \frac{\xi \cdot \delta(z_m \!=\! z_n)}{\sum_j \beta_j \cdot \delta(z_m \!=\! z_j)}\right)^{q_{mn}} \\ &= \beta_1^{q_{\cdot 1}} \beta_2^{q_{\cdot 2}} \cdot \ldots \cdot \beta_2^{q_{\cdot M}} \prod_m^M \prod_n^M \left(1 + \frac{\xi \cdot \delta(z_m \!=\! z_n)}{\sum_j \beta_j \cdot \delta(z_m \!=\! z_j)}\right)^{q_{mn}},\end{aligned}$$

with the likelihood term that we would be using in the IHMM—in other words, without applying any $\boldsymbol{\beta}$ modification at all:

$$\beta_1^{q_{\cdot 1}} \beta_2^{q_{\cdot 2}} \cdot \ldots \cdot \beta_2^{q_{\cdot M}}.$$

In the case of ordinary Dirichlet process inference, which is what the IHMM uses, the mathematical properties of the Dirichlet process explored in Chapter 2 allow us to use a finite set of proportions to summarize the countably infinite set of proportions $\boldsymbol{\beta}$. At a minimum, this finite set comprises the $\boldsymbol{\beta}$ values corresponding (at least) to the set of hidden states visited by the inferred hidden state trajectory $\boldsymbol{v}$ and one additional value, which we call β_{new}. This value is the sum of all of the remaining proportions in $\boldsymbol{\beta}$; restated,

$$\beta_{\text{new}} = \sum_{m \notin \boldsymbol{v}} \beta_m = 1 - \sum_{m=1}^{M} \beta_m.$$

As demonstrated in §2.2.5, it is possible to think of these proportions in aggregate because the Dirichlet process establishes a Dirichlet density over probability distributions associated with any finite partition of a given measurable space (c.f. §2.2.1). In this case, the measurable space is the space of observation model distribution parameters $\boldsymbol{\theta}$, and the partition that we are interested in comprises the singleton members $\theta_1, \theta_2, \ldots, \theta_M$ and one additional, non-overlapping subset of $\boldsymbol{\theta}$ containing every other $\boldsymbol{\theta}$ value. The probability distribution we associate with this partition is the set of proportions $\beta_1, \beta_2, \ldots, \beta_M, \beta_{\text{new}}$ respectively. The Dirichlet process expresses a prior on these proportions where the parameters for all but the last dimension are infinitesimal:

$$\mathrm{Dir}(\gamma H(\theta_1),\ \gamma H(\theta_2),\ \ldots\ ,\ \gamma H(\theta_M),\ \gamma),$$

but thanks to the form of the likelihood, the conditional distribution of these parameters is a well-formed Dirichlet distribution:

$$Dir(\gamma H(\theta_1),\ \gamma H(\theta_2),\ \ldots\ ,\ \gamma H(\theta_M),\ \gamma)\ \cdot\ \beta_1^{q_{\cdot 1}} \beta_2^{q_{\cdot 2}} \cdot \ldots \cdot \beta_2^{q_{\cdot M}}\ \propto\ \mathrm{Dir}(q_{\cdot 1}, q_{\cdot 2}, \ldots, q_{\cdot M}, \gamma).$$

Applying the same procedure for the BD-IHMM $\boldsymbol{\beta}$ conditional is much less convenient, since the likelihood terms contain infinite sums over all $\boldsymbol{\beta}$ parameters. Fortunately, the approximation proposed in §3.3 encourages us to replace portions of these sums with their expected values. For $\boldsymbol{\beta}$ inference, it is sensible to aggregate the $\boldsymbol{\beta}$ values paired to unvisited states as β_{new}, then use the approximation to portion out β_{new} in the various $\boldsymbol{\beta}$ sums. Thus, in the $\boldsymbol{\beta}$ likelihood shown earlier,

$$\prod_m^M \prod_n^M \left(1 + \frac{\xi \cdot \delta(z_m = z_n)}{\sum_j \beta_j \cdot \delta(z_m = z_j)}\right)^{q_{mn}} \approx \prod_m^M \prod_n^M \left(1 + \frac{\xi \cdot \delta(z_m = z_n)}{\rho_{z_m} \beta_{\text{new}} + \sum_j^M \beta_j \cdot \delta(z_m = z_j)}\right)^{q_{mn}}.$$

With this approximation, we express this conditional density for $\beta_1, \beta_2, \ldots, \beta_M, \beta_{\text{new}}$:

$$p(\beta_1, \beta_2, \ldots, \beta_M, \beta_{\text{new}} \,|\, \boldsymbol{q}, \boldsymbol{z}, \xi) \propto \\ \mathrm{Dir}(q_{\cdot 1}, q_{\cdot 2}, \ldots, q_{\cdot M}, \gamma) \prod_m^M \prod_n^M \left(1 + \frac{\xi \cdot \delta(z_m = z_n)}{\rho_{z_m} \beta_{\text{new}} + \sum_j^M \beta_j \cdot \delta(z_m = z_j)}\right)^{q_{mn}}, \quad (3.8)$$

With some rearrangement, we can restate the density as:

$$p(\beta_1, \beta_2, \ldots, \beta_M, \beta_{\text{new}} \,|\, \boldsymbol{q}, \boldsymbol{z}, \xi) \propto \\ \text{Dir}(q_{\cdot 1}, q_{\cdot 2}, \ldots, q_{\cdot M}, \gamma) \prod_{k}^{K} \left(1 + \frac{\xi \cdot \delta(z_m \!=\! z_n)}{\rho_{z_m} \beta_{\text{new}} + \sum_j^M \beta_j \cdot \delta(z_m \!=\! z_j)} \right)^{q'_k}, \quad (3.9)$$

where K is the largest block label assigned to any of the states along the inferred hidden state trajectory $\boldsymbol{v}$ and q'_k is the sum of all q_{mn} where $z_m = k$ and $z_n = k$.

Although the density approximated by Equations 3.8 and 3.9 is not as simple as the Dirichlet density discussed just prior, we note that important marginal properties are still present. In particular, a derivation very similar to the first proof in §2.1.2 will show that the aggregation property remains, so long as the proportions being aggregated are associated with states within the same block. To restate with an example for slightly more clarity, if $z_1 = z_2$ and we define $G = \beta_1 + \beta_2$, we can show that

$$p(G, \beta_3, \ldots, \beta_M, \beta_{\text{new}} \,|\, \boldsymbol{q}, \boldsymbol{z}, \xi) \propto \\ \text{Dir}(q_{\cdot 1} \!+\! q_{\cdot 2}, q_{\cdot 3}, \ldots, q_{\cdot M}, \gamma) \prod_{k}^{K} \left(1 + \frac{\xi \cdot \delta(z_m \!=\! z_n)}{\rho_{z_m} \beta_{\text{new}} + G \cdot \delta(z_m \!=\! z_1) + \sum_j^M \beta_j \cdot \delta(z_m \!=\! z_j)} \right)^{q'_k},$$

an expression of essentially the same form as Equation 3.9. What this implies is that the approximation strategy of using $\rho_{z_m} \beta_{\text{new}}$ as a stand-in for an infinite sum over $\boldsymbol{\beta}$ values associated with unvisited hidden states does not entail any further assumptions beyond the stipulation in §3.3 that these sums be proportional to the block label probabilities in $\boldsymbol{\rho}$. The within-block aggregation property just cited "allows" us to think of these $\boldsymbol{\beta}$ values in aggregate, and our handy approximation gives us one means of doing so.

So, we have a density. To sample it, we can employ a hierarchical strategy in this case that alternates between sampling the proportions of the $\boldsymbol{\beta}$ values for all the states inside one of the blocks, then sampling the proportions of the sums of $\boldsymbol{\beta}$ values for each of the blocks. This approach is made possible through a variable transformation that re-expresses the $\boldsymbol{\beta}$ density in terms of β_{new} and two conditionally independent sets of variables, $\boldsymbol{G}$ and $\boldsymbol{g}$, where

$$G_k = \sum_j^M \beta_j \cdot \delta(z_j \!=\! k)$$
$$g_m = \frac{\beta_m}{G_{z_m}},$$

expressing the division of the problem just described. We can express the conditional density over these transformed variables as

$$\begin{aligned} p(G_1, G_2, \ldots, G_K, g_1, g_2, \ldots, g_M, \beta_{\text{new}} \,|\, \boldsymbol{q}, \boldsymbol{z}, \xi) \propto& \\ &(G_{z_1} g_1)^{q_{\cdot 1}-1} (G_{z_2} g_2)^{q_{\cdot 2}-1} \cdot \ldots \cdot (G_{z_M} g_M)^{q_{\cdot M}-1} \\ &\cdot G_1^{\sum_j^M \delta(z_j = 1)} G_2^{\sum_j^M \delta(z_j = 2)} \cdot \ldots \cdot G_M^{\sum_j^M \delta(z_j = M)} \\ &\cdot \prod_k^K \left(1 + \frac{\xi}{\rho_{z_m} \beta_{\text{new}} + G_k}\right)^{q'_k}, \end{aligned} \tag{3.10}$$

a somewhat messy expression; the exponents on the third line simply count the number of “used” states with particular block labels. Nevertheless, it is easier to sample. Rather than introduce rigorous new notation for a simple point, let $g_a, g_b, \ldots, g_z$ be a loose shorthand for all of the $\boldsymbol{g}$ values corresponding to states in a single block. Here, $a, b, \ldots, z$ pick out all of the indices of states in the block. The density for these per-block $\boldsymbol{g}$ values is proportional to

$$p(g_a, g_b, \ldots, g_z \,|\, \boldsymbol{q}, \boldsymbol{z}, \xi) \propto \text{Dir}(q_a, q_b, \ldots, q_z). \tag{3.11}$$

Meanwhile, the conditional density for $\boldsymbol{G}$ and β_{new} is somewhat more complex:

$$\begin{aligned} p(G_1, G_2, \ldots, G_K, \beta_{\text{new}} \,|\, \boldsymbol{q}, \boldsymbol{z}, \xi) \propto& \\ &\text{Dir}(q'_1, q'_2, \ldots, q'_K, \gamma) \prod_k^K \left(1 + \frac{\xi}{\rho_{z_m} \beta_{\text{new}} + G_k}\right)^{q'_k}. \end{aligned} \tag{3.12}$$

This density is not as straightforward to sample as, say, a Dirichlet distribution, and though it may then seem as though we have achieved only a marginal improvement on the original density in Equation 3.9, in nearly all BD-IHMM applications we are likely to be much better off under this new alternative parameterization scheme. This is because Equation 3.9 is an M+1-dimensional density, while Equation 3.12 is a K+1-dimensional density. Usually $K \ll M$, making the latter option easier to deal with.

This said, the specifics on how to sample a density like Equation 3.12 will be deferred until §5.3.7, when we present a method powerful enough to handle even densities like the original Equation 3.9. The second density, which will still be faster to sample, may be “plugged into” this method without alteration.

Two closing notes for $\boldsymbol{\beta}$ sampling follow:

¶ **On deriving the variable transformation** — Readers tempted to try deriving the variable transformation out at home are encouraged to do so—with the seemingly obvious caution that they will avoid frustration only if the number of variables they end up with (gs and $\boldsymbol{G}$s) is equal to the number of variables they start out with ($\boldsymbol{\beta}$s). Thus, instead of a g_m corresponding to each β_m in a block, one g_m per block will have to be represented instead with $(1 -$ all of the other g_ms in the block$)$. The proofs in §2.1.2 contain algebraic steps that will be useful for this derivation.

¶ **On the auxiliary variable scheme** — Readers who are familiar with the literature on HDPs will recognize a departure in this section from the usual approach to resampling $\boldsymbol{\beta}$ in so-called "direct assignment" sampling schemes, specifically with regard to computing the auxiliary variable sums q_{mn}. These sums are usually presented and sampled as auxiliary variables themselves from the following density:

$$q_{mn} \mid \boldsymbol{c}, \boldsymbol{\beta}^*, \alpha_0 \;\sim\; \mathrm{St}(c_{mn}, q_{mn})(\alpha_0 \beta^*_{mn})^{q_{mn}} \frac{\Gamma(\alpha_0 \beta^*_{mn})}{\Gamma(\alpha_0 \beta^*_{mn} + c_{mn})}, \tag{3.13}$$

where $\mathrm{St}(a, b)$ is the absolute value of the Stirling number of the first kind for arguments a and b. Stirling numbers are difficult to compute (and even to represent—they become very large very quickly) for even modest values of a and b, and workarounds for this necessity, which typically involve multiple iterations of the Chinese restaurant process, are not particularly inexpensive to compute either. The per-count auxiliary variable scheme presented here is orders of magnitude faster, and reassuringly it can also be shown to be an auxiliary variable method for drawing samples from the density in Equation 3.13. Those interested in this topic are referred to an unpublished technical note by the author that details the derivation of the above expression [94].

3.4.4 Preparing additional "unused" hidden states

The following two inference steps infer a hidden state trajectory $\boldsymbol{v}$ for the observation sequence $\boldsymbol{y}$. In the BD-IHMM, there are a countably infinite number of such states available to explain the data. The BD-IHMM prior ensures that only a limited number of these are actually used, a number that grows and shrinks as the sampler explores the posterior distribution over model parameters. Up until now, to the extent that we have had to think about the infinity of states implied by the model, we have dealt with them through the approximation described in §3.3 and the β_{new} proportion introduced in §3.4.3.

The trajectory inference steps that we will describe require additional consideration of all of the hidden states, since these steps consider how to use the states to describe the observations. To do so, they introduce a second approximation: instead of having all of the states at their disposal, the techniques only consider a finite, albeit arbitrarily large, collection of states. In addition to the ones that are actually present in the inferred trajectory (we occasionally call these "used" states), we supply these inference steps with at least one (and potentially many) additional states that are not initially used to explain data. The inference steps may expand the hidden state trajectory into these states, or conversely decrease the number of states that appear in the trajectory and increase the number of "unused" states. Each iteration of the inference repeats the process of preparing unused states, allowing the number of states in the inferred hidden state trajectory to become arbitrarily large.

This kind of approach to inference in Dirichlet process and related models, where a finite subset stands in for the infinite collection of hidden states (mixture components, etc.) in the model, is said to employ a *truncated* representation, a technique elaborated in [95]. The more hidden states are present in the subset, the greater the number of trajectories (clusterings, etc.) the inference system can hypothesize in a single step becomes. In this way, large truncated representations are more accurate than smaller ones. That said, since the truncation scales itself dynamically as the sampler iterates (in particular, it always ensures that there are some "fresh" states available for expanding the number of states that explain the data), the representation can always grow as large as it needs to be.

At present, the introduction of unused hidden states into our truncated representation, in preparation for the two trajectory inference steps that follow, involves the following three steps:

1. Use the stick-breaking process to sample $\boldsymbol{\beta}$ values for the unused states.
2. Sample observation models for the unused states.
3. Sample block labels for the unused states.

These steps are detailed below. Following this, we speculate on a more unified and rigorous approach to maintaining collections of unused states in our BD-IHMM truncated representation.

The first step establishes $\boldsymbol{\beta}$ values for the unused states. We sample these values by iterating the stick-breaking process once for each time that we wish to instantiate a new unused state. Each iteration consists of the following steps, with M representing the number of instantiated hidden states before the iteration begins:

1. Sample the stick-breaking proportion $\eta \sim \mathrm{Beta}(1, \gamma)$.
2. Break β_{new} to create the new unused state's β proportion, and update $\boldsymbol{\beta}$ accordingly.
   $\beta_{M+1} \leftarrow \eta\beta_{\mathrm{new}}$.
   $\beta_{\mathrm{new}} \leftarrow (1-\eta)\beta_{\mathrm{new}}$.

Unless a "collapsed" approach is in use (c.f. §3.4.1, [93]) the next step is to draw new observation models for the unused states. Lacking data, each of these will be sampled directly from the base measure, H:

$$\theta_m \mid H \sim H.$$

In most applications, these sampled observation models won't be appropriate for any of the data, but we have to have something to stand in for θ_m for the immediate time being. The second trajectory sampling method in §3.4.6 is capable of changing θ_m as it allocates observations to unused hidden states.

The third step assigns a block label to new unused hidden states. This assignment is more involved than simply drawing from the block label probabilities $\boldsymbol{\rho}$: as soon as a state is assigned to a block, then for all states in the same block, the approximated base measure modification term sums change (c.f. Equation 3.1 and §3.3). As these change, so does the probability of the hidden state trajectory (Equation 3.5). To properly sample a block label, we combine this dependent probability—the likelihood of the inferred model parameters given (among other things) this label—with the prior probabilities in $\boldsymbol{\rho}$. We draw from this density:

$$z_m \mid \boldsymbol{v}, \boldsymbol{z}_{\setminus m}, \boldsymbol{\beta}, \boldsymbol{\rho}, \alpha_0, \xi \;\sim\; \rho_{z_m} \cdot P(\boldsymbol{v} \mid \boldsymbol{\beta}, \boldsymbol{z}, \alpha_0, \xi).$$

The number of block labels to consider for z_m is usually fairly small: the few "instantiated" block labels already assigned to the other hidden states, and all other possible labels. The prior probabilities of all unused block labels sum to ρ_{new}, so the conditional probability of drawing any novel label is proportional to $\rho_{\mathrm{new}} \cdot P(\boldsymbol{v} \mid \boldsymbol{\beta}, \boldsymbol{z}, \alpha_0, \xi)$. The selection of this outcome means that a hitherto-unused label should be assigned to z_m, and that the $\boldsymbol{\rho}$ vector should be updated to reflect the instantiation of this new block.

The block labels are not conditionally independent of each other given their prior and the rest of the model parameters: the likelihood term depends on all of $\boldsymbol{z}$. If we are instantiating multiple states, we therefore iteratively resample each block label sampling step for each of these states multiple times. This Gibbs sampling procedure ensures that we are drawing the novel $\boldsymbol{z}$ values from the joint distribution of the block labels for the newly-instantiated states.

¶ **Future approaches** — The procedure just described takes care to ensure that when choosing hidden state block labels for "new" states, the sampling procedure accounts for the effect of this choice on the probability of the inferred hidden state sequence $\boldsymbol{v}$. This care is not extended to the selection of $\boldsymbol{\beta}$ values for these states, whose sizes also affect the base measure modification term sums; instead, these values are drawn directly via the stick-breaking process.

Under many practical circumstances, these effects are unlikely to have a meaningful impact on inference. Since β_{new} typically starts out being rather small even *before* it gets portioned out for new hidden states, the effects on the base measure modification term denominator sums from block label choice or β_{M+1} size will not be of great significance. However, if a more rigorous procedure for introducing unused states is desired, we prescribe (but have not used in practice) the following.

Let M' be the number of "used" states and M be the desired number of total instantiated states, used and unused. Replace the Dirichlet term in the $\boldsymbol{\beta}$ conditional density (Equation 3.9); that is, $\text{Dir}(q_{\cdot 1}, q_{\cdot 2}, \ldots, q_{\cdot M}, \gamma)$, with

$$\left(1 - \sum_{m}^{M'} \beta_m\right)^{\gamma - 1} \prod_{m}^{M'} \beta_m^{q_{\cdot m} - 1} \quad \cdot \quad \frac{\prod_{m=M'+1}^{M} \text{Beta}\left(\frac{\beta_m}{1 - \sum_{n=1}^{m-1} \beta_n}\ ; 1, \gamma\right)}{\prod_{m=M'+1}^{M} (1 - \sum_{n=1}^{m-1} \beta_n)}, \tag{3.14}$$

and employ the resulting density in the $\boldsymbol{\beta}$ sampling procedure described in §3.4.3. This revision expresses a density corresponding to the following generative procedure for sampling a finite set of proportions $\boldsymbol{\beta}$:

1. Draw proportions $\beta_1, \beta_2, \ldots, \beta_{M'}$ from $\text{Dir}(q_{\cdot 1}, q_{\cdot 2}, \ldots, q_{\cdot M}, \gamma)$. Note that the γ parameter accounts for the remainder proportion $1 - \sum_{m=1}^{M'} \beta_m$.

2. Partition this remainder via $M - M'$ iterations of the stick-breaking process.

3. Let $\beta_{\text{new}} = 1 - \sum_{m=1}^{M} \beta_m$.

The two terms of (3.14) reflect the first two steps of this procedure respectively. The denominator in the second term accounts for the variable transformation from the values $\beta_{M'+1}, \ldots, \beta_M$ to the beta-distributed stick break lengths.

Unlike the first proposed method for preparing unused hidden states, the combined conditional distribution over all instantiated $\boldsymbol{\beta}$ values employing the expression in (3.14) *does* account for the likelihood of the hidden state trajectory $\boldsymbol{v}$. It is a more complicated density to sample than the original $\boldsymbol{\beta}$ conditional, and the "block-based" approach in §3.4.3 may have to be adapted or abandoned. Luckily, the inference method described in §5.3.7 is capable of sampling proportional vectors with thousands of dimensions.

Given that the block labels and the $\boldsymbol{\beta}$ values associated with the unused hidden states jointly affect the likelihood of the hidden state sequence $\boldsymbol{v}$, we recommend several iterations of a Gibbs sampling procedure that alternates between resampling both sets of values when instantiating unused hidden states for trajectory inference.

3.4.5 Hidden state trajectory ($\boldsymbol{v}$)

We employ a two-part approach to sampling the hidden state trajectory $\boldsymbol{v}$. Each part addresses a conceptually different aspect of the problem of fitting data to an (in principle) unbounded number of states. The first method seeks mainly to refine the trajectory among the existing set of "used" hidden states, an update that can take advantage of rapid dynamic programming-based methods for HMM inference. The second method seeks to increase or diminish the number of used hidden states to better match the data.

This two-part strategy differs from existing sampling methods for IHMMs, particularly from the original Chinese restaurant process-based approach described in [16] and [1]. The motivations behind this new strategy are speed and flexibility. The CRP method just mentioned samples trajectories one time step at a time—that is, it samples v_t conditioned on $v_1, \ldots, v_{t-1}, v_{t+1}, \ldots, v_T$, the observation y_t, and the rest of the model; then it moves on to sampling v_{t+1}, then v_{t+2}, and so on. Besides being purely time-consuming, the difficulty of this method is that it is particularly susceptible to local optima. If $\boldsymbol{v}$ settles on a particular configuration, when in fact another trajectory with a third of the entries different is considerably more probable, the sampler must change each one of the offending time steps, one by one, until it achieves the more optimal configuration. If some of the intermediate configurations, with, say, half of the revisions made and the other half waiting to be made, are highly improbable, the chance of the sampler even getting to this point along the path to the true optimum is exceedingly low—rather than even venture here, the sampler will more likely revert any initial steps in this direction and scurry back up to the nearby local optimum it started from. The sampler tends to get "trapped" in suboptimal configurations for this reason.

In a sense, the one-by-one approach to sampling $\boldsymbol{v}$ is insufficiently daring. Each change is a small step in the direction of an optimum, a step so small that it is unlikely to cross low-probability "valleys" in the conditional density of $\boldsymbol{v}$. The two methods we use in our hybrid approach to sampling this density take larger, bolder steps, and as such are capable of moving more quickly to and among high-probability regions. These methods are not entirely new by themselves, but the details of their application and their combined use for IHMM inference, to the best of our knowledge, are novel.

Both parts of the inference scheme employ the Metropolis-Hastings algorithm to sample new trajectories. A brief review of this technique is worthwhile before moving on. Metropolis-Hastings (MH) is a Markov chain Monte Carlo method for drawing samples from densities that are otherwise difficult to sample, where "difficult" reflects varying, even extreme degrees of impracticality. Our circumstance is not so bad: in principle, the one-at-a-time Chinese restaurant sampler will get the job done, which is more than can be said for some other MH settings. What it allows us to do here, though, is consider the larger, bolder steps we need for faster inference. These steps are drawn from an easier-to-sample *proposal density* conditioned on the sample from the previous MH iteration; in our case:

$$\boldsymbol{v}' \mid \boldsymbol{v}, \boldsymbol{y}, \boldsymbol{\beta}^*, \boldsymbol{\theta}, \boldsymbol{\alpha}_0 \sim p_{\text{prop}}(\boldsymbol{v}' \mid \boldsymbol{v}, \boldsymbol{y}, \boldsymbol{\beta}^*, \boldsymbol{\theta}, \boldsymbol{\alpha}_0).$$

Subsequently, the proposals are tested against the *target density*: in our case, the actual conditional density over $\boldsymbol{v}$. The test is performed by computing the following ratio (again, adapted to our problem):

$$a = \frac{p_{\text{target}}(\boldsymbol{v}' \mid \boldsymbol{y}, \boldsymbol{\beta}^*, \boldsymbol{\theta}, \boldsymbol{\alpha}_0)\; p_{\text{prop}}(\boldsymbol{v} \;\mid \boldsymbol{v}', \boldsymbol{y}, \boldsymbol{\beta}^*, \boldsymbol{\theta}, \boldsymbol{\alpha}_0)}{p_{\text{target}}(\boldsymbol{v} \;\mid \boldsymbol{y}, \boldsymbol{\beta}^*, \boldsymbol{\theta}, \boldsymbol{\alpha}_0)\; p_{\text{prop}}(\boldsymbol{v}' \mid \boldsymbol{v} \;, \boldsymbol{y}, \boldsymbol{\beta}^*, \boldsymbol{\theta}, \boldsymbol{\alpha}_0)},$$

(where the positioning of $\boldsymbol{v}$ versus the proposed $\boldsymbol{v}'$ is critical) and then by drawing a random number uniformly in $[0, 1]$. If this number is less than a, then we adopt the proposal and assign $\boldsymbol{v}'$ to $\boldsymbol{v}$. Otherwise, we leave $\boldsymbol{v}$ unchanged. Note that some proposals will generate an a value greater than 1, and thus are always accepted. As with any Markov chain Monte Carlo approach, the string of $\boldsymbol{v}$ values sampled in this way, after an initial convergence period, are considered to be samples from the target distribution. Gibbs sampling, the broader framework that characterizes the posterior sampling for the BD-IHMM, can be shown to be a specific case of Metropolis-Hastings where the acceptance ratio a always equals 1.

A proposal density (or an ensemble of proposal densities—in $\boldsymbol{v}$ inference, we have two, with the second described in the following section) used in Metropolis-Hastings must be *admissible*—it must enable the sampler to eventually draw any outcome in the support of the target density in finite time. Aside from this important requirement, practical proposal densities will usually be those whose smallest acceptance ratios a are minimally on the order of 10^{-1} or even 10^{-2}, since they will waste less time generating proposals that are unlikely to be accepted (e.g. $a < 10^{-6}$).

Further information on Metropolis-Hastings appears in [96], among other places. Now, with this review complete, it remains for us only to clearly specify the proposal and target distributions used in this particular mode of $\boldsymbol{v}$ sampling. We begin with the proposal distribution, which is most clearly characterized through its sampling procedure, the goal of which is to allow the application of traditional dynamic programming-based HMM inference techniques to sampling state trajectories in the BD-IHMM. This procedure, in aim and implementation, bears a strong resemblance to that of [97], a technique for rapid IHMM inference. Nevertheless, to deal with the more complex BD-IHMM posterior, it differs from this method in a few important respects. First, our approach does not seek to change the number of hidden states explicitly instantiated in the current sample of the model parameters. Second, our method is employed strictly to make MH proposals—the greater structural simplicity of the IHMM permits the method of [97] to draw true samples from the IHMM posterior. Third, our method is paired with a second, complementary trajectory sampling strategy (§3.4.6); while we expect that this strategy would also speed IHMM inference, the simplicity of that model again makes such efforts unnecessary for most applications.

At any stage of BD-IHMM posterior inference, the subset of base measure probability masses $\boldsymbol{\beta}$ that we actually explicitly represent in our algorithm associate non-infinitesimal mass with a finite collection of states, a collection that includes at least those states visited by the inferred hidden state trajectory, as well as whatever states were added through the procedure described in §3.4.4. Paired with each of these states, which we will call "instantiated" states for now, is a sampled observation model in $\boldsymbol{\theta}$.[2] The proposal distribution, proposes a new trajectory $\boldsymbol{v}'$ through this finite collection of states by using a technique partly resembling the forward-backward algorithm in traditional HMM inference:

[2] This discussion will assume that a collapsed approach is not being used, although altering the methods described in this section for the collapsed case is not difficult.

1. For each pair of instantiated state indices m, n in $1 \ldots M$, with M the number of instantiated states, compute a transition probability π_{mn} as $\frac{\alpha_0 \beta^*_{mn} + c_{mn}}{\alpha_0 + c_{m\cdot}}$. Scale the π_{mn} such that $\sum_{n=1}^{M} \pi_{mn} = 1$.

2. Initialize the first row of the forward probability lattice $\boldsymbol{L}$, a TxM matrix, as

$$\begin{aligned} L_{1n} &= P(v_1 = n \mid v_0, \boldsymbol{\pi})\, p(y_1 \mid v_1 = n, \boldsymbol{\theta}) \\ &= \pi_{v_0 n}\, g(y_1; \theta_n). \end{aligned}$$

3. Fill in remaining rows of $\boldsymbol{L}$, for $t = 2 \ldots T$ in order, as

$$\begin{aligned} L_{tn} &= p(y_t \mid v_t = n, \boldsymbol{\theta}) \sum_{m=1}^{M} L_{t-1m} P(v_t = n \mid v_{t-1} = m, \boldsymbol{\pi}) \\ &= g(y_t; \theta_n) \sum_{m=1}^{M} L_{t-1m} \pi_{mn}. \end{aligned}$$

4. Sample the last entry in $\boldsymbol{v}'$ by drawing in proportion to the last row of L: $v'_T \sim \boldsymbol{L}_T$.

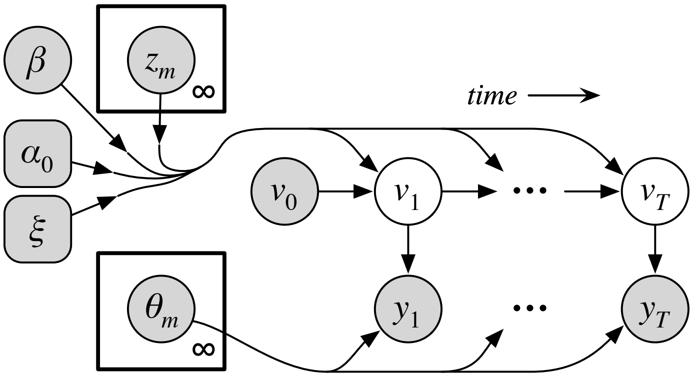
5. Sample the remaining entries in $\boldsymbol{v}'$, for $t = T-1 \ldots 1$ in order, as

$$v'_t \sim \boldsymbol{L}_t \star \boldsymbol{\pi}_{\text{-}v'_{t+1}},$$

   where $\boldsymbol{\pi}_{\text{-}v'_{t+1}}$ denotes the v'_{t+1}th column of $\boldsymbol{\pi}$, and $\star$ is the element-wise (Hadamard) product. In a pseudo-MATLAB notation, one might write $v'_t \sim \boldsymbol{L}(t,:)' \,.\!*\, \boldsymbol{\pi}(:, v'_{t+1})$, which might be somewhat clearer.

Since the first three steps of this procedure are deterministic, computing the probability of a particular proposal $p_{\text{prop}}(\boldsymbol{v}' \mid \boldsymbol{v}, \boldsymbol{y}, \boldsymbol{\beta}^*, \boldsymbol{\theta}, \boldsymbol{\alpha}_0)$ amounts to computing the probability of all of the draws in steps 4 and 5; or, in the case of computing the reverse probabilities $p_{\text{prop}}(\boldsymbol{v} \mid \boldsymbol{v}', \boldsymbol{y}, \boldsymbol{\beta}^*, \boldsymbol{\theta}, \boldsymbol{\alpha}_0)$, the draws the sampler would have needed to make to return to the original $\boldsymbol{v}$ configuration from the sampled proposal. Note that in both cases, the $\boldsymbol{\pi}$ in use is different. Finally, in order to avoid scaling errors, it is necessary to scale $\boldsymbol{L}_T$ and $v'_t \sim \boldsymbol{L}_t \star \boldsymbol{\pi}_{\text{-}v'_{t+1}}$ to sum to 1 when computing the combined probabilities of all of the draws.

Now onto the target density $p_{\text{target}}(\boldsymbol{v}' \mid \boldsymbol{y}, \boldsymbol{\beta}^*, \boldsymbol{\theta}, \boldsymbol{\alpha}_0)$ we mentioned earlier, which, in keeping with previous sections, can be depicted graphically as:

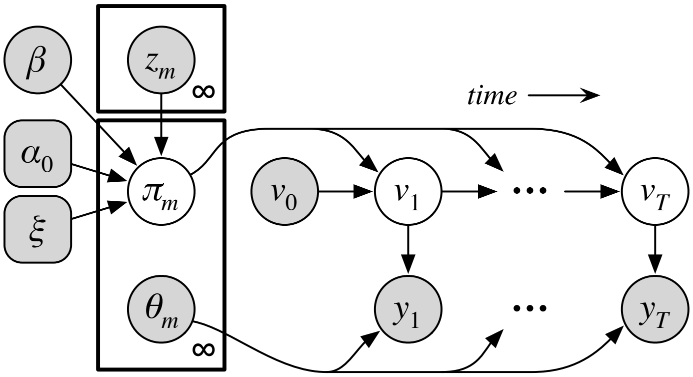


From this, we might try to express our target distribution for $\boldsymbol{v}$ proportionally as

$$\begin{aligned} p_{\text{target}}(\boldsymbol{v}' \,|\, \boldsymbol{y}, \boldsymbol{\beta}^*, \boldsymbol{\theta}, \alpha_0) \;&\propto\; P(\boldsymbol{v}' \,|\, \boldsymbol{\beta}, \boldsymbol{z}, \alpha_0, \xi)\, p(\boldsymbol{y} \,|\, \boldsymbol{v}', \boldsymbol{\theta}) \\ &\propto \prod_m^M \frac{\displaystyle\prod_n^M \frac{\Gamma(\alpha_0 \beta^*_{mn} + c'_{mn})}{\Gamma(\alpha_0 \beta^*_{mn})}}{\displaystyle\frac{\Gamma(\alpha_0 + c'_{m\cdot})}{\Gamma(\alpha_0)}} \prod_t^T p(y_t \,|\, \theta_{v'_t}), \end{aligned}$$

invoking Equation 3.5 for the first term. Indeed, a first version of the sampler did work with this target density; unfortunately, proposed trajectories were rejected nearly every time.

In light of this poor performance, a second strategy was developed. Here, for the only time in the entire BD-IHMM inference procedure, we instantiate the state-to-state transition probabilities $\boldsymbol{\pi}$. These values are used only in this inference step; they are marginalized away everywhere else. In contrast to the previous graphical model, we now confront this inference problem:



Let us begin by assuming that we have a sampled value for $\boldsymbol{\pi}$. The target distribution is now the $\boldsymbol{v}$ trajectory conditioned only on $\boldsymbol{\pi}$ and the observations $\boldsymbol{y}$:

$$\begin{aligned} p_{\text{target}}(\boldsymbol{v}' \,|\, \boldsymbol{y}, \boldsymbol{\pi}, \boldsymbol{\theta}) &\propto P(\boldsymbol{v}' \,|\, \boldsymbol{\pi})\, p(\boldsymbol{y} \,|\, \boldsymbol{v}', \boldsymbol{\theta}) \\ &\propto \prod_t^T \pi_{v'_{t-1} v'_t}\, p(y_t \,|\, \theta_{v'_t}), \text{ or equivalently} \\ &\propto \prod_m^M \prod_n^M \pi_{mn}^{c_{mn}} \prod_t^T p(y_t \,|\, \theta_{v'_t}). \end{aligned}$$

The proposal distribution remains the same as before. Our Metropolis-Hastings acceptance ratio becomes

$$a = \frac{p_{\text{target}}(\boldsymbol{v}' \,|\, \boldsymbol{y}, \boldsymbol{\pi}, \boldsymbol{\theta})\; p_{\text{prop}}(\boldsymbol{v} \;\,|\, \boldsymbol{v}', \boldsymbol{y}, \boldsymbol{\beta}^*, \boldsymbol{\theta}, \boldsymbol{\alpha}_0)}{p_{\text{target}}(\boldsymbol{v} \;\,|\, \boldsymbol{y}, \boldsymbol{\pi}, \boldsymbol{\theta})\; p_{\text{prop}}(\boldsymbol{v}' \,|\, \boldsymbol{v} \;, \boldsymbol{y}, \boldsymbol{\beta}^*, \boldsymbol{\theta}, \boldsymbol{\alpha}_0)}.$$

In applied settings, this new proposal distribution is very frequently accepted.

Now we detail how to instantiate the transition probabilities $\boldsymbol{\pi}$ through sampling. In principle, $\boldsymbol{\pi}$ is a transition matrix with countably infinite rows and countably infinite columns; each row $\boldsymbol{\pi}_m$ lists the probability of transitioning from state m to all of the other hidden states. In keeping with the truncated representation strategy detailed in §3.4.4, we do not try to represent this vector explicitly. Instead, we draw values $\pi_{m1}, \pi_{m2}, \ldots, \pi_{mM}$ corresponding to transitions to all of the instantiated states. We also draw $\pi_{m\,\text{new}}^{[\text{in}]}$, the aggregate probability of transitioning to any uninstantiated state within the same block as state m, and $\pi_{m\,\text{new}}^{[\text{out}]}$, the aggregate probability of transitioning to any uninstantiated state in a different block than state m's. These proportions are Dirichlet distributed:

$$\begin{aligned} &[\pi_{m1}, \pi_{m2}, \ldots, \pi_{mM}, \pi_{m\,\text{new}}^{[\text{in}]}, \pi_{m\,\text{new}}^{[\text{out}]}] \sim \boldsymbol{\pi}_m \,|\, \boldsymbol{v}, \boldsymbol{\beta}, \boldsymbol{z}, \alpha_0, \xi \\ &\qquad\qquad \sim \text{Dir}(\alpha_0\beta^*_{m1}{+}c_{m1},\; \alpha_0\beta^*_{m2}{+}c_{m2},\; \ldots\;,\; \alpha_0\beta^*_{mM}{+}c_{mM},\; \alpha_0\beta^{*[\text{in}]}_{m\,\text{new}},\; \alpha_0\beta^{*[\text{out}]}_{m\,\text{new}}). \end{aligned}$$

The $\beta^{*[\text{in}]}_{m\,\text{new}}$ and $\beta^{*[\text{out}]}_{m\,\text{new}}$ values are new to us. $\beta^{*[\text{in}]}_{m\,\text{new}}$ is the sum of the modified base measure values corresponding to all uninstantiated states in the same block as state m, while $\beta^{*[\text{out}]}_{m\,\text{new}}$ is the sum of the modified base measure values corresponding to uninstantiated states in all blocks besides state m's. As we found when computing the base measure modification term denominator sums in §3.3, we cannot compute these sums exactly, since we don't know which of the countably infinite uninstantiated states take on which label. Nevertheless, we can compute the expectations of these sums. The values we use for $\beta^{*[\text{in}]}_{m\,\text{new}}$ and

$\beta^{*[\text{out}]}_{m\,\text{new}}$ are:

$$\beta^{*[\text{in}]}_{m\,\text{new}} = \frac{\rho_{z_m}\beta_{\text{new}}}{1+\xi}\left(1 + \frac{\xi}{\rho_{z_m}\beta_{\text{new}} + \sum_j^M \beta_j \cdot \delta(z_m = z_j)}\right), \tag{3.15}$$

$$\beta^{*[\text{out}]}_{m\,\text{new}} = \frac{(1-\rho_{z_m})\beta_{\text{new}}}{1+\xi}. \tag{3.16}$$

We now know how to sample $\boldsymbol{v}$ conditioned on $\boldsymbol{\pi}$ and the rest of the model, as well as $\boldsymbol{\pi}$ conditioned on $\boldsymbol{v}$ and the rest of the model. With these, the process of sampling of the hidden state trajectory $\boldsymbol{v}$ is ultimately a Gibbs sampling of the joint $\boldsymbol{\pi}$, $\boldsymbol{v}$ conditional distribution. Sampling alternates between each quantity for a number of iterations, and the final $\boldsymbol{v}$ sample is adopted as the inferred hidden state trajectory.

¶ **On admissibility** — Having now seen all the details of this first mode of $\boldsymbol{v}$ sampling, observant readers might object that our proposal distribution does not permit any of the sequence to be assigned to heretofore unused hidden states. Since the domain of possible $\boldsymbol{v}$ configurations permits a countably infinite number of such assignments to each of the time steps, the Markov chain defined by this single procedure is indeed not admissible. Nevertheless, the pairing of this method and the second approach for drawing $\boldsymbol{v}$ described below, along with other parts of the inference technique described in this section, can draw any possible configuration of hidden state assignments.

Readers who are still feeling slightly uneasy about this arrangement can compare it to any Gibbs sampling approach to drawing samples from some multi-variable joint distribution. Each step draws from the conditional distribution for only one variable, and as such, a single step is not an admissible proposal for the entire joint distribution. In spite of this, all of the sampling steps together work in concert to draw from the joint distribution. For this reason, some sources refer to an entire Gibbs sampling "sweep" through the variables as a single step of the sampler.

3.4.6 Hidden state trajectory, again ($\boldsymbol{v}$)

This second variety of Metropolis-Hastings sampling for the hidden state trajectory $\boldsymbol{v}$ has a complementary purpose to the one presented in the previous subsection. Rather than iterate through the trajectory and determine its best fit to the set of states in the model, this sampling step focuses on the states themselves. It determines whether data currently associated with a single state might be better described if they were associated with two states;

or, conversely, whether data described by two states would be better off associated with a single, combined state. This kind of approach, where the Metropolis-Hastings moves are the splitting and merging of states, is appropriately known as split-merge MCMC, and was originally described for Dirichlet process mixture models in [98]. To our knowledge, we present here the first application of split-merge sampling to HDP-based time series models, using a method that accounts for both the observations and the local dynamic characteristics of the portions of the data sequence under consideration. This said, related approaches using both cues have appeared for finite HMMs, including [99], which restricts itself to splitting states as part of a strategy for handling large amounts of data.

Broadly, the proposal works as follows. At random, it selects two distinct time steps in the sequence. If both time steps are associated with different states, it will propose assigning all of the observations associated with both states to just one of the states—a *merger*. If both time steps are associated with the same state, it will propose moving some of the observations associated with the state to a "new" state, or more precisely, a hitherto unused state—a *split*. In both cases, new observation model parameters are proposed for both states. Because it can shift several time steps between states *en masse*, it has an advantage over the original IHMM one-by-one Chinese restaurant-based sampling approach (c.f. the discussion at the beginning of the last subsection §3.4.5), which must populate new states starting from a single vanguard observation. In any case, because this method can recruit unused states, it can alter $\boldsymbol{\theta}$ and $\boldsymbol{z}$ in addition to $\boldsymbol{v}$.

Some consideration will confirm that the proposal method just sketched could, in principle, yield a Markov chain that achieves any configuration of $\boldsymbol{v}$ assignments. Certain modifications to $\boldsymbol{v}$, however, will be particularly unlikely if only this approach is used. Consider changing only the association of one particular observation y_t from one hidden state to another. The most direct way would split the hidden state indexed by v_t such that only time step t goes into a new state. Next, this state should be merged into the "target" second state. For split-merge methods, this would be a sequence of rather serendipitous and extraordinary events; fortunately, it is just this kind of "fine-tuning" change at which the first variety of $\boldsymbol{v}$ sampler excels.

Once again, we proceed by describing the process of generating a proposal.

1. At random, select two distinct time steps t_a, t_b such that $z_{t_a} = z_{t_b}$. Let $\boldsymbol{u}$ here be the set of indices of states t such that $v_t = v_{t_a}$ or $v_t = v_{t_b}$.

2. Regardless of whether $v_{t_a} = v_{t_b}$, use several iterations of the "two-state-restricted Chinese restaurant process" (described below) to sample an initial partition of the time steps indexed by $\boldsymbol{u}$ into two subsets, with t_a and t_b always in separate subsets.

3. (a) If $v_{t_a} = v_{t_b}$, i.e. if the proposal will split one state into two, use one further iteration of the two-state-restricted CRP to sample a final partition of $\boldsymbol{u}$ into two subsets. Record the probability of the outcome of this iteration in $P_{\text{crp-split}}$.

   (b) If $v_{t_a} \neq v_{t_b}$, i.e. if the proposal will merge two states into one, compute the probability of *undoing* the merger proposal by computing the probability of drawing the original $\boldsymbol{v}$ configuration of the $\boldsymbol{u}$ time steps, using the two-state-restricted CRP, from the initial partition drawn in Step 2. Store in $P_{\text{crp-unmerge}}$.

4. (a) If the proposal effects a split,

      i. Select v_{new}, one of the available unused hidden states, at random in proportion to its $\boldsymbol{\beta}$ value. Flip an evenly-weighted coin to determine which of the partitions computed in Step 3a will have its time steps associated with the original v_{t_a} hidden state and which will have its time steps associated with v_{new}. Let v' be v updated with the corresponding new data labeling.

      ii. Create an updated $\boldsymbol{z}'$ as well, where both states involved in the split have the same block label as the original state, $z'_{v'_{t_a}} = z'_{v'_{t_b}} = z_{v_{t_a}}$.

      iii. Finally, unless a collapsed approach is in use, create an updated $\boldsymbol{\theta}'$ by drawing new observation models $\theta'_{v'_{t_a}}, \theta'_{v'_{t_b}}$ for both states involved in the split.

   (b) On the other hand, if the proposal effects a merge,

      i. Drawing in proportion to $\beta_{v_{t_a}}, \beta_{v_{t_b}}$, sample which of the hidden states v_{t_a} and v_{t_b} will be assigned the all of data associated with both. Let $\boldsymbol{v}'$ be $\boldsymbol{v}$ updated with this new labeling to reflect the merge, and let v'_{new} refer to the hidden state assigned none of the data.

      ii. Create the updated $\boldsymbol{z}'$ by drawing a new block label for v'_{new}: $z'_{v'_{\text{new}}} \sim \boldsymbol{\rho}$.

      iii. Finally, unless a collapsed approach is in use, create an updated $\boldsymbol{\theta}'$ by drawing new observation models $\theta'_{v'_{t_a}}$ and $\theta'_{v'_{\text{new}}}$ for both states involved in the merge.

There is clearly a certain degree of bookkeeping involved in generating proposals, and surely a certain degree of mystery surrounding the "initial partition" and the necessity for computing $P_{\text{crp-unmerge}}$. For the latter issue, the authors of [98] explain that proposing a split (Step 3a) is itself a matter of sampling from a distribution using MCMC, and like all MCMC, the initial starting point whence the sampling Markov chain departs has to come from somewhere. In Step 2, we get it by sampling it—as the initial partition. If doubts linger about this, perhaps one way to feel less uneasy is to imagine taking some time before $\boldsymbol{v}$ inference begins and sampling an initial partition for all possible nonempty subsets of time steps $\boldsymbol{u}$. Then, rather than generating the initial partition in Step 2, we would simply retrieve the appropriate pre-sampled copy. This approach is impractical for elementary reasons, so we sample on-demand instead.

As for $P_{\text{crp}-\text{unmerge}}$, recall that the acceptance ratio for Metropolis-Hastings takes the following general form:

$$a = \frac{P(x \;\,|\, x')\; P(x')}{P(x' \,|\, x\;)\; P(x\;)}.$$

There is only one way to merge two states—it is a deterministic step. Nevertheless, in order to compute this ratio, specifically the $P(x \;\,|\, x')$ term, we need to compute the probability of undoing the merge. This is the probability of splitting the merged state into the original label configuration in $\boldsymbol{v}$, and this is what we store in $P_{\text{crp}-\text{unmerge}}$.

To complete the description of the proposal process, we now detail the two-state-restricted Chinese restaurant process mentioned above. As the name suggests, the two-state-restricted CRP is a Pólya urn-based clustering scheme where the number of colors/clusters is fixed at two. It partitions the collection of time steps $\boldsymbol{u}$ into two non-overlapping subsets. The original randomly-selected time steps t_a and t_b are fixed to be in opposite subsets; the subset assignments for the remaining time steps are resampled iteratively as part of the following (by now vaguely familiar) Gibbs sampling scheme. Referring to subset assignments for the ith time step indexed by u_i as w_i, we draw each w_i (except for those corresponding to t_a and t_b) according to

$$\begin{aligned} w_i \,|\, \boldsymbol{w}_{\setminus i}, \boldsymbol{v} \sim \frac{\sum_j \delta(w_i = w_j)}{|\boldsymbol{w}|} &\cdot p(y_{u_i} \,|\, \boldsymbol{y}_{\{u_j:\, w_i = w_j, i \neq j\}}) \\ &\cdot P(\text{Context}(u_i) \mid \text{Context}(\boldsymbol{u}_{\{j:\, w_i = w_j, i \neq j\}})), \end{aligned} \tag{3.17}$$

where, again, only two outcomes for w_i are possible. Let's unpack this expression into its three constituent terms. The first term expresses the Pólya urn prior on cluster arrangements and is simply a ratio of counts. The second term, $p(y_{u_i} \,|\, \boldsymbol{y}_{\{u_j:\, w_i = w_j, i \neq j\}})$, expresses a collapsed likelihood for observation y_{u_i} given all of the other observations grouped into this partition subset. Strictly speaking, it is not fully necessary to use a collapsed approach here—instead, you might employ an explicit observation model, replacing this term with something like $p(y_{u_i} \,|\, \eta_{w_i})$ and then resampling the two $\boldsymbol{\eta}$ parameters as part of the Gibbs sampling. The use of η, incidentally, underscores the fact that it is not necessary for the observation models in this portion of the inference to be identical to the BD-IHMM observation models parameterized by $\boldsymbol{\theta}$—for the sake of generating proposals, even for computing the integral implicit in $p(y_{u_i} \,|\, \boldsymbol{y}_{\{u_j :\, w_i = w_j, i \neq j\}})$, it may be expedient to use a simpler class of models.

Lastly, there is the $P(\text{Context}(u_i) \mid \text{Context}(\boldsymbol{u}_{\{j :\, w_i = w_j, i \neq j\}}))$, term, a shorthand for a means of comparing the sequences around each of the time steps indexed by $\boldsymbol{u}$, thereby clustering time steps by the shape of the trajectory within their local context. Context(t) is a function that counts the number of times that other states are visited in the vicinity of time step t according to the original trajectory $\boldsymbol{v}$, with "vicinity" some integer parameter chosen *a priori* by the user. If this is set to 3, for example, then Context(6) for the following example sequence

t	1	2	3	4	5	6	7	8	9	10	11	12	13	$\cdots$
v_t	1	3	3	2	2	4	4	4	5	5	1	1	2	$\cdots$

will be the vector $(0, 2, 1, 2, 1)$. This vector will always be of dimension M, the largest index in the sequence $\boldsymbol{v}$. Meanwhile, for multiple-index arguments to Context, as in Context$(\boldsymbol{u}_{\{j :\, w_i = w_j, i \neq j\}})$, the vectors for each individual index are simply added together.

If you've thought to yourself by now that Context seems to return draws from some kind of multinomial distribution, you may be anticipating where we are going with this. Let us indeed assume that within each of the two subsets in the proposal partitioning, the results of Context called on single indices are multinomially distributed, and let us assert an M-dimensional Dirichlet prior on the parameters of this multinomial distribution. For the M parameters of the Dirichlet distribution, let us, for simplicity's sake, use μ/M for each, with μ also chosen *a priori*. Now, briefly, for a more convenient notation,[3] let $\boldsymbol{A} = \text{Context}(u_i)$ and $\boldsymbol{B} = \text{Context}(\boldsymbol{u}_{\{j :\, w_i = w_j, i \neq j\}})$. Then

$$P(\text{Context}(u_i) \mid \text{Context}(\boldsymbol{u}_{\{j :\, w_i = w_j, i \neq j\}})) = \frac{\displaystyle\prod_k \frac{\Gamma(A_k + B_k + \mu/M)}{\Gamma(B_k + \mu/M)}}{\dfrac{\Gamma(\sum_k A_k + \sum_k B_k + \mu)}{\Gamma(\sum_k B_k + \mu)}}, \tag{3.18}$$

which is another compact expression of computing the probability of a series of Pólya urn draws, in the same style as Equation 3.5.

[3] With sincere apologies for the burgeoning alphabet of symbols... *-TSS*

Well, that does it for the two-state-restricted Chinese restaurant process. Now on to the Metropolis-Hastings acceptance ratio. Thanks to the somewhat involved nature of the proposal distribution, the ratio for a split-merge move is the rather ponderous

$$a = \frac{p(\boldsymbol{v}\ ,\boldsymbol{z}\ ,\boldsymbol{\theta}\ \,|\,\boldsymbol{v}',\boldsymbol{z}',\boldsymbol{\theta}',\boldsymbol{y},H)\ p(\boldsymbol{v}',\boldsymbol{z}',\boldsymbol{\theta}'\,|\,\boldsymbol{y},\boldsymbol{\beta},\boldsymbol{\rho},\gamma,\alpha_0,\xi,H)}{p(\boldsymbol{v}',\boldsymbol{z}',\boldsymbol{\theta}'\,|\,\boldsymbol{v}\ ,\boldsymbol{z}\ ,\boldsymbol{\theta}\ ,\boldsymbol{y},H)\ p(\boldsymbol{v}\ ,\boldsymbol{z}\ ,\boldsymbol{\theta}\ \,|\,\boldsymbol{y},\boldsymbol{\beta},\boldsymbol{\rho},\gamma,\alpha_0,\xi,H)}.$$

Let us consider the ratio of proposal probability terms first. For all the complexity of the proposal-making process, several of the steps are deterministic—not only the merging as discussed earlier, but also the $\boldsymbol{z}'$ update in split proposals. Other parts of the proposal, like steps 1 and 2, are shared between splitting and merging, and as such cancel out in the ratio of probabilities. Thus, when merging, the proposal probability terms ratio is

$$\frac{P_{\text{crp}-\text{unmerge}}}{1} \cdot \frac{\beta_{v'_{\text{new}}}}{2(1-\sum_{m\in\boldsymbol{v}'}\beta_m)} \cdot \frac{\beta_{v_{t_a}}+\beta_{v_{t_b}}}{\beta_{v'_{t_a}}} \cdot \frac{1}{\rho_{z'_{v'_{\text{new}}}}} \cdot \frac{p(\theta_{v_{t_a}}\,|\,\boldsymbol{v},\boldsymbol{y},H)\ p(\theta_{v_{t_b}}\,|\,\boldsymbol{v},\boldsymbol{y},H)}{p(\theta'_{v'_{t_a}}\,|\,\boldsymbol{v}',\boldsymbol{y},H)\ p(\theta'_{v'_{\text{new}}}\,|\,H)}, \quad (3.19)$$

and when splitting, the ratio is

$$\frac{1}{P_{\text{crp}-\text{split}}} \cdot \frac{2(1-\sum_{m\in\boldsymbol{v}}\beta_m)}{\beta_{v_{\text{new}}}} \cdot \frac{\beta_{v_{t_a}}}{\beta_{v'_{t_a}}+\beta_{v'_{t_b}}} \cdot \frac{\rho_{z_{v_{\text{new}}}}}{1} \cdot \frac{p(\theta_{v_{t_a}}\,|\,\boldsymbol{v},\boldsymbol{y},H)\ p(\theta_{v_{\text{new}}}\,|\,H)}{p(\theta'_{v'_{t_a}}\,|\,\boldsymbol{v}',\boldsymbol{y},H)\ p(\theta_{v'_{t_b}}\,|\,\boldsymbol{v}',\boldsymbol{y},H)}. \quad (3.20)$$

The five terms of these proposal ratios correspond to the following steps of the proposal outlined previously:

1. $\frac{P_{\text{crp}-\text{unmerge}}}{1}$, $\frac{1}{P_{\text{crp}-\text{split}}}$: Step 3.
2. $\frac{\beta_{v'_{\text{new}}}}{2(1-\sum_{m\in\boldsymbol{v}'}\beta_m)}$, $\frac{2(1-\sum_{m\in\boldsymbol{v}}\beta_m)}{\beta_{v_{\text{new}}}}$: Step 4(a)i.
3. $\frac{\beta_{v_{t_a}}+\beta_{v_{t_b}}}{\beta_{v'_{t_a}}}$, $\frac{\beta_{v_{t_a}}}{\beta_{v'_{t_a}}+\beta_{v'_{t_b}}}$: Step 4(b)i.
4. $\frac{1}{\rho_{z'_{v'_{\text{new}}}}}$ $\frac{\rho_{z_{v_{\text{new}}}}}{1}$: Step 4(b)ii.
5. The large final term: Steps 4(a)iii and 4(b)iii.

Now for the other half of the acceptance ratio. Whether splitting or merging, the dependency structure of the BD-IHMM (Figure 3.2) allows us to factor this ratio as

$$\frac{P(\boldsymbol{z}'\,|\,\boldsymbol{\rho})}{P(\boldsymbol{z}\ \,|\,\boldsymbol{\rho})} \cdot \frac{P(\boldsymbol{v}'\,|\,\boldsymbol{z}',\boldsymbol{\beta},\alpha_0,\xi)}{P(\boldsymbol{v}\ \,|\,\boldsymbol{z}\ ,\boldsymbol{\beta},\alpha_0,\xi)} \cdot \frac{p(\boldsymbol{\theta}'\,|\,H)}{p(\boldsymbol{\theta}\ \,|\,H)} \cdot \frac{p(\boldsymbol{y}\,|\,\boldsymbol{v}',\boldsymbol{\theta}')}{p(\boldsymbol{y}\,|\,\boldsymbol{v}\ ,\boldsymbol{\theta}\ )}.$$

We now consider several of these ratios separately.

¶ — Splitting or merging, nearly all of the $\boldsymbol{z}$ labels are unchanged in $\boldsymbol{z}'$, and their probabilities in the first term cancel accordingly. All that remains in the target density ratio are the probabilities associated with the label that changed when a state was recruited for splitting or discarded in a merge. For splitting, this ratio is

$$\frac{\rho_{z'_{v'_{t_a}}}}{\rho_{z_{v_{\mathrm{new}}}}},$$

while for merging, this ratio is

$$\frac{\rho_{z'_{v'_{\mathrm{new}}}}}{\rho_{z_{v_{t_a}}}}.$$

We would like to apologize at this time for the rather ludicrous fourth-order subscripting.

¶ — Now for $P(\boldsymbol{v}' \mid \boldsymbol{z}', \boldsymbol{\beta}, \alpha_0, \xi) \,/\, P(\boldsymbol{v} \mid \boldsymbol{z}, \boldsymbol{\beta}, \alpha_0, \xi)$. While there are surely faster ways to compute this ratio than plugging numerator and denominator separately into Equation 3.5, we are content to leave this as an exercise.

¶ — The remaining two ratios are fairly straightforward and are omitted here.

Now we can compose the acceptance ratios at last. When merging, the acceptance ratio is

$$a = \frac{P_{\mathrm{crp-unmerge}}}{1} \cdot \frac{\beta_{v'_{\mathrm{new}}}}{2(1 - \sum_{m \in \boldsymbol{v}'} \beta_m)} \cdot \frac{\beta_{v_{t_a}} + \beta_{v_{t_b}}}{\beta_{v'_{t_a}}} \cdot \frac{1}{\rho_{z_{v_{t_a}}}} \cdot \frac{\int p(\boldsymbol{y}_{\{t:\, v'_t = v'_{t_a}\}} \mid \theta)\, p(\theta \mid H)\, d\theta}{\int p(\boldsymbol{y}_{\{t:\, v_t = v_{t_a}\}} \mid \theta)\, p(\theta \mid H)\, d\theta \int p(\boldsymbol{y}_{\{t:\, v_t = v_{t_b}\}} \mid \theta)\, p(\theta \mid H)\, d\theta}, \tag{3.21}$$

and when splitting, the acceptance ratio is

$$a = \frac{1}{P_{\mathrm{crp-split}}} \cdot \frac{2(1 - \sum_{m \in \boldsymbol{v}} \beta_m)}{\beta_{v_{\mathrm{new}}}} \cdot \frac{\beta_{v_{t_a}}}{\beta_{v'_{t_a}} + \beta_{v'_{t_b}}} \cdot \frac{\rho_{z'_{v'_{t_a}}}}{1} \cdot \frac{\int p(\boldsymbol{y}_{\{t:\, v'_t = v'_{t_a}\}} \mid \theta)\, p(\theta \mid H)\, d\theta \int p(\boldsymbol{y}_{\{t:\, v'_t = v'_{t_b}\}} \mid \theta)\, p(\theta \mid H)\, d\theta}{\int p(\boldsymbol{y}_{\{t:\, v_t = v_{t_a}\}} \mid \theta)\, p(\theta \mid H)\, d\theta}. \tag{3.22}$$

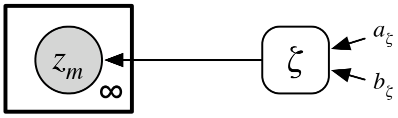
These integrals which have crept in at the last minute are due to the observation model terms in the proposal probabilities (Equations 3.19 and 3.20) not fully canceling with the observation model priors and likelihoods in the target probability ratios—the denominators from Bayes' rule remain. This is a frustrating complication of an otherwise very convenient sampling method, but in our experience the approach is sufficiently beneficial to merit taking the time to devise a means of computing or approximating these integrals.

3.4.7 Hidden state block labels ($\boldsymbol{z}$)

Inference on the hidden state block labels ($\boldsymbol{z}$) is a topic of some complexity. Chapter 4 treats this subject in detail.

3.4.8 Block label concentration hyperparameter (ζ)

Most approaches to sampling concentration hyperparameters for Dirichlet processes marginalize away the sampled proportions (if they were ever represented in the first place) so that they can express their conditional densities in terms of the Chinese restaurant process. Graphically, then, the problem of resampling ζ is represented as



where two parameters for the ζ prior make an appearance, a_ζ and b_ζ. These are the so-called shape and rate parameters for a gamma distribution respectively, using the "second" style of writing the probability distribution function:

$$\mathrm{Gamma}(x; a, b) = \frac{b^a}{\Gamma(a)} e^{-bx} x^{a-1}.$$

Ideally, a_ζ and b_ζ should be selected so that the prior on ζ is vague, or not very informative about ζ at all. Under the parameterization of the gamma distribution written above, a vague prior will have very small values for a_ζ, b_ζ so that its variance, a_ζ / b_ζ^2, is large. Usually, the parameters are also selected so that the prior's mean, a_ζ / b_ζ, is some sensible value.

With the proportions $\boldsymbol{\rho}$ marginalized away, the likelihood function for ζ reflects the probability of drawing M block labels z_m from a generalized Pólya urn, one for each instantiated hidden state, and having come up with K unique block labels in doing so. Through reasoning relatable to the q_{mn} sampling for $\boldsymbol{\beta}$ inference in §3.4.3, and detailed in [1], this likelihood is

$$P(\boldsymbol{z} \mid \zeta) = \mathrm{St}(M, K) \zeta^K \frac{\Gamma(\zeta)}{\Gamma(\zeta + M)}.$$

The resulting conditional can be expressed proportionally as

$$\begin{aligned} p(\zeta \mid \boldsymbol{z}, \alpha_\zeta, \beta_\zeta) &\propto \mathrm{Gamma}(\zeta; \alpha_\zeta, \beta_\zeta)\, \zeta^K \frac{\Gamma(\zeta)}{\Gamma(\zeta + M)} \\ &\propto e^{-b_\zeta \zeta} \zeta^{K + a_\zeta - 1} \frac{\Gamma(\zeta)}{\Gamma(\zeta + M)}. \end{aligned}$$

An auxiliary variable-based sampling scheme is detailed in [1], which notes that

$$\begin{aligned} \frac{\Gamma(\zeta)}{\Gamma(\zeta + M)} &= \frac{1}{\Gamma(M)} \int_0^1 w^{\zeta - 1} (1 - w)^{M-1}\, dw \\ &= \frac{1}{\Gamma(M)} \int_0^1 w^{\zeta} \quad (1 - w)^{M-1}\, dw \left(1 + \frac{M}{\zeta}\right), \end{aligned}$$

the first line taking advantage of the left hand side's near resemblance to the beta function. This leads us to an augmented density over three variables:

$$p(\zeta, w, u \mid \boldsymbol{z}, \alpha_\zeta, \beta_\zeta) \propto e^{-b_\zeta \zeta} \zeta^{K + a_\zeta - 1} \cdot w^\zeta (1 - w)^{M-1} \cdot \left(\frac{M}{\zeta}\right)^u,$$

with w taking a value in $(0, 1)$ and u a value in $\{0, 1\}$. A Gibbs sampling alternates between drawing one of the three variables conditioned on the others. Some manipulation of the above expresses these sampling steps as

$$\begin{aligned} w &\sim \mathrm{Beta}(\zeta + 1, M) \\ u &\sim \mathrm{Bernoulli}\left(\frac{M/\zeta}{1 + M/\zeta}\right) \\ \zeta &\sim \mathrm{Gamma}(a_\zeta + K - u,\ b_\zeta - \log w). \end{aligned}$$

Relatively few Gibbs iterations are necessary for this auxiliary variable-based approach to converge on high probability regions of the ζ conditional density. However, under certain circumstances where K is particularly small, implementors are warned that an unusually tiny ζ sample in one iteration can provoke a numerical instability that traps the sampler to draw small ζ values for all subsequent iterations.

3.4.9 Top-level DP concentration hyperparameter (γ), with a theoretical tangent

In contrast to the situation of the block label concentration hyperparameter ζ, the base measure probability masses $\boldsymbol{\beta}$ cannot be integrated away analytically. This is more than an inconvenience. In ordinary Bayesian inference with Gibbs sampling, we might just try to express a conditional density for γ given $\boldsymbol{\beta}$, its sole descendant in the BD-IHMM graphical model, as in

$$p(\gamma \,|\, \boldsymbol{\beta}) \propto p(\boldsymbol{\beta} \,|\, \gamma) p(\gamma).$$

We cannot do this here: vexingly, the likelihood function $p(\boldsymbol{\beta} \,|\, \gamma)$ is degenerate! What follows is a tangential discussion about why this must be the case and the consequences that it entails—readers interested in just sampling γ already may skip ahead to the paragraph marked **Sampling**.

Recall from the definition in §2.2.1 that a Dirichlet process is a method for assigning probabilities to finite partitions of a measurable space. Let us restate this again algebraically for easy reference. If Θ_1, Θ_2, …, Θ_{J+1} are $J+1$ pairwise disjoint subsets of (in our case) the space of observation model parameters where the $\boldsymbol{\theta}$ dwell, and if H is a probability measure on this space, then the Dirichlet process we impose on the observation model parameter space expresses a probability distribution on probabilities assigned to Θ_1, Θ_2, …, Θ_{J+1} such that

$$[\beta_{\Theta_1}, \beta_{\Theta_2}, \ldots, \beta_{\Theta_{J+1}}] \sim \mathrm{Dir}(\gamma H(\Theta_1), \gamma H(\Theta_2), \ldots, \gamma H(\Theta_{J+1})).$$

The $\boldsymbol{\beta}$ values in our likelihood are each paired with specific singleton $\boldsymbol{\theta}$ parameter values. If the observation model parameter space is continuous and H is too, then we've got trouble. Our likelihood function ought to be

$$p(\boldsymbol{\beta} \,|\, \gamma) \;=\; p(\beta_1, \beta_2, \ldots, \beta_{\mathrm{new}} \,|\, \gamma) \;= \\ \mathrm{Dir}(\gamma H(\theta_1),\ \gamma H(\theta_2),\ \ldots\ ,\ \gamma(1 - H(\theta_1) - H(\theta_2) - \ldots)),$$

but $H(\theta_m)$ will equal 0, giving us $\mathrm{Dir}(0, 0, \ldots, \gamma)$, which is degenerate. Probabilities like $\boldsymbol{\beta}$ that we assign to Dirichlet atoms are unusual: strictly speaking, we can't assign a prior probability to specific instances of $\boldsymbol{\beta}$, but we can say what happens when we marginalize $\boldsymbol{\beta}$ away (we get the Chinese restaurant process, §2.2.5), we can sample finite sets of $\boldsymbol{\beta}$ elements (e.g. stick breaking, §2.2.3), and we can even express a conditional density for them, in a way. This latter item may seem dubious, since our would-be likelihood $p(\boldsymbol{\beta} \,|\, \gamma)$ might also be called a prior for $\boldsymbol{\beta}$, and what is a prior but a posterior for circumstances without data?

We are wandering a bit, but for our own edification, let's clarify what we mean by saying that we can express a conditional density for atom proportions in Dirichlet process mixtures and related models. To be sure, we cannot express probabilities for everything there is to know about an infinite vector of proportions like $\boldsymbol{\beta}$. Instead, we can assign probabilities to two kinds of subsets of the observation model space: swaths of the space that have nonzero measure according to H, and those singleton atoms θ_m associated with mixture components (or, say, HMM states) that we "know about": the ones the inference process has assigned to the data. Usually, we wish to infer $\boldsymbol{\beta}$ values for all of these singletons—we call them β_1, β_2, etc.—and for one swath of the space, the one that contains everything *but* those singletons. Its $\boldsymbol{\beta}$ value is β_{new}. As for the conditional without data—the prior—all it tells us is that in the absence of data, $\beta_{\text{new}} = 1$ with probability 1.

¶ **Sampling** — Back to the BD-IHMM, and back to drawing new values for γ. If we assume that $\boldsymbol{\beta}$ was generated with a Dirichlet process, we also assume that we could draw the proportions in $\boldsymbol{\beta}$ using the stick-breaking process (SBP) (§2.2.3). Can we use the sampling probabilities encountered in the stick-breaking process as a likelihood for $\boldsymbol{\beta}$? We could—if we knew more. First of all, the SBP is order-specific. Although the set of proportions $\boldsymbol{\beta}$ associated with Dirichlet process atoms has no inherent ordering, the order in which elements of this set are sampled by the stick-breaking process is not completely random. Larger fragments of the stick tend to come before smaller ones. Drawing $[.3, .5]$ as the initial two proportions from the SBP will be less likely than drawing $[.5, .3]$.[4] Our $\boldsymbol{\beta}$ vector comes with no SBP ordering information.

The second bit of missing knowledge is related to the first. Just as we have no SBP order for our $\boldsymbol{\beta}$ values, we cannot be certain either that a set of $\boldsymbol{\beta}$ values with M elements actually represents the first M values drawn via an SBP. As mentioned in the digression above, the $\boldsymbol{\beta}$ values we infer are those associated with hidden states our data sequence has actually visited. It could be that the first value produced by an SBP that generated $\boldsymbol{\beta}$ was oddly small, and that its corresponding state was never used. Without accounting for this state somehow, our probability computation for $\boldsymbol{\beta}$ will not be exact.

Unfortunately, we cannot integrate these unknowns away analytically. In that light, we might then try to effect a numerical integration over SBP sequence ordering and missing values by sampling this information via a nested Markov chain Monte Carlo sampler.

[4] Readers verifying statements like these at home can easily deceive themselves into thinking all permutations of a sequence of $\boldsymbol{\beta}$ proportions are equiprobable according to the SBP. A few moments generating such sequences with the SBP will reassure you that this is not the case, since large proportions reliably appear early on in sampled sequences every single time. Recall that to compute the probability of a $\boldsymbol{\beta}$ sequence, you must transform the $\boldsymbol{\beta}$ values into the stick-break *fractions* that were actually sampled. It's easy to forget to factor in the inverse transformation Jacobian determinant when computing the probability of these fractions.

Since γ is a *hyperparameter*, however, we settle for an approximation—after all, if we have such sparse or ambiguous data that small modulations of γ have a profound impact on inference outcomes, we are probably approaching our data analysis task with impure hearts. We use the SBP probability for the γ likelihood, under the assumptions that there are no missing values, and that the $\boldsymbol{\beta}$ values were drawn in descending order (which, if there were no missing values, would be the most likely ordering).

The stick-breaking process probability of a sequence of M beta values (with $\boldsymbol{\beta}_{\text{new}}$ assumed to be $1 - \sum_{i=1}^{M} \beta_m$) is:

$$p(\,[\beta_1, \ldots, \beta_M] \,|\, \gamma) = \frac{\gamma^M (1 - \sum_{m=1}^{M} \beta_m)^{\gamma-1}}{\prod_{n=1}^{M-1} 1 - \sum_{m=1}^{n} \beta_m}.$$

Assuming that the prior for γ is a gamma distribution with parameters a_γ and b_γ, we have

$$\begin{aligned} p(\gamma \,|\, \boldsymbol{\beta}) &\propto \text{Gamma}(a_\gamma, b_\gamma)\; \gamma^M \left(1 - \sum_m^M \beta_m\right)^{\gamma-1} \\ &\propto e^{-b_\gamma \gamma} \gamma^{M+a_\gamma-1} \left(1 - \sum_m^M \beta_m\right)^{\gamma-1} \\ &\propto e^{-\left(b_\gamma + \log(1-\sum_m^M \beta_m)\right)\gamma} \gamma^{M+a_\gamma-1}. \end{aligned}$$

Thus

$$\gamma \,|\, \boldsymbol{\beta}, a_\gamma, b_\gamma \sim \text{Gamma}(a_\gamma + M,\; b_\gamma + \log \beta_{\text{new}})\,.$$

3.4.10 Subordinate DP concentration hyperparameter (α_0)

As with the base measure proportions $\boldsymbol{\beta}$, the likelihood function for the subordinate DP concentration parameter α_0 is the hidden state trajectory probability expressed compactly by Equation 3.5. Starting instead with a normalized version of the more verbose Equation 3.6 (since we must account for the α_0s in the normalization term), we begin by introducing the same $\boldsymbol{s}$ auxiliary variables as we did for $\boldsymbol{\beta}$ inference in §3.4.3:

$$\begin{aligned} p(\boldsymbol{v} \,|\, \boldsymbol{\beta}, \boldsymbol{z}, \alpha_0, \xi) &= \prod_{m=1}^{M} \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_0 + c_{m\cdot})} \prod_{n=1}^{M} \prod_{k=0}^{c_{mn}-1} (\alpha_0 \beta^*_{mn} + k) \\ p(\boldsymbol{v}, \boldsymbol{s} \,|\, \boldsymbol{\beta}, \boldsymbol{z}, \alpha_0, \xi) &= \prod_{m=1}^{M} \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_0 + c_{m\cdot})} \prod_{n=1}^{M} \prod_{k=0}^{c_{mn}-1} (\alpha_0 \beta^*_{mn})^{s_{mnk}} \cdot k^{1-s_{mnk}}. \end{aligned}$$

For a particular configuration of auxiliary variables $\boldsymbol{s}$, the likelihood function for α_0 is

$$\prod_{m=1}^{M} \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_0 + c_{m\cdot})} \prod_{n=1}^{M} (\alpha_0 \beta^*_{mn})^{q_{mn}} \;\propto\; \alpha_0^{q_{\cdot\cdot}} \prod_{m=1}^{M} \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_0 + c_{m\cdot})},$$

where again $q_{mn} = \sum_{k=0}^{c_{mn}-1} s_{mnk}$. Combining the right hand side with a Gamma prior on α_0 parameterized by a_{α_0} and b_{α_0}, we express the unscaled conditional density

$$\begin{aligned} p(\alpha_0 \,|\, \boldsymbol{s}, \boldsymbol{v}, \boldsymbol{\beta}, \boldsymbol{z}, \alpha_0, \xi) &\propto \text{Gamma}(\alpha_0; a_{\alpha_0}, b_{\alpha_0})\; \alpha_0^{q_{\cdot\cdot}} \prod_{m=1}^{M} \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_0 + c_{m\cdot})} \\ &\propto e^{-b_{\alpha_0}\alpha_0} \alpha_0^{q_{\cdot\cdot}+a_{\alpha_0}-1} \prod_{m=1}^{M} \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_0 + c_{m\cdot})}. \end{aligned}$$

From here, we echo the auxiliary variable approach described in [1], a technique similar to the one used for ζ:

$$p(\alpha_0, \boldsymbol{w}, \boldsymbol{u} \,|\, \boldsymbol{s}, \boldsymbol{v}, \boldsymbol{\beta}, \boldsymbol{z}, \alpha_0, \xi) \;\propto\; e^{-b_{\alpha_0}\alpha_0} \alpha_0^{q_{\cdot\cdot}+a_{\alpha_0}-1} \prod_{m=1}^{M} w_m^{\alpha_0} (1 - w_m)^{c_{m\cdot}-1} \cdot \left(\frac{c_{m\cdot}}{\alpha_0}\right)^{u_m},$$

with Gibbs sampling steps

$$\begin{aligned} w_m &\sim \text{Beta}(\alpha_0 + 1, c_{m\cdot}) \\ u_m &\sim \text{Bernoulli}\left(\frac{c_{m\cdot}/\alpha_0}{1 + c_{m\cdot}/\alpha_0}\right) \\ \alpha_0 &\sim \text{Gamma}(a_{\alpha_0} + q_{\cdot\cdot} - \textstyle\sum_m c_{m\cdot},\; b_{\alpha_0} - \sum_m \log w_m). \end{aligned}$$

The warnings about numerical stability issues in ζ inference apply here as well.

3.4.11 Base measure modification hyperparameter (ξ)

The nature of the base measure modification hyperparameter ξ does not suggest avenues for straightforward sampling. Although it is easy to impose a simple gamma prior here too (parameters a_ξ, b_ξ), the likelihood function for ξ is the sequence probability in Equation 3.5. The way in which ξ is “tied up” in this expression (or equivalent forms) makes isolation of a simple-to-sample conditional density elusive. Lacking this, we employ Metropolis-Hastings to draw from

$$p(\xi \,|\, \boldsymbol{v}, \boldsymbol{\beta}, \boldsymbol{z}, \alpha_0) \;\propto\; \text{Gamma}(\xi; a_\xi, b_\xi) \prod_m^M \frac{\displaystyle\prod_n^M \frac{\Gamma(\alpha_0 \beta^*_{mn} + c_{mn})}{\Gamma(\alpha_0 \beta^*_{mn})}}{\displaystyle\frac{\Gamma(\alpha_0 + c_{m\cdot})}{\Gamma(\alpha_0)}}.$$

We recommend the log-normal distribution as a proposal distribution, specifically

$$\begin{aligned} p_{\text{prop}}(\xi' \,|\, \xi, \sigma_\xi) &= \text{Log-Normal}(\xi'; \log \xi, \sigma_\xi) \\ &= \frac{1}{\xi' \sigma_\xi \sqrt{2\pi}} \exp\left[-\frac{(\log \xi' - \log \xi)^2}{2\sigma_\xi^2} \right]. \end{aligned}$$

The proposal ratio term of the Metropolis-Hastings acceptance ratio simplifies conveniently as

$$\frac{p_{\text{prop}}(\xi \;\,|\, \xi', \sigma_\xi)}{p_{\text{prop}}(\xi' \,|\, \xi \;, \sigma_\xi)} = \frac{\xi'}{\xi}.$$

For large problems where computing the likelihood term as expressed in Equation 3.5 becomes inconvenient, it is likely that an auxiliary variable approach identical to the one used for $\boldsymbol{\beta}$ and α_0 inference (c.f. Equation 3.7) can yield a much simpler expression.

3.4.12 Bringing all of the inference components together

Prospective implementors will be pleased to learn that uniting all of these samplers into a complete Gibbs sampler for the BD-IHMM is not terribly difficult. Iterative resampling of model parameters in the order of the subsections above yields the experimental results described later on. Initialization can also be straightforward: assigning each observation to a single state ($v_1 = v_2 = \ldots = v_T = 1$) leads to a succession of hidden state splits from the second variety of $\boldsymbol{v}$ sampling and eventually a sensible allocation of hidden states. Nearly all of the results achieved in Chapter 6 were achieved with samplers initialized in this way. This said, a slightly more sophisticated approach, such as initializing the hidden state trajectory with labels generated by clustering the observation data, can give the BD-IHMM sampler a head start on convergence.

Under certain circumstances, particularly those where the BD-IHMM sampler is initialized to have a very small number of hidden states or blocks (c.f. above), the first few Gibbs iterations can draw small values for the concentration parameters, values that can delay convergence by slowing the creation of new states and blocks. It can be beneficial to skip hyperparameter resampling for several initial Gibbs sampling iterations, allowing the number of states and blocks to increase first.

3.5 In use

Chapter 6 demonstrates several experimental applications of the BD-IHMM, including an artificial data task, a musical theme identification task, and the object model learning task that motivates this thesis.

Chapter 4

Partitioning Sets of Elements Related by Countable Events

The previous chapter describing the Block-Diagonal Infinite Hidden Markov Model (BD-IHMM) lacked a detailed description of how to infer the block labels $\boldsymbol{z}$ conditioned on the rest of the model. Although this omission was partly motivated by the complexity of the inference technique presented in this chapter, inferring the block labels is an interesting problem on its own, one whose consideration yields new insights into how the BD-IHMM works and suggests worthwhile additional adaptations of its probabilistic machinery.

4.1 The task at hand

If we perform Markov chain Monte Carlo (MCMC) inference on the entire BD-IHMM via Gibbs sampling, we eventually reach a step where we must sample the hidden state block labels $\boldsymbol{z}$ conditioned on sampled values for the rest of the variables in the model. Given the dependency structure of the model (characterized graphically in Figure 3.2), and again marginalizing away the Markov transition probabilities $\boldsymbol{\pi}$ as we did in the prior chapter, this sampling will depend only on the values assigned to the hyperparameters α_0, ξ and the parameters $\boldsymbol{\beta}$, $\boldsymbol{\rho}$, and $\boldsymbol{v}$. Moreover, since we have taken pains to achieve a Markov exchangeable model, the order in which we consider transitions between states in the state trajectory $\boldsymbol{v}$ is irrelevant. Instead, just as we only needed to know how many times certain transitions had been taken when sampling a new value for a particular v_t, here, too, we require only the *counts* of how often transitions have taken place between any two different

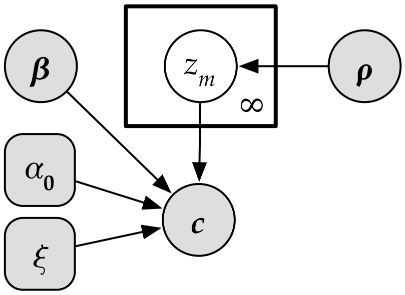


FIGURE 4.1: A graphical model, simplified from the BD-IHMM graphical model in Figure 3.2, showing only the variables relevant to sampling the hidden state block labels $\boldsymbol{z}$ via the methods of this chapter.

states. For a transition from state m to n, we will denote this count c_{mn}, and we will use $\boldsymbol{c}$ to refer to the entire table of counts.

With these considerations in mind, we can present in Figure 4.1 a simplified graphical model containing only the variables relevant to the inference we consider in this chapter. Later on, it will be useful to relate this model to other settings where we can apply this inference.

As with the hidden state trajectory $\boldsymbol{v}$ sampling, the ordinary Chinese Restaurant process approach to inferring hidden state block labels is susceptible to getting stuck in local minima (c.f. discussion at the beginning of §3.4.5). For the hidden state trajectory, the solution was to use sampling methods capable of making multiple changes to $\boldsymbol{v}$ in a single step. We adopt the same kind of strategy for block label inference in this chapter; however, the way in which the block label sampler realizes rapid change differs greatly from the approaches used in trajectory inference.

4.2 Visualizing the block label conditional

To understand the inference technique presented in this chapter, we will deconstruct this proportional expression for the conditional probability of the hidden state block labels:

$$P(\boldsymbol{z} \mid \boldsymbol{\beta}, \boldsymbol{\rho}, \alpha_0, \xi) \propto P(\boldsymbol{z} \mid \boldsymbol{\rho}) P(\boldsymbol{c} \mid \boldsymbol{z}, \boldsymbol{\beta}, \alpha_0, \xi), \tag{4.1}$$

to highlight how its “workings” may be exploited for faster inference. The first term of this expression presents no particular advantage, being simply the prior probability of drawing particular block labels for each state:

$$P(\boldsymbol{z} \mid \boldsymbol{\rho}) = \prod_m \rho_{z_m}.$$

The likelihood term is more complex. With the transition probabilities $\boldsymbol{\pi}$ marginalized away, this term can be thought of as the likelihood of drawing a hidden state trajectory whose transition counts add up to $\boldsymbol{c}$ using the adaptation of the Chinese restaurant process described in the last chapter. Although it is possible to express this likelihood compactly (c.f. Equation 3.5), it is instructive to develop it in a way that makes each step of the process explicit. For this, we need an order in which to consider each of the $\sum_{m,n} c_{mn}$ transitions counted in $\boldsymbol{c}$. The hidden state trajectory $\boldsymbol{v}$ specifies one such order, but any order will yield the same value for the likelihood.

We will clarify this idea with an example. Consider this table of counts of transitions among states 1 and 2:

	1	**2**
1	3	1
2	1	2

and assume that there have been no transitions to or from any other states. Assuming that the hidden state trajectory that produced this table was

$$1 \rightarrow 1 \rightarrow 1 \rightarrow 1 \rightarrow 2 \rightarrow 2 \rightarrow 2 \rightarrow 1,$$

we can picture the transition counts table evolving according to the trajectory like this:

	1	**2**
1	1	0
2	0	0

→

	1	**2**
1	2	0
2	0	0

→

	1	**2**
1	3	0
2	0	0

→

	1	**2**
1	3	1
2	0	0

→

	1	**2**
1	3	1
2	0	1

→

	1	**2**
1	3	1
2	0	2

→

	1	**2**
1	3	1
2	1	2

.

We could just as easily take the trajectory and chop it into its seven individual transitions, then shuffle the transitions into a random order and build the table up that way—regardless of whether this order corresponds to a viable trajectory through the states. The shuffled transition sequence

$$2 \to 2,\ 1 \to 2,\ 1 \to 1,\ 1 \to 1,\ 2 \to 1,\ 2 \to 2,\ 1 \to 1$$

yields this next sequence of counts tables:

$$\begin{array}{c|c|c} & \mathbf{1} & \mathbf{2} \\ \hline \mathbf{1} & 0 & 0 \\ \hline \mathbf{2} & 0 & 1 \end{array} \rightarrow \begin{array}{c|c|c} & \mathbf{1} & \mathbf{2} \\ \hline \mathbf{1} & 0 & 1 \\ \hline \mathbf{2} & 0 & 1 \end{array} \rightarrow \begin{array}{c|c|c} & \mathbf{1} & \mathbf{2} \\ \hline \mathbf{1} & 1 & 1 \\ \hline \mathbf{2} & 0 & 1 \end{array} \rightarrow \begin{array}{c|c|c} & \mathbf{1} & \mathbf{2} \\ \hline \mathbf{1} & 2 & 1 \\ \hline \mathbf{2} & 0 & 1 \end{array} \rightarrow .$$

$$\begin{array}{c|c|c} & \mathbf{1} & \mathbf{2} \\ \hline \mathbf{1} & 2 & 1 \\ \hline \mathbf{2} & 1 & 1 \end{array} \rightarrow \begin{array}{c|c|c} & \mathbf{1} & \mathbf{2} \\ \hline \mathbf{1} & 2 & 1 \\ \hline \mathbf{2} & 1 & 2 \end{array} \rightarrow \begin{array}{c|c|c} & \mathbf{1} & \mathbf{2} \\ \hline \mathbf{1} & 3 & 1 \\ \hline \mathbf{2} & 1 & 2 \end{array}$$

Let us denote by $c_{mn,t}$ the number of transitions from state m to n after we've tallied t transitions into the table according to some order. In the example just above, $c_{11,3} = 1$, while $c_{11,7} = 3$. Let us further use m_t and n_t to refer to the origin and destination respectively of the tth transition in the order. We can now express the likelihood of the transition counts given the hidden state block labels as

$$P(\boldsymbol{c} \mid \boldsymbol{z}, \boldsymbol{\beta}, \alpha_0, \xi) = \prod_{t=1}^{T} \frac{c_{m_t n_t, t-1} + \alpha_0 \frac{1}{1+\xi} \beta_{n_t} \left(1 + \frac{\xi}{\sum_k \beta_k \cdot \delta(z_{m_t} = z_k)}\right)^{\delta(z_{m_t} = z_{n_t})}}{c_{m_t \cdot, t-1} + \alpha_0}, \tag{4.2}$$

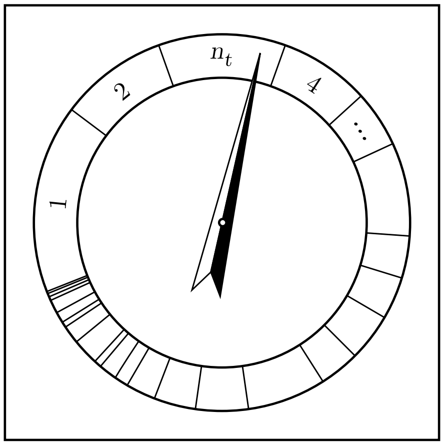
where the · subscript notation refers to a sum over that index, as in the last chapter.

This expression, though unwieldy, is a realization of the Chinese Restaurant Process and can be broken down into intelligible bits. Within the product, the indexed c terms are the same incrementing counts we saw in the last chapter while describing the inference of the hidden state trajectory $\boldsymbol{v}$. The remaining terms are fixed, although the one in the numerator depends on the hidden state block labels, and indeed it is there that the bias promoting transitions between hidden states with the same block label lies. We can rewrite the numerator to highlight the contribution of this bias:

$$c_{m_t n_t, t-1} + \frac{\alpha_0 \beta_{n_t}}{1+\xi} + \delta(z_{m_t} = z_{n_t}) \cdot \frac{\alpha_0 \beta_{n_t}}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_{m_t} = z_k)}, \tag{4.3}$$

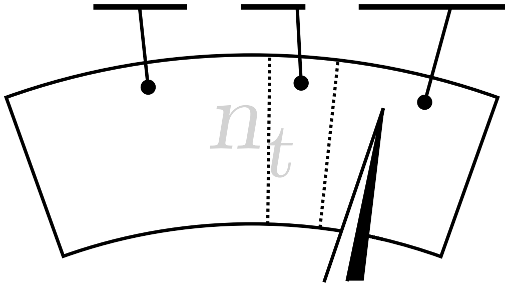
which now rests exclusively in the third term. It is evident here that if the origin and destination states of the transition are in the same block, the delta function in the third term will be nonzero, and the numerator will be larger.

Moving forward, we might ask ourselves: for any given transition, which of the three summed terms in (4.3) was "responsible" for causing the tth transition to have the destination n_t? This is a fanciful idea, but it yields the intuition that will give rise to our accelerated sampling technique. The fraction in Equation 4.2 expresses the probability of the tth transition in our order having drawn destination n_t. We might imagine performing this draw using a spinner like the one that comes with the party game "Twister", except here the face is divided into intervals that each proportionally represent the probabilities of selecting the corresponding destination state:



This time, the pointer has "landed on" the angular interval corresponding to n_t as a destination state. If we look closer, we can see which part of the interval it occupies:

$$c_{m_t n_t, t-1} + \frac{\alpha_0 \beta_{n_t}}{1+\xi} + \delta(z_{m_t} = z_{n_t}) \cdot \frac{\alpha_0 \beta_{n_t}}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_{m_t} = z_k)}$$



specifically, the extra portion of the interval created because states m_t and n_t are in the same block. Hypothetically speaking, if m_t and n_t were not in the same block, this location where the spinner's pointer landed would have belonged to some other destination state, and the transition event we know to have taken place—an event that we are factoring into our likelihood in Equation 4.2—would not have happened at all. This contradiction thus requires that m_t and n_t have the same block label, and also that if the label z_{m_t} changes, then z_{n_t} must change in exactly the same way. Their fates, label-wise, are bound together.

The key to our rapid inference technique is to use a mathematical mechanism relatable to the spinner's pointer to bind together the block labels of many hidden states. When this is done, we can sample a change for all of these labels at once, rather than one at a time.

4.3 Introducing auxiliary variables

Here is a modification of Equation 4.2 incorporating our rewritten numerator (4.3):

$$P(\boldsymbol{c} \mid \boldsymbol{q}, \boldsymbol{z}, \boldsymbol{\beta}, \alpha_0, \xi) = \prod_{t=1}^{T} \text{Step}\left(\frac{c_{m_t n_t, t-1} + \frac{\alpha_0 \beta_{n_t}}{1+\xi} + \delta(z_{m_t} = z_{n_t}) \cdot \frac{\alpha_0 \beta_{n_t}}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_{m_t} = z_k)}}{c_{m_t \cdot, t-1} + \alpha_0} - q_t \right). \quad (4.4)$$

We have introduced auxiliary variables q_t—one for each transition—whose values fall in $[0, 1]$. Additionally, we incorporate the step function Step:

$$\text{Step}(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0. \end{cases}$$

While it may seem peculiar to adopt a new likelihood whose value can be only either 0 or 1, note that integrating the auxiliary variables q_t out of this new expression yields the original likelihood in Equation 4.2. If we substitute it for the likelihood in Equation 4.1 and factor in independent, uniform prior distributions for each of the $\boldsymbol{q}$, we achieve a similarly augmented conditional density for $\boldsymbol{z}$ and $\boldsymbol{q}$ that reverts to the original $\boldsymbol{z}$ conditional when all the $\boldsymbol{q}$ are integrated away. The motivation for this modification is this: if it is possible to sample the joint $\boldsymbol{z}, \boldsymbol{q}$ conditional efficiently, we can discard the $\boldsymbol{q}$ samples and use the $\boldsymbol{z}$ samples for BD-IHMM inference.

In that vein, let us write the new conditional density and consider a Gibbs sampler alternating between updates to $\boldsymbol{q}$ and $\boldsymbol{z}$:

$$P(\boldsymbol{q}, \boldsymbol{z} | \boldsymbol{\beta}, \boldsymbol{\rho}, \alpha_0, \xi) = P(\boldsymbol{q}) P(\boldsymbol{z} \mid \boldsymbol{\rho}) P(\boldsymbol{c} \mid \boldsymbol{q}, \boldsymbol{z}, \boldsymbol{\beta}, \alpha_0, \xi). \quad (4.5)$$

Sampling new values for each q_t is fairly straightforward: since the prior for each is uniform over its $[0, 1]$ range, the conditional density for each q_t is uniform over whatever range keeps the likelihood in Equation 4.4 from falling to 0. Thus, each q_t must be less than or equal to the fractional term in Equation 4.4.

In sampling new values for $\boldsymbol{z}$, we see how the q_t variables take on the role of the spinner's pointer in the prior section. Consider a circumstance where only a single transition between two separate states, 1 and 2, has been counted, reducing the product in Equation 4.4 to a single term. Assume that these states are in the same block, so the third term in the numerator is nonzero. Now assume that our q_t sample for this transition is greater than the sum of the first two numerator terms. If we change the block label for the first state z_1 to something different from z_2, then q_t will be larger than the fraction, and the probability of the labeling will be zero. Evidently z_1 and z_2 must change together or not at all.

There is a second, subtler way in which the auxiliary variables can restrict particular arrangements of block labels—this time by *preventing* states or collections of states from having the same label. For this, picture a new situation where two transitions are observed: one from state 1 to state 2, and another from state 2 to itself. This time, both states have different block labels. Now imagine that the q_t for this second transition happens to be exactly equal to the probability of the transition—metaphorically, the spinner's pointer is resting just at the right edge of the corresponding interval. If we change z_2 to equal z_1, the third term of the numerator will shrink, since the sum of betas for states with this block label increases. The fraction will then be less than q_t, and the probability of the labeling will be zero. In general, very large values of q_t can *restrict* opportunities for joining collections of hidden states together.

These two example circumstances show the effects of auxiliary variables on very small collections of transitions. To understand the auxiliary variable approach on real problems, it's important to keep these summary points in mind:

- Only states with the same block label can bind.
- More than two states can bind together: if states 1 and 2 are bound, and states 2 and 3 are bound, then all three must change their block labels in lockstep. Indeed, changing many states' block labels *en masse* is what makes this sampler efficient.
- A pair of states has more than one opportunity to bind—in fact, it has as many chances as there are transitions between them.

- The process of applying the same block label to large collections of states over multiple iterations is affected by the presence or absence of large sampled q_t values.

This established, we can sketch the process of sampling new values for $\boldsymbol{z}$ given sampled auxiliary variable $\boldsymbol{q}$ values as follows:

1. Identify the sets of states bonded together.
2. For each set of states, draw a new label for all states in the set in proportion to Equation 4.5. For now, it's easiest to stipulate that this conditional must be evaluated for all of the hidden labels, not just the set that changes, although speed-ups exist (and will be described later on). Some candidate labels will have probability 0 due to large sampled q_t values, but it will always be possible to draw the set's original label or a hitherto unused label.

The procedure just sketched is very similar to, and indeed was inspired by, the Swendsen-Wang technique for rapid sampling inference in Ising and Potts models [100, 101]. As here, this method involves using auxiliary variables associated with potential functions to bind discrete label variables (vertices) together so that their values must be resampled simultaneously and identically. It is worth remarking on a particular computational aspect of these approaches before moving on: some papers (e.g. [101]) suggest economizing the time spent sampling auxiliary variables by only sampling those which might expand the set of bound vertices—thus, if vertex pairs 1,2 and 2,3 are bound, there's no advantage in sampling to determine whether vertex 1 should be bound to vertex 2. This time-saving omission is not directly available to the block label inference technique just described, since any transition could yield a q_t that is large enough to restrict certain label changes. We must either sample and deal with each transition's q_t, or account for the transition some other way.

In fact the next section shows that we shall have to learn not to rely entirely on auxiliary variables, since the sampler as described so far turns out to be a rather poor sampler indeed.

4.4 Overbonding and its remedy

Although we've described a mechanism that binds the block labels of hidden states to change together, it happens that this technique is not yet suitable for practical use. To see this, we can compute the probability that none of the counted transitions from state m to n, both in the same block, will yield a bond:

$$P(\text{no bond from any } m \to n \text{ transition}) = \prod_{i=1}^{c_{mn}} \frac{i - 1 + \frac{\alpha_0 \beta_n}{1+\xi}}{i - 1 + \frac{\alpha_0 \beta_n}{1+\xi} + \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}}.$$

To derive this expression, recall that q_t for each transition must be smaller than its corresponding fraction in Equation 4.4, and that moreover for q_t *not* to yield a bond, it must not be greater than the portion of that fractional mass created by the counted transitions and the base measure—the spinner's pointer must land in the first two segments of the interval in the earlier picture. Each term in the product above is the proportion of those two segments in the interval, and thus the probabilities of landing there, as the counts of transitions from m to n increment to their final total. This expression can be computed more efficiently as

$$P(\text{no bond etc.}) = \frac{\Gamma\left(c_{mn} + \frac{\alpha_0 \beta_n}{1+\xi}\right) \Gamma\left(\frac{\alpha_0 \beta_n}{1+\xi} + \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}\right)}{\Gamma\left(c_{mn} + \frac{\alpha_0 \beta_n}{1+\xi} + \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}\right) \Gamma\left(\frac{\alpha_0 \beta_n}{1+\xi}\right)}, \tag{4.6}$$

which contains the same familiar bits of the numerators from (4.2) and (4.4) as the previous expression.

Let's try evaluating this probability with some sensible values plugged into the variables, e.g. 5 for c_{mn}; $\beta_n = 0.02$; $\alpha_0 = 1$; $\xi = 10$; and 0.2 for $\sum_k \beta_k \cdot \delta(z_m = z_k)$. The probability of avoiding a bond is quite low: around 0.016, and this doesn't even count the transitions in the reverse direction (from n to m)! This has dire implications for our auxiliary variable based sampling scheme: nearly every transition between states with the same block label will result in a bond, and it will be extremely unlikely for sets of states to "split off" and adopt their own block label even if the conditional density for z strongly favors that configuration.

Fortunately, we can overcome this problem with more auxiliary variables—or fewer, depending on how you look at it. Consider this new augmented likelihood:

$$
\begin{aligned}
&P(\boldsymbol{c} \mid \boldsymbol{q}, \boldsymbol{r}, \boldsymbol{z}, \boldsymbol{\beta}, \alpha_0, \xi) \;=\\
&\quad\prod_{t=1}^{T} \text{Step}\left(\frac{c_{m_t n_t, t-1} \;+\; \frac{\alpha_0 \beta_{n_t}}{1+\xi} \;+\; \delta(z_{m_t} = z_{n_t}) \cdot \frac{\alpha_0 \beta_{n_t}}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_{m_t} = z_k)}}{c_{m_t \cdot, t-1} + \alpha_0} \;-\; q_t \right)^{r_t}\\
&\qquad\cdot \left(\frac{c_{m_t n_t, t-1} \;+\; \frac{\alpha_0 \beta_{n_t}}{1+\xi} \;+\; \delta(z_{m_t} = z_{n_t}) \cdot \frac{\alpha_0 \beta_{n_t}}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_{m_t} = z_k)}}{c_{m_t \cdot, t-1} + \alpha_0} \right)^{1 - r_t},
\end{aligned}
\tag{4.7}
$$

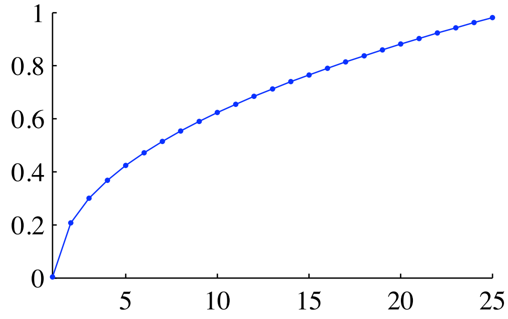
where each r_t is in $\{0, 1\}$, effectively operating as a switch that determines whether transition t is accounted for with a term from the original likelihood (4.2) or from the q_t-augmented likelihood (4.4). Because either term can account for the transition on its own, it is the case that no matter what the prior on the r_t is, integrating r_t and q_t out of an augmented conditional distribution will yield the original block label conditional (4.1). We are therefore free to use whichever kind of term we like for each transition. This freedom allows us to overcome the overbonding phenomenon present in our original sampling scheme by handling some transitions with the ordinary likelihood terms, reducing the number of opportunities for a pair of states to bond.

We can now present a method that uses this bond type selection to ensure that the probability of the transitions from m to n binding their respective states' block labels is as close as possible to, but no greater than, a specified threshold. We begin by specifying the probability of none of the $m \to n$ transitions *from the τth onward through the c_{mn}th transition* yielding a bond:

$$
\begin{aligned}
&P(\text{no bond from any } m \to n \text{ transition from the } \tau\text{th on})\\
&\qquad = \prod_{i=\tau}^{c_{mn}} \frac{i - 1 + \frac{\alpha_0 \beta_n}{1+\xi}}{i - 1 + \frac{\alpha_0 \beta_n}{1+\xi} + \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}}\\
&\qquad = \frac{\Gamma\left(c_{mn} + \frac{\alpha_0 \beta_n}{1+\xi}\right) \Gamma\left(\frac{\alpha_0 \beta_n}{1+\xi} + \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)} + \tau - 1\right)}{\Gamma\left(c_{mn} + \frac{\alpha_0 \beta_n}{1+\xi} + \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}\right) \Gamma\left(\frac{\alpha_0 \beta_n}{1+\xi} + \tau - 1\right)}\\
&\qquad = P(\text{Bond}_{mn} \mid \tau) \text{ for short.}
\end{aligned}
\tag{4.8}
$$

As τ increases, this probability does too, since there are fewer transitions that might yield a bond, and also because the likelihood of any individual transition bonding diminishes as the incrementing count i grows.

Using mostly the same "sensible values" from before, except setting $\alpha = 5$ and $c_{mn} = 25$ for greater visual effect, we can plot the probability of avoiding a bond as a function of τ:



.

For this set of $m \to n$ transitions, we can see that if we use unaugmented likelihood terms for the first fifteen transitions, then augmented likelihood terms from the 16th onward, we have a slightly greater than 20% chance of sampling a bond between m and n (ignoring the transitions in the other direction for now).

The reader is invited to privately reassure themselves that $P(\text{Bond}_{mn} \mid \tau)$ increases monotonically through the valid range of τ, then to consider the function $\text{Bp}_{mn}(x)$, a piecewise-linear interpolation of this probability on the real numbers. This new function is not meaningful as a probability; we introduce it for notational convenience. More specifically,

$$\text{Bp}_{mn}(x) = \begin{cases} x \cdot P(\text{Bond}_{mn} \mid \tau = 1) & 0 \leq x < 1 \\ (x - \lfloor x \rfloor) \cdot P(\text{Bond}_{mn} \mid \tau = \lfloor x \rfloor) \\ \quad + (\lceil x \rceil - x) \cdot P(\text{Bond}_{mn} \mid \tau = \lceil x \rceil) & 1 \leq x \leq c_{mn} \\ (x - c_{mn}) \cdot P(\text{Bond}_{mn} \mid \tau = c_{mn}) \\ \quad + (c_{mn} + 1 - x) \cdot 1 & c_{mn} \leq x \leq c_{mn} + 1, \end{cases} \tag{4.9}$$

where the function is bookended by two additional linear segments that draw its value down to 0 at $x = 0$ and up to 1 at $x = c_{mn} + 1$. These segments ensure that the function's inverse Bp_{mn}^{-1} is defined on the entire interval $[0, 1]$.

With this, let $r_{mn,i}$ be the r_t auxiliary variable associated with the ith transition from state m to n. For fixed m and n, we can use the following procedure to choose values for all

$r_{mn,i}$ such that the probability of bonding labels z_m and z_n through the $m \to n$ transitions is exactly some fixed value p, or as close as possible without exceeding it:

1. Let $x := \mathrm{Bp}_{mn}^{-1}(1-p)$.
2. Select the appropriate outcome:
    - If $0 \leq x < 1$, then the probability of bonding on $m \to n$ can never exceed $P(\mathrm{Bond}_{mn} \,|\, \tau = 1)$, which is less than p. Set all $r_{mn,i} = 1$ to make the bond as probable as possible.
    - If $1 \leq x \leq c_{mn} + 1$, then for any $i < \lfloor x \rfloor$, set $r_{mn,i} = 0$. Draw $r_{mn,\lfloor x \rfloor} \sim \mathrm{Bernoulli}(\lceil x \rceil - x)$, then set any remaining $r_{mn,i} = 1$.

The Bernoulli sample in the second condition above determines whether to use an augmented likelihood term or a regular likelihood term to account for the $\lfloor x \rfloor$th transition. The probability of bonding on $m \to n$ in these cases are higher than p and lower than p respectively, and by randomly choosing which option to take in proportion to $\lceil x \rceil - x$, the ultimate probability of bonding is exactly p.

There are other ways to manipulate the probability of bonding on a set of transitions from one state to another. The probability of bonding on only the very first transition is $\left(\frac{\alpha_0\beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}\right) \Big/ \left(\frac{\alpha_0\beta_n}{1+\xi} + \frac{\alpha_0\beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}\right) \approx 0.98$ in the plotted example, so in that case one might achieve a 20% probability of bonding by drawing $q_{mn,1} \sim \mathrm{Bernoulli}(0.2/0.98)$ and setting all remaining $q_{mn,i} = 0$. This second strategy differs from the one outlined earlier in that it would use augmented likelihood terms for the first or first few $m \to n$ transitions rather than the last several. It is purely speculative for the time being; although future work might investigate whether it confers any advantage or disadvantage in $\mathbf{z}$ inference, in what follows, we describe an algorithm that employs the first method exclusively. Either way, it is worth noting that a method of portioning transitions between normal likelihood and augmented likelihood terms will require far less bookkeeping later on if, as the count increments from 1 to c_{mn}, it only switches once between the two types of terms.

4.5 The sampler

The title of this section is a deliberate deception to trick the reader into enduring one further elaboration of a bookkeeping convenience, this one yielding a worthwhile reduction in storage requirements. We will arrive at the algorithm before long. Meanwhile, eager readers who have jumped here directly to avoid pages of tiresome exposition are admonished to return to the beginning of this chapter in a spirit of contrition and forbearance.

Earlier, we worked out how larger q_t auxiliary variable values can actually prevent sets of bound states from changing their block labels in certain ways. Prospective implementers might therefore believe that it is necessary to store many of these values, one for each augmented term in the likelihood, in order to check them against potential label changes. In fact, for each set, all but one of the sampled q_t values may be discarded once the sets themselves have been determined. The one q_t to save is the one that will invalidate a candidate label change for the set before any of the others in the set do, since it only takes one zero-valued augmented likelihood term to collapse the likelihood.

To determine which q_t in a set has this distinction, we define

$$q_t^* = \frac{q_t(c_{m_t\cdot,t-1} + \alpha_0) - c_{m_t n_t,t-1} - \frac{\alpha_0 \beta_{n_t}}{1+\xi}}{\frac{\alpha_0 \beta_{n_t}}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_{m_t} = z_k)}}. \tag{4.10}$$

For q_t that do not bind two states' block labels together, q_t^* will be negative and can be discarded. Binding q_t have q_t^* values that range between 0 and 1, representing how far clockwise the spinner pointer extends into the interval corresponding to same-label transition enhancement. Conveniently, thanks to this proportional representation, q_t^* variables from all transitions can be compared against each other, in spite of the differing counts and β values associated with each.

Referring back to (4.3), consider that to recompute this numerator for a new arrangement of block labels (call them $\boldsymbol{z}'$), we resize the third term by multiplying it with $\frac{\sum_k \beta_k \cdot \delta(z_{m_t} = z_k)}{\sum_k \beta_k \cdot \delta(z'_{m_t} = z'_k)}$. Thus, to test a q_t^* against a new arrangement of block labels, we need only check that $q_t^* \leq \frac{\sum_k \beta_k \cdot \delta(z_{m_t} = z_k)}{\sum_k \beta_k \cdot \delta(z'_{m_t} = z'_k)}$. Several q_t^* in the set could fail this test for a given configuration of $\boldsymbol{z}'$, but the one that will always fail if any of them fail is the largest q_t^*—the fraction, if it is less than zero, will sink below this threshold first. Thus it is this q_t^* that we must save.

At last we are prepared to present an algorithmic characterization of the hidden state block label sampler, which appears on the following pages. The pseudocode there contains a few worthwhile optimizations based on methods in the preceding discussion, but further speedups are possible.

Variables

$\boldsymbol{c}, \boldsymbol{z}, \boldsymbol{\beta}, \boldsymbol{\rho}, \alpha_0, \xi, \zeta, \boldsymbol{r}, p$	As characterized in the prior discussion.
i, j, k, m, n, τ	Iterators and temporaries.
M	Largest state index; see remarks below.
a_p, b_p	Bond probability distribution parameters.
s_m	Label indicating which set of bound states contains state m.
$q_{mn,i}, q^*_{mn,i}$	Related to q_t and q^*_t; see remarks below.
$q^*_{s_m}$	Largest $q^*_{mn,i}$ among state pairs with set label s_m.
$P, P_{\text{base}}, P_j, P^*_j$	Conditional $\boldsymbol{z}$ labeling probabilities and related intermediate quantities.

Algorithm

```
Repeat until convergence:

1. Initialize set labels $s_m$.
   $s_m \leftarrow m$ for $m$ in $1 : M$.

2. Draw bond probability for this iteration.
   $p \sim \text{Beta}(a_p, b_p)$.

3. Draw auxiliary variables for pairs of states.
   for each ordered pair $m, n$ such that $c_{mn} > 0$, in random order,

   (a) Don't bother bonding states that are already in the same set. Instead, use ordinary likelihood terms to account for $m \rightarrow n$ transitions.
       if $s_m = s_n$ then
           $r_{mn,i} \leftarrow 0$ for $i$ in $1 : c_{mn}$.
           continue to the next ordered pair of states.
   (b) Set all $r_{mn,i}$ as described in §4.4 and determined by $p$. Let $\tau$ be the smallest $i$ such that $r_{mn,i} > 0$. (Pseudocode omitted.)
   (c) Draw necessary $q_{mn,i}$ auxiliary variables for this state pair.
       for $i$ in $\tau : c_{mn}$,
           i.  $q_{mn,i} \sim \text{Uniform}\left(0,\ \ i-1+\frac{\alpha_0\beta_n}{1+\xi}+\delta(z_m=z_n)\cdot\frac{\alpha_0\beta_n}{1+\xi}\cdot\frac{\xi}{\sum_k \beta_k\cdot\delta(z_m=z_k)}\right)$.
           ii. Handle bonding if it has occurred: assign sets connected by this bond the same label and identify the strongest bond in the new, unified set.
               if $q_{mn,i} \ > \ i-1+\frac{\alpha_0\beta_n}{1+\xi}$ then
```

```
            **if** $s_m \neq s_n$ **then**
                $q^*_{s_m} \leftarrow \max(q^*_{s_m}, q^*_{s_n})$.
                $s_j \leftarrow s_m$ **for each** $j$ such that $s_j = s_n$.
            $q^*_{mn,i} \leftarrow \frac{q_{mn,i} - (i-1) - \frac{\alpha_0 \beta_n}{1+\xi}}{\frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}}$.
            $q^*_{s_m} \leftarrow \max(q^*_{s_m}, q^*_{mn,i})$.

4. Determine the unnormalized augmented conditional probability $P$ of the current $\boldsymbol{z}$.

    (a) Factor in the label prior first.
        $P \leftarrow \prod_{m=1}^{M} \rho_{z_m}$.
    (b) Factor in the unnormalized, non-augmented likelihood terms.
        **for each** ordered pair $m, n$ such that $c_{mn} > 0$,
            $\tau \leftarrow$ the smallest $i$ such that $r_{mn,i} > 0$.
            $P \leftarrow P \cdot \frac{\Gamma\left(\tau - 1 + \frac{\alpha_0 \beta_n}{1+\xi} + \delta(z_m = z_n) \cdot \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}\right)}{\Gamma\left(\frac{\alpha_0 \beta_n}{1+\xi} + \delta(z_m = z_n) \cdot \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}\right)}$.

5. For each set of bound states, sample new block labels.
   **for each** $m$ such that at least one set label $s_j = m$, in random order,

    (a) Prepare to compute the probabilities of candidate block labels by factoring out certain contributions of states bearing the block label assigned to states in set $m$. This specifically includes transitions within these states, but not transitions to these states, most of which will not be affected by changing the block label of the set.
        $k \leftarrow$ the block label assigned to states in set $m$.
        $P_{\text{base}} \leftarrow P \Big/ \rho_k^{\sum_{m=1}^{M} \delta(z_m = k)}$.
        **for each** ordered pair $m, n$ such that $c_{mn} > 0$ **and** $z_m = z_n = k$,
            $\tau \leftarrow$ the smallest $i$ such that $r_{mn,i} > 0$.
            $P_{\text{base}} \leftarrow P_{\text{base}} \Bigg/ \frac{\Gamma\left(\tau - 1 + \frac{\alpha_0 \beta_n}{1+\xi} + \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}\right)}{\Gamma\left(\frac{\alpha_0 \beta_n}{1+\xi} + \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}\right)}$.
    (b) Compute the conditional probability $P_1, P_2, \ldots$ of each candidate block label $1, 2, \ldots$ for the states in this set.
        **for** $j$ **in** $1$ : the dimensionality of $\boldsymbol{\rho}$

        i. **if** $j = k$ **then**
               $P_j \leftarrow P$.
               **continue** to the next candidate block label.
        ii. Determine whether $q^*_{s_m}$ is large enough to prevent hidden states in set $m$ from adopting block label $j$.
            if $q^*_{s_m} > \frac{\sum_i \beta_i \cdot \delta(z_i = j)}{\sum_i \beta_i \cdot \max(\delta(z_i = j), \delta(s_i = m))}$ then
                $P_j = 0$.
                continue to the next candidate block label.
        iii. Factor out contributions to the conditional probability from states that currently have the block label $j$, particularly transitions within these states and transitions to and from these states from states with block label $k$.
            if $z_n = j$ for any $n$ then
                $P_{\text{base}} \leftarrow P \Big/ \rho_j^{\sum_{m=1}^{M} \delta(z_m = j)}$.
            for each ordered pair $m, n$ such that $c_{mn} > 0$ and ($z_m = j$ or $z_n = j$)
                $\tau \leftarrow$ the smallest $i$ such that $r_{mn,i} > 0$.
                $P_j \leftarrow P_j \Bigg/ \frac{\Gamma\left(\tau - 1 + \frac{\alpha_0 \beta_n}{1+\xi} + \delta(z_m = z_n) \cdot \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}\right)}{\Gamma\left(\frac{\alpha_0 \beta_n}{1+\xi} + \delta(z_m = z_n) \cdot \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}\right)}$.
        iv. Construct a new set of block labels $\boldsymbol{z}'$ reflecting the change of set $m$'s state's block labels from $k$ to $j$.
        v. Factor in contributions to the conditional, particularly transitions into, out of, and between states that now have block labels $j$ and $k$.
            $P_j \leftarrow P_{\text{base}} \cdot \rho_j^{\sum_{m=1}^{M} \delta(z'_m = j)}$.
            if $z'_n = j$ for any $n$ then
                $P_j \leftarrow P_{\text{base}} \cdot \rho_k^{\sum_{m=1}^{M} \delta(z'_m = k)}$.
            for each ordered pair $m, n$ such that $c_{mn} > 0$ and ($z_m = k$ or $z_m = j$ or $z_n = k$ or $z_n = j$)
                $\tau \leftarrow$ the smallest $i$ such that $r_{mn,i} > 0$.
                $P_j \leftarrow P_j \cdot \frac{\Gamma\left(\tau - 1 + \frac{\alpha_0 \beta_n}{1+\xi} + \delta(z'_m = z'_n) \cdot \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z'_m = z'_k)}\right)}{\Gamma\left(\frac{\alpha_0 \beta_n}{1+\xi} + \delta(z'_m = z'_n) \cdot \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z'_m = z'_k)}\right)}$.
    (c) Scale $P_j$ to sum to 1 and draw a new block label for all hidden states in set $m$.
        $P^*_j \leftarrow \eta P_j$ for $j$ in $1$ : the dimensionality of $\boldsymbol{\rho}$ with $\eta$ such that $\sum_j P^*_j = 1$.
        $k \sim \text{Multinomial}(P^*_1, P^*_2, \ldots)$.
    (d) Update block labels $\boldsymbol{z}$ such that all hidden states in set $m$ have label $k$.
    (e) $P \leftarrow P_k$.
6. Sample new block label proportions $\boldsymbol{\rho}$; see remarks below.
```

Some remarks on the algorithm:

¶ **Notation** — To limit the number of variables in use, the deeply nested sum $\sum_k \beta_k \cdot \delta(z_m = z_k)$ seen throughout the algorithm reuses the variable k as an index, even though k also appears elsewhere as temporary storage for label assignments. Please consider the k that appear within the sums as distinct from the label assignment k: if you like, imagine that they exist within a separate "scope."

¶ **Bonding strategy** — The discussion on overbonding in §4.4 described one way in which the number of transitions accounted for via augmented likelihood terms was attenuated—this to avoid the otherwise likely circumstance of all states with the same block label binding together. The algorithm above introduces a second strategy for avoiding augmented likelihood terms, for a different purpose. In step 3a, if two states are already bound into the same set, all of the transitions between them are accounted for via non-augmented likelihood terms. This way, the sampler runs a lower risk of drawing a large q auxiliary variable value for that transition, which in turn might inhibit otherwise worthwhile label changes. Any practical dependency this strategy imposes on the order in which the algorithm compares pairs of states is mitigated by the random ordering employed at each sampling iteration.

Notably, this approach of only sampling auxiliary variables that might change the configuration of the sets of bound states *is* the "economical" approach that Section 4.3 indicated that we could not actually do. Now, however, since we can use the non-augmented likelihood terms to account for transitions between whatever state pairs we like, we don't have to worry about sampling auxiliary variables that we don't want to sample.

¶ **Sampled bonding probabilities** p — Each iteration of the sampler draws a new maximum probability p of bonding same-labeled states together from a beta distribution parameterized by a_p, b_p. Large and small p values engender different behaviors from the sampler. A small p usually yields few bonds and thus many small sets of bound hidden states, while the many bonds caused by large p often lead to a small number of large sets. These configurations in turn result in changes to the arrangement of block labels that resemble localized, minor edits and large-scale, multi-state label swaps respectively.

At the time of writing, the optimal policy for adjusting p, if it exists, has not been investigated. Intuition suggests that a large p would be most appropriate during early iterations, allowing the sampler to make large steps toward a labeling coarsely resembling the optimum. In subsequent iterations, smaller p values for more careful refinement of the labels seem most useful. For the time being, we hedge our bets by sampling p to use a range of values at different stages of the inference process. In terms of the sampler making "bad choices", there appears not to be much risk in this strategy, since small p-derived changes early on seem unlikely to deflect subsequent large steps toward high probability label configurations, and very large sets from large p values later on in the iteration will seldom switch their labels from a highly probable arrangement to an improbable one.

¶ **Counts of states:** M — In the pseudocode, M is assumed to represent both the largest hidden state index in the hidden state trajectory $\boldsymbol{v}$ and the number of unique hidden state indices in $\boldsymbol{v}$. This cannot be the case if, in $\boldsymbol{v}$, state 5 is never visited but all the other states from 1 to 10 are. It's certainly possible for other parts of the inference code to sample such a trajectory, though, and an easy solution is to permute the indices of visited states and blocks such that they form a contiguous block starting from 1.

¶ **Approximation of infinite sums** — The familiar base measure augmentation term denominator sum $\sum_k \beta_k \cdot \delta(z_m = z_k)$ appears in many places throughout this chapter. Although this representation of the sum reflects the theoretical formulation of the model, most implementations will approximate this sum over a countably infinite number of $\boldsymbol{\beta}$ values. Invoking the "necessary approximation" discussed in §3.3, we recommend implementors replace all instances of $\sum_k \beta_k \cdot \delta(z_m = z_k)$ in this chapter with $\rho_{z_m}\beta_{\text{new}} + \sum_k^M \beta_k \cdot \delta(z_m = z_k)$.

¶ **Truncated approximation of $\boldsymbol{\rho}$** — The block label sampling scheme in this chapter does not actually employ a countably infinite prior distribution $\boldsymbol{\rho}$ over individual block labels but a finite, truncated approximation of this distribution. This approximation has minimal impact on the outcome of sampling for two reasons: first, most BD-IHMM applications will have a relatively small number of sub-behaviors; second, in nearly all cases, a relatively small number of the entries in $\boldsymbol{\rho}$ account for almost all of its probability mass.

To draw a new truncated set of $\boldsymbol{\rho}$ values conditioned on the rest of the model parameters, as directed in Step 6 of the pseudocode, let K be the largest block label in $\boldsymbol{z}$ and do the following:

1. Draw $\rho_1, \rho_2, \ldots, \rho_K, \rho_{\text{new}} \sim \text{Dir}(\sum_{m=1}^M \delta(z_m = 1),\ \sum_{m=1}^M \delta(z_m = 2),\ \ldots\ , \sum_{m=1}^M \delta(z_m = K),\ ,\ \zeta)$.
2. Subdivide ρ_{new} according to the stick-breaking process to generate $\boldsymbol{\rho}$ values corresponding to new block label candidates.

The fidelity of the approximation increases with the number of "extra" ρ_{new} entries are generated via the stick-breaking process.

¶ **Differences in q auxiliary variables** — The bond threshold auxiliary variables in the sampler differ in two ways from the q_t variables discussed prior to the pseudocode. First, instead of being indexed by time step t, they are indexed by source and destination state indexed pairs, then the incrementing transition count i between those states, as in $q_{mn,i}$ and $q^*_{mn,i}$. This is a fairly innocuous change on its own, and if we wanted to, we could make a mapping from these mn, i style subscripts to time steps—we might use an ordering that stepped through columns n in rows m of c, incrementing through counts c_{mn} at each location, more example.

More subtly, when drawing a $q_{mn,i}$, we do not use the actual probability of that particular transition given the ordering just described—or any one specific ordering, for that matter—as an upper bound on the uniform sample. Instead, in step 3(c)i, we draw from an interval ranging from 0 to the numerator of this probability. We could compute the denominator dictated by some ordering in order to draw from the proper interval, but since it would be canceled out anyway in the computation of $q^*_{mn,i}$, there's not much reason to bother. The algorithm can do everything it needs to do with the $q_{mn,i}$ as sampled without normalization. This said, $q^*_{mn,i}$ and $q^*_{s_m}$ *are* computed to specification—there is no peculiar scaling of these values.

¶ **Logarithmic number system** — The unscaled conditional probabilities and related intermediate values in the sampler $(P, P_{\text{base}}, P_j)$ commonly become too small for accurate numeric representation via ordinary IEEE 754 double precision floating point values—or even too large, since the missing normalization terms in their computation can be enormous themselves. A common trick under either of these circumstances is to store and perform arithmetic on the logarithms of these probabilities instead, a scheme broadly referred to as a logarithmic number system. Since none of the manipulations of the conditional in the algorithm require addition or subtraction, this technique is fairly straightforward, particularly with the use of common numerical library functions that compute the logarithm of the gamma function directly.

One slightly tricky step under this scheme is 5c, which can be accomplished by adding some offset to the logarithmic representations of the P_j values, enough so that the largest P_j is easily captured by floating point representations when all of these values are converted back into real numbers. These may then be renormalized to yield the P^*_j for sampling, which is why the initial scaling represented by the additive offset to the log values is inconsequential. Some P_j values besides the largest may still be minuscule after the log-to-real conversion, but the chances of the sampler selecting these is so small that scrupulously accurate representation is not very important.

4.6 Further enhancements

The development of the sampling technique described in this chapter was aided by a computer program that generates artificial counts matrices for varying values of the $\boldsymbol{\beta}$, α_0, ξ, and $\boldsymbol{\rho}$ parameters. Improvements to the sampler have been marked by successful partitions of counts matrices generated with smaller and smaller ξ parameters, and also with fewer and fewer transitions tallied in the matrices—in other words, with less data. While we successfully applied the sampler described in the previous section (§4.5) to a challenging collection of artificial matrices, we were able handle even harder problems after making some additional enhancements.

4.6.1 An elaboration on bond probability scheduling

Perhaps the most difficult kind of situation for the sampler to overcome is one in which all hidden states start out with the same block label, and the evidence for the differently-configured ground truth labeling, while tangible, is rather weak. In these circumstances, almost every pair of states with transitions between them can be bonded with probability p, the transition probability selected at Step 2 of the sampling algorithm. This may not seem like a bad thing—after all, p is supposed to modulate bond sampling. In fact, p is an *upper bound* on the probability of binding two states together, not a prescription. The sampler as presented operates best when the overall average probability of binding together two states that ought not to be bound (that is, two states that should be in separate blocks) is usually somewhat smaller than the average probability of binding together two states that do belong in the same block.

In difficult cases, the difference between these two probabilities for any pair of states is slight. As a result, p is the main determiner of whether states will bond. A large p value means that nearly all states bond, and their labels either all change together or not at all. In either case, the state clustering result is equivalent, and not useful. A small p value means that very few states bond, and their labels either change one by one or not at all. The latter case is far more likely, since such small changes are unlikely to allow the sampler to escape local optima within the space of block label configurations (c.f. also the beginning of §3.4.5 for commentary on a similar situation in hidden state trajectory inference).

To mitigate this phenomenon, we introduce a dynamic schedule for p and auxiliary variable sampling designed to bond states together in intermediate-sized clumps. These clumps are intended to occupy a "happy medium" between the two conditions described above: they are small enough that changing the labels of their constituent states will represent a meaningful modification to the labeling, and large enough that such changes have a real chance of extending beyond the confines of local optima. The clumping works by "growing" sets of bound states from individual, randomly-selected states. States in these sets attempt to bind to neighbor states according to a decaying p parameter; when they cease to bind, a new set is started and the procedure repeated until all of the elements have been accounted for. We express this procedure in a replacement for steps 2 and 3 in the pseudocode of §4.5:

New variables

$a_{p\,\text{start}}, b_{p\,\text{start}}$	Parameters of the initial starting bond probability PDF.
$a_{p\,\text{decay1}}, b_{p\,\text{decay1}}$	Parameters of the starting bond probability decay rate PDF.
$a_{p\,\text{decay2}}, b_{p\,\text{decay2}}$	Parameters of the per-state bond probability decay rate PDF.
U	Set of all states awaiting bonding.
$w_{\text{open}}, w_{\text{outbound}}$	Queues for growing bound state sets.
p_g, p_l	Bond probabilities.
λ_g, λ_l	Bond probability decay rates.

Algorithm modification

```
2. Initialize set $U$ of states awaiting bonding.
   $U \leftarrow \{1 : M\}$

3. Grow sets of bonded states.
   **while** $U$ is unempty,

   (a) Prepare a "seed state" to begin growing a new set of bonded states.
       $m \leftarrow$ a random selection from $U$.
       $s_m \leftarrow$ a new bound state set label.
       $q^*_{s_m} \leftarrow 0$.
       $U \leftarrow U$ without state $m$.
       Enqueue $m$ into $w_{\text{open}}$.
   (b) Draw the initial starting bond probability for this set.
       $p_g \leftarrow \text{Beta}(a_{p\,\text{start}}, b_{p\,\text{start}})$.
   (c) Draw the set-wide bond probability decay rate.
       $\lambda_g \leftarrow \text{Beta}(a_{p\,\text{decay1}}, b_{p\,\text{decay1}})$.
   (d) Grow the set from the initial seed state.
       **while** $w_{\text{open}}$ is unempty,

       i. Get the next state in the set's state queue and retrieve its outbound neighbors. Sort these by transition count and establish the queue of bond targets to examine.
          $m \leftarrow$ dequeue next item from $w_{\text{open}}$.
          $w_{\text{outbound}} \leftarrow$ all $n$ such that $c_{mn} > 0$, in descending order of $c_{mn}$.
       ii. Prepare the per-state bond probability.
          $p_l \leftarrow p_g$.
```

```
iii. Draw auxiliary variables for pairs of states; if states do bond, add the target
     state to the collection of states to examine.
     while $w_{\text{outbound}}$ is unempty,
     A. Get the next state in the bond target queue, but don't bother bonding
        states in the same set, or states that are no longer awaiting bonding.
        Instead, use ordinary likelihood terms to account for the $m \leftarrow n$ transitions.
        $n \leftarrow$ dequeue next item from $w_{\text{outbound}}$.
        if $s_m = s_n$ or $n \notin U$ then
            $r_{mn,i} \leftarrow 0$ for $i$ in $1 : c_{mn}$.
            continue to the next item in $w_{\text{outbound}}$.
     B. Set all $r_{mn,i}$ as described in §4.4 and determined by $p$. Let $\tau$ be the smallest
        $i$ such that $r_{mn,i} > 0$. (Pseudocode omitted.)
     C. Draw necessary $q_{mn,i}$ auxiliary variables for this state pair. Handle bonding
        if it occurs: maintain the strongest bond in the set and the collection
        of states to examine.
        for $i$ in $\tau : c_{mn}$,
            $q_{mn,i} \sim \text{Uniform}\left(0, \quad i-1 + \frac{\alpha_0\beta_n}{1+\xi} + \delta(z_m = z_n) \cdot \frac{\alpha_0\beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}\right)$.
            if $q_{mn,i} \ > \ i-1 + \frac{\alpha_0\beta_n}{1+\xi}$ then
                $q^*_{mn,i} \leftarrow \frac{q_{mn,i} - (i-1) - \frac{\alpha_0\beta_n}{1+\xi}}{\frac{\alpha_0\beta_n}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_m = z_k)}}$.
                $q^*_{s_m} \leftarrow \max(q^*_{s_m}, q^*_{mn,i})$.
                Enqueue $n$ into $w_{\text{open}}$ if $n \notin w_{\text{open}}$.
                $U \leftarrow U$ without state $n$.
     D. Decay the per-state bond probability.
        $p_l \leftarrow \lambda_l p_l$.
iv. Decay the starting bond probability.
    $p_g \leftarrow \lambda_g p_g$.
```

4.6.2 Initialization and annealing

The preceding bond scheduling technique greatly reduces the tendency of the sampler to get stuck in certain pathological block label configurations, particularly the configuration that assigns all hidden states to the same block. Nevertheless, in circumstances where the small ξ/low count conditions mentioned in the previous section are exacerbated even further, it can still take hundreds of iterations for the sampler to break free of these configurations. These hundreds of iterations don't take terribly long—the algorithm in this chapter, for all its complexity, can be implemented to run relatively quickly—but faster convergence is always preferable.

We confront these adverse circumstances with two final tricks up our sleeve. The first is to simply initialize the sampler with each state assigned to a different block. For most counts matrices encountered in practical use, this is a highly improbable block label configuration, and the sampler is unlikely to dwell there. It has the additional benefit of being on the other "side" of a good $\mathbf{z}$ configuration—presumably comprising a limited number of moderately-sized blocks—from the single-label arrangement, and so the sampler is more likely to converge on a set of good hidden state arrangements well before it risks being bogged down in sticky terrain.

This "liberating" approach to initialization is formalized somewhat in our second trick for initializing the sampler, which is inspired by the stochastic optimization technique known as *simulated annealing* [102]. The mechanism of this procedure is to temporarily inject extra randomness into the step in which the sampler draws new labels for bound hidden state sets (step 5c in the pseudocode of §4.5).

When choosing a new block label for a set of bound hidden states, the sampler computes a list of probabilities for each possible block label outcome, including one outcome corresponding to the creation of a new block label. We inject additional randomness by adding some quantity κ to each outcome probability and renormalizing the probabilities before sampling. If κ is large, the distribution of block label choices is essentially uniform, and the sampler effects a random walk over the space of block labelings. When necessary, we apply this technique in practice by starting κ with a large value and causing it to decay with each sampling iteration, eventually setting it to 0. The scrambled labelings that come with large κ values closely match the one-state-per-block initialization just described; meanwhile, the gradual decay, comparable to the "temperature" schedule in simulated annealing, allows the sampler to maintain a degree of flexibility as it settles into higher-probability regions of the block label configuration space. Once this annealing regime is complete, the newly-initialized sampler is iterated multiple times as usual to generate an authentic draw from the block label conditional distribution.

4.7 In use

As hinted in the introduction to this chapter, the sampling method just described has uses beyond the inference of hidden state block labels $\boldsymbol{z}$ in the BD-IHMM. Consider that the sufficient statistics for the distribution over labels amount to an assortment of Greek-lettered parameters and, most critically, a collection of counts $\boldsymbol{c}$ (c.f. Figure 4.1). Although in the BD-IHMM $\boldsymbol{c}$ happens to count the number of inferred transitions between the hidden states, there is no requirement for $\boldsymbol{c}$ to have this or any other interpretation. Just as hidden states are entities where pairs can have a certain kind of discrete, directional, countable relationship between each other—the number of times a time series has transitioned from the first state to the second—other sorts of entities can have numerically identical relationships with different interpretations. Section §6.5 demonstrates our sampling technique applied to a problem with this same structure.

Chapter 5

Beyond Blocks: General Structure

The Block-Diagonal Infinite Hidden Markov Model (BD-IHMM) described in the previous chapters is an elaboration on the Hierarchical Dirichlet Process (HDP) where sampled proportions—used to describe the hidden state transition dynamics—exhibit additional structure. Each hidden state is assigned to one of a countably infinite collection of blocks, and the transition dynamics are generated in such a way that transitions between states in the same block *usually* have an enhanced probability relative to transitions between states in different blocks (c.f. Figure 3.1).[1]

This sort of block structure is a useful kind of modification to the HDP, as the experiments in Chapter 6 hopefully attest. Nevertheless, further varieties of modification are possible, and indeed are necessary for our original goal of using perceived appearance similarity and experienced patterns of visual change to create models of visual objects. The BD-IHMM, as noted in the first chapter, considers only the latter half of this pairing. For some objects, this may not be a great liability: we can pick up a fork, for example, and turn it in all directions to see how it looks from different angles. We cannot do this with many other objects, however—cars, for example, are difficult to manipulate freely by hand—and yet we are able to recognize them from novel perspectives, as a child might do the first time she sees a car from inside a tall building. We can even make guesses about how these different viewpoints relate to one another: observed visual similarities or inferred geometric relationships might make the child expect the car's appearance to change in a

[1] The "usually" qualification reflects the flexibility of the model—transition probabilities are permitted to vary and may not always exhibit this behavior, though in nearly all worthwhile applications of the model, they are very likely to do so.

particular way under dramatic rotations, even if she has never seen any of the *Die Hard* movies.

Within the context of visual object models, the BD-IHMM prior captures the notion that our visual experiences of objects exhibit persistence—objects tend to occupy our attention for contiguous spans of time. To meet our original goals, we look now for an enhanced prior that encodes the following additional stipulations discussed in the first chapter:

- Different views of the same visual object tend to share some common appearance characteristics (global traits).
- Views that are similar in appearance are more likely to have transitions between them than views that are dissimilar (smoothness).

In this chapter we present a framework for constructing such a prior through a more general structural modification of HDP-based hidden Markov models.

5.1 A general framework

Consider the equation from the BD-IHMM that modifies the base measure mass for the transition between states m and n (Equation 3.1, reproduced here for convenience and slightly modified to fit the subsequent discussion):

$$\beta^*_{mn} = \frac{\beta_n}{1+\xi}\left(1 + \frac{\xi\delta(z_m = z_n)}{\sum_k \beta_k \cdot \delta(z_m = z_k)}\right).$$

The multiplicative augmentation of the base measure term is switched on or off depending on the block labels $\boldsymbol{z}$, and its magnitude is always the same: $\xi \,/\, \sum_k \beta_k \cdot \delta(z_m = z_k)$. Suppose we replaced this augmentation with a different scheme where the magnitude of the augmentation is allowed to assume any non-negative value. Since we now need some basis for choosing this amount of augmentation, let us say that it now depends on the values of the base measure atoms $\boldsymbol{\theta}$. We now write

$$\beta^*_{mn} = \frac{\beta_n}{1+\xi}\left(1 + \frac{\xi f(\theta_n;\theta_m)}{\sum_k \beta_k \cdot f(\theta_k;\theta_m)}\right), \tag{5.1}$$

with f a non-negative real function of θ_m, θ_n. There is no requirement for f to be symmetric (i.e. $f(\theta_n;\theta_m) = f(\theta_m;\theta_n)$) or have a finite integral, though $f(\theta_n;\theta_m)$ must itself be finite for all pairs of atoms θ_m, θ_n. As with the BD-IHMM, the $\sum_k \beta_k \cdot f(\theta_k; \theta_m)$ and $1 + \xi$ terms ensure that the modified $\boldsymbol{\beta}^*$ values sum to 1.

Forgetting about block diagonality or multiple objects for now, let us imagine modeling a single object with an infinite hidden Markov model whose base measure is modified according to Equation 5.1. Each state m is an object view whose appearance characteristics are encoded in the corresponding observation model parameters θ_m. Suppose now that f is some simple measure of similarity between pairs of encoded appearance characteristics. In the event where θ_m and θ_n describe similar-looking views, the corresponding augmentations will be larger than the case where θ_m and θ_n look nothing alike, a pattern that will tend to be reflected in the respective transition probabilities π_{mn} and π_{nm}. This behavior enables our new modification to endow our prior with the second structural characteristic listed above—smoothness.

5.1.1 Re-expressing the BD-IHMM

Equation 5.1 is rather general and admits for modifications based on phenomena other than similarity. In fact, this approach subsumes all realizations of the BD-IHMM, which becomes evident when we think about that model slightly differently than we have before. Previously, we described the base measure for the top-level Dirichlet process as a distribution over observation model parameters $\boldsymbol{\theta}$. The hidden state block labels $\boldsymbol{z}$ were sampled separately from an infinite-outcome categorical distribution parameterized by proportions $\boldsymbol{\rho}$, which were generated via the stick-breaking process. An equivalent formulation, however, would have both θ_m, z_m for a hidden state m within a single atom sampled from a joint base measure over observation model parameters and hidden state block labels. Let us refer to these pairs with the symbol ω, as in $\omega_m = \{\theta_m, z_m\}$. For the BD-IHMM, we say

$$\begin{aligned} \omega_m = \{\theta_m, z_m\} \,|\, H &\sim H \\ &\sim H_\theta H_z \\ &\sim H_\theta \cdot \boldsymbol{\rho}\,, \end{aligned}$$

the key aspect being that this new joint base measure H factors into two independent components H_θ and H_z, for observation model parameters and block labels respectively.

The last step in expressing the BD-IHMM under the new framework is specifying f. It is, quite simply

$$f(\omega_n; \omega_m) = \delta(z_m = z_n).$$

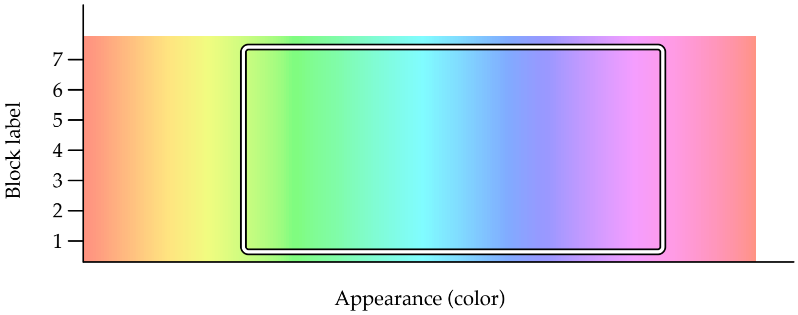


FIGURE 5.1: A cartoon depiction of a base measure for object model views where appearance characteristics θ_m—object color, in this case—and block labels z_m are independent. The rectangle indicates the one high probability region. Knowing the block label of a view gives no additional information about the kind of appearance it is likely to have; compare with the dependent case in Figure 5.2.

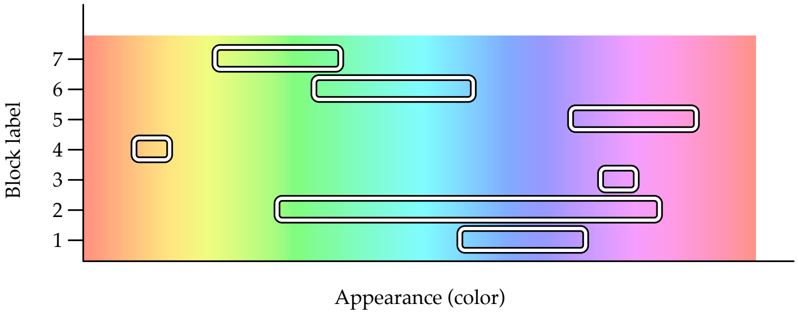


FIGURE 5.2: A cartoon depiction of a base measure for object model views where appearance characteristics θ_m and block labels z_m are not independent. Rectangles indicate high probability regions. Here, knowing the block label of a view does supply additional information about how it might appear; compare with the independent case in Figure 5.1.

Inserting this into Equation 5.1 yields an expression identical to the one at the start of this section. The rest of the BD-IHMM generative process proceeds from here as it did in §3.1.

5.1.2 The prior we were waiting for: a preview/overview

So far we have discussed two models that separately embody two of the object model representation stipulations we would like to encode in our prior. We can unite the desirable properties of both models and incorporate the third stipulation—that different views of visual objects tend to share some commonalities—by changing the joint top-level base measure H to remove the independence between θ and z, and by using a more expressive f than the one employed by the BD-IHMM.

Figures 5.1 and 5.2 portray the differences in expressiveness between the BD-IHMM top-level base measure and a top-level base measure where θ_m and z_m are no longer independent. For objects modeled with the BD-IHMM, knowing the block label of an object view can give you no additional insight into what the view looks like, since the generative density over all object views H_θ is the same. Under the new scheme, knowing block labels is informative, since each block label is associated with a different distribution over appearance characteristics.

An actual realization of this new variety of base measure will come later, but for now we still have to account for the other two desiderata of our object representation. This occurs in f, which we express to be based on the similarity of object views *and* on the block labels as well. Let $\mathrm{Sm}(\theta_n;\theta_m)$ be some function indicating how similar θ_n is to θ_m; requirements of this function are the same as those for f enumerated earlier. We now specify f as

$$f(\omega_n;\omega_m) = \mathrm{Sm}(\theta_n;\theta_m)\delta(z_m = z_n), \tag{5.2}$$

which says that the base measure mass associated with a transition between two object views will receive a multiplicative augmentation in proportion to the similarity between those two views (smoothness)—but only if those two views belong to the same object (persistence). All three desiderata for our object representation are now in place.

5.2 The structurally-modified infinite hidden Markov model

We refer to an infinite hidden Markov model altered under this new scheme, that is, one whose subordinate Dirichlet process base measures are modified according to Equation 5.1, as a *structurally-modified infinite hidden Markov model* (SM-IHMM). A BD-IHMM is an SM-IHMM, and so is a model that uses the kind of f described by Equation 5.2. Leaving things general for the time being, we can express the generative process of an SM-IHMM as:

$$\begin{aligned}
\omega_m \,|\, H &\sim H \\
\boldsymbol{\beta} \,|\, \gamma \sim \mathrm{SBP1}(\gamma), \quad \beta^*_{mn} &= \frac{\beta}{1+\xi}\left(1 + \frac{\xi f(\omega_n;\omega_m)}{\sum_k \beta_k \cdot f(\omega_k;\omega_m)}\right) \\
\boldsymbol{\pi}_m \,|\, \alpha_0, \boldsymbol{\beta}^*_m &\sim \mathrm{SBP2}(\alpha_0, \boldsymbol{\beta}^*_m) \\
v_t \,|\, v_{t-1}, \boldsymbol{\pi} \sim \boldsymbol{\pi}_{v_{t-1}}, \quad y_t \,|\, v_t, \boldsymbol{\omega} &\sim g(\omega_{v_t}),
\end{aligned} \tag{5.3}$$

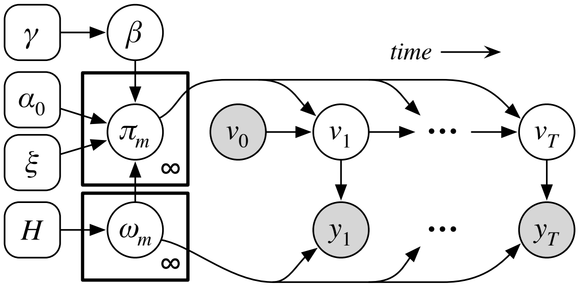


FIGURE 5.3: Graphical model depiction subsuming all structurally-modified infinite hidden Markov models. Descriptions of the variables appear in the text.

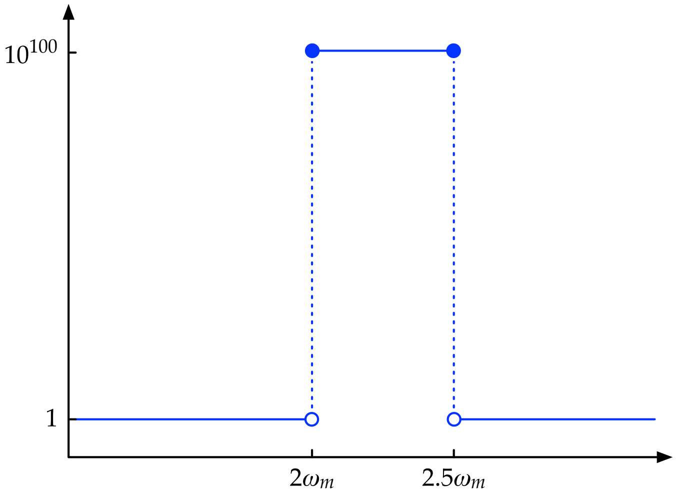
where H is the base measure and ω_m an atom sampled from H for hidden state m. All or part of ω_m may serve as parameters for the observation model for m, as determined by the parameterized density g. $\boldsymbol{\beta}^*_m$, the state m-specific modified version of the first-layer DP's mixing proportions $\boldsymbol{\beta}$, along with hyperparameter α_0, determine the distribution over probabilities for transitions out of state m. The hidden state trajectory $\boldsymbol{v}$ is sampled according to the countably infinite-dimensional transition matrix $\boldsymbol{\pi}$; this, and the observation models, determine the sampling of observations $\boldsymbol{y}$ corresponding to the trajectory.

A graphical depiction of this general generative process appears in Figure 5.3. While it may initially appear to be between the IHMM and the BD-IHMM in complexity, the additional dependencies from all ω_m to all π_m variables introduce considerable expressive power to the model—and, inevitably, a degree of additional necessary bookkeeping to the inference.

5.2.1 A necessary approximation for SM-IHMMs

As with the BD-IHMM, the base measure modification of Equation 5.1 involves a sum over a countably infinite number of terms. This sum is impossible to compute exactly, for only obvious reasons, and so for sampling and inference it must be approximated.

In §3.3, we considered bounds on the infinite sum in the BD-IHMM base measure modification term. In the SM-IHMM, the lack of restrictions on f makes it difficult to perform this kind of analysis in general. To see why, consider the following $f(\omega_n;\omega_m)$, where all ω values are simply positive real numbers:



Here, the blue function charts $f(\omega_n;\omega_m)$ for a specific ω_m. This function is small for most of its range, but for the interval between $2\omega_m$ and $2.5\omega_m$ it is overwhelmingly large. This sows chaos in the base measure modification for two reasons. First, we can expect the modification for any particular ω_n to vary wildly depending on whether ω_n falls into the interval. Second, the variance of any particular modification can also be discouragingly large depending on what *other* atoms fall into the interval.

This second point becomes clearer after imagining several different situations. First, consider the case where the prior on ω_m, H, places very nearly all ω_m into the interval $[2, 2.5]$. In this case, despite the extremes of f, we can state that within the base measure modification,

$$\frac{\xi f(\omega_n;\omega_m)}{\sum_k \beta_k \cdot f(\omega_k;\omega_m)} \approx \xi.$$

At the other extreme, where H is almost certain to place any given ω_m outside of $[2, 2.5]$, the above ratio is again very nearly ξ. Both of these two conditions exhibit little variance in base measure modification. Between them, however, it is a different story. This perilous medial circumstance can be illustrated by imagining that we know about three atoms, ω_1, ω_2, and ω_3. Assume that these are configured so that $f(\omega_1;\omega_3) = f(\omega_2;\omega_3) = f(\omega_3;\omega_3) = 1$. If we use the same approximation strategy for $\sum_k \beta_k \cdot f(\omega_k;\omega_m)$ that the BD-IHMM uses—replacing the missing sum terms with their expectation—we have, in this situation

$$\begin{aligned}\sum_k \beta_k \cdot f(\omega_k;\omega_3) &\approx \beta_1{+}\beta_2{+}\beta_3 + E_{\beta_4,\beta_5,\ldots,\omega_4,\omega_5,\ldots}\left[\sum_{k=4}\beta_k \cdot f(\omega_k;\omega_3)\right]\\ &\approx \beta_1{+}\beta_2{+}\beta_3 + \int_{\beta_4,\beta_5,\ldots}\int_{\omega_4,\omega_5,\ldots}\sum_{k=4}\beta_k \cdot f(\omega_k;\omega_3)\; dP(\omega_4,\omega_5,\ldots)\; dP(\beta_4,\beta_5,\ldots)\\ &\approx \beta_1{+}\beta_2{+}\beta_3 + \int_{\beta_4,\beta_5,\ldots}\sum_{k=4}\beta_k \int_{\omega_k} f(\omega_k;\omega_3)\; dP(\omega_k)\; dP(\beta_4,\beta_5,\ldots)\\ &\approx \beta_1{+}\beta_2{+}\beta_3 + (1-\beta_1{+}\beta_2{+}\beta_3)\int_{\omega_n} f(\omega_n;\omega_3)\; dP(\omega_n),\end{aligned} \tag{5.4}$$

with the third line taking advantage of independence among the ω_k, and the fourth observing that all configurations of $\beta_4, \beta_5, \ldots$ will yield the same sum.

Now imagine that β_4 and ω_4 are revealed to us, and that ω_4 lies within the interval $[2\omega_3, 2.5\omega_3]$. To update our approximate sum, we replace the expectation term accordingly, obtaining

$$\sum_k \beta_k \cdot f(\omega_k;\omega_3) \approx \beta_1{+}\beta_2{+}\beta_3 + 10^{100}\beta_4 + E_{\beta_5,\beta_6,\ldots,\omega_5,\omega_6,\ldots}\left[\sum_{k=5}\beta_k \cdot f(\omega_k;\omega_3)\right].$$

For base densities H whose support is coincident with and somewhat larger than the interval, this new approximation will be considerably larger than the old one, and the approximate modified base measure proportions dependent on this computation will change dramatically to match. While effects like these are also present for the BD-IHMM's more prosaic modification function, the absence of a towering "slab" like the one in the pathological example just discussed make it more tenable to use the expectation-based approximation described in §3.3. Likewise, if we wish to use an expectation-based approach for modification term denominator sums, as is done in (5.4), we are well advised to make our f functions similarly accommodating; ultimately, to make the variance of the modification term denominator sum relatively small.

5.3 Inference

Inference for SM-IHMMs has much in common with inference for the BD-IHMM described in previous chapters—here, as there, we must infer hidden state trajectories $\boldsymbol{v}$ for observations $\boldsymbol{y}$ and observation models $\boldsymbol{\theta}$. In some SM-IHMM applications, it may also be worthwhile to infer details of the modification function f; in this dissertation, though, f will be given and not inferred.[2]

Despite these overall commonalities, the specifics of SM-IHMM inference depends on the specifics of the models used in the framework. In this section we will examine a specific application of the SM-IHMM framework: the "prior we were waiting for" sketched in §5.1.2. As an elaboration of the BD-IHMM, this application allows us to reuse much of our existing inference infrastructure while still highlighting considerations that will arise in many SM-IHMM settings.

5.3.1 The similarity-structured block-diagonal infinite hidden Markov model

The similarity-structured block-diagonal infinite hidden Markov model (SS-BD-IHMM) is a burdensome name for an augmented BD-IHMM in which

- block labels z and observation models θ are not independent; in particular, observation models corresponding to states with the same block label can cluster together,
- modified base measure mass associated with transitions in the same block favor transitions between states whose observation models are similar to one another.

As in §5.1.1, we consider a base density H yielding atoms $\omega_m = \{\theta_m, z_m\}$ containing an observation model and a block label. We can unpack the generative inner workings of H as follows:

$$\begin{aligned} \phi_k \,|\, \phi_0 &\sim \phi_0 \\ \rho \,|\, \zeta &\sim \mathrm{SBP1}(\zeta) \\ z_m \,|\, \rho &\sim \rho \\ \theta_m \,|\, \boldsymbol{\phi}, z_m &\sim \phi_{z_m}. \end{aligned} \tag{5.5}$$

In this generative description, block labels are drawn from a categorical distribution $\boldsymbol{\rho}$, just as they were in the BD-IHMM. Observation models θ_m, however, now come from a corresponding block-specific distribution ϕ_{z_m}, which is drawn from a new top-level density ϕ_0. As described in §5.1.2, these block-specific distributions give additional semantic significance to a state's block membership.

[2] Note that although BD-IHMM block label $\boldsymbol{z}$ inference may seem to involve learning an important aspect of f, recall that within the SM-IHMM framework, block labels are actually paired with the observation models $\boldsymbol{\theta}$ in combined atoms $\boldsymbol{\omega}$. Thus, f remains fixed, and $\boldsymbol{\theta}$ and $\boldsymbol{z}$ inference are really just two aspects of inferring $\boldsymbol{\omega}$.

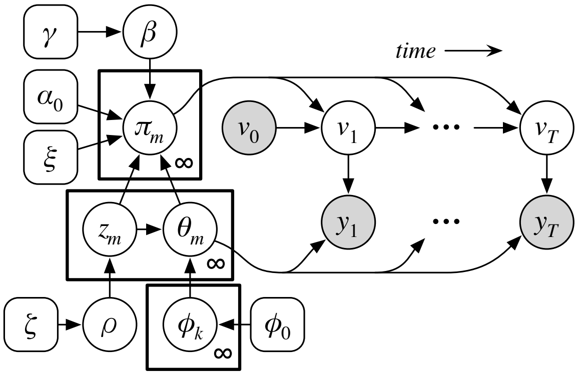


FIGURE 5.4: Graphical model depiction of the similarity-structured block-diagonal infinite hidden Markov model. Descriptions of the variables appear in the text. Compare with the general SM-IHMM graphical model in Figure 5.3; this depiction expands the H and ω_m variables in that model.

From here on we will refer to the variables in these inner workings specifically instead of the combined atoms ω_m and base measure H. We continue the generative account of SS-BD-IHMM as follows:

$$\begin{aligned} \boldsymbol{\beta} \,|\, \gamma \sim \mathrm{SBP1}(\gamma), \quad \beta^*_{mn} &= \frac{\beta}{1+\xi}\left(1 + \frac{\xi\,\mathrm{Sm}(\theta_n;\theta_m)\delta(z_m = z_n)}{\sum_k \beta_k \cdot \mathrm{Sm}(\theta_k;\theta_m)\delta(z_m = z_k)}\right) \\ \boldsymbol{\pi}_m \,|\, \alpha_0, \boldsymbol{\beta}^*_m &\sim \mathrm{SBP2}(\alpha_0, \boldsymbol{\beta}^*_m) \\ v_t \,|\, v_{t-1}, \boldsymbol{\pi} \sim \boldsymbol{\pi}_{v_{t-1}}, \quad & y_t \,|\, v_t, \boldsymbol{\omega} \sim g(\omega_{v_t}). \end{aligned} \tag{5.6}$$

Note the expansion of f into $\mathrm{Sm}(\theta_n;\theta_m)\delta(z_m = z_n)$, as mentioned in §5.1.2. Specific similarity functions $\mathrm{Sm}(\theta_n;\theta_m)$, tailored to individual applications, will be presented in the next chapter. Figure 5.4 shows a graphical depiction of the SS-BD-IHMM.

Since the SS-BD-IHMM has much in common with the BD-IHMM, we can modify the BD-IHMM's Gibbs sampling steps to suit the new model. We begin by detailing inference steps that require little modification between settings, then move on to steps needing greater customization.

5.3.2 BD-IHMM sampling steps requiring no change

Many of the descriptions of BD-IHMM sampling steps in Chapter 3 could be copied verbatim into a description of SS-BD-IHMM inference. These are:

- Block label probabilities $\boldsymbol{\rho}$ (§3.4.2)

- Hidden state trajectory $\boldsymbol{v}$, second sampling strategy (§3.4.6), although the difficult integral calculations necessary for a collapsed approach are probably not practical within the SS-BD-IHMM framework.
- Block label concentration hyperparameter ζ (§3.4.8)
- Top-level DP concentration hyperparameter γ (§3.4.9)
- Subordinate DP concentration hyperparameter α_0 (§3.4.10).
- Base measure modification hyperparameter ξ (§3.4.11).

Some of these steps require computation of the modified base measure proportion terms β^*_{mn}. Applying the approximation discussed in §5.2.1 to the modification described in the SS-BD-IHMM generative process (5.6), this is

$$\beta^*_{mn} = \frac{\beta_n}{1+\xi}\left(1+\frac{\xi\,\mathrm{Sm}(\theta_n;\theta_m)\delta(z_m=z_n)}{\beta_{\mathrm{new}}\rho_{z_m}\int \mathrm{Sm}(\theta;\theta_m)\,dP(\theta\,|\,\phi_{z_m})+\sum_{k=1}^{M}\beta_k\cdot\mathrm{Sm}(\theta_n;\theta_m)\delta(z_m=z_n)}\right), \tag{5.7}$$

where K is the largest block label of any of the instantiated blocks.

5.3.3 Observation model generating distribution parameters ($\boldsymbol{\phi}$)

Formally, the conditional density for ϕ_k, the observation model generating distribution parameters for block k, is conditionally independent of the rest of the model given ϕ_0 and all θ_m where $z_m = k$. Unfortunately, the integral in our approximation of the base measure modification denominator sum (§5.2.1) introduces a "hidden dependence" of $\boldsymbol{\pi}$ on $\boldsymbol{\phi}$. As long as this dependence is present, we may as well account for it in our ϕ_k conditional. Thus, instead of sampling a new ϕ_k from

$$p(\phi_k\,|\,\boldsymbol{\theta},\boldsymbol{z},\phi_0)\;\propto\;p(\phi_k\,|\,\phi_0)\prod_{\{m:\,z_m=k\}}p(\theta_m\,|\,\phi_k), \tag{5.8}$$

we instead sample

$$p(\phi_k\,|\,\boldsymbol{\theta},\boldsymbol{z},\phi_0,\boldsymbol{v},\beta,\rho,\alpha_0,\xi)\;\propto\;p(\phi_k\,|\,\phi_0)\prod_{\{m:\,z_m=k\}}p(\theta_m\,|\,\phi_k)\prod_{n}^{M}\frac{\Gamma(\alpha_0\beta^*_{mn}+c_{mn})}{\Gamma(\alpha_0\beta^*_{mn})}, \tag{5.9}$$

where M is the largest index of all the states visited in the trajectory $\boldsymbol{v}$.

This density cannot be sampled directly, so instead we use a simple variant of Metropolis-Hastings (MH) called independence chain Metropolis-Hastings. Like all MH variants, independence chain MH features an easier-to-sample proposal distribution that suggests new values for samples from the harder-to-sample target distribution—here, samples of ϕ_k drawn from the conditional density in Equation 5.9. Unlike traditional MH (c.f. §3.4.5, for example), however, this proposal distribution does not condition its proposal on the previous sample from the target density. Proposed samples are independent of prior samples; hence the name. Often this independence would be a liability: most MH proposal distributions exhibit a very strong dependence, proposing by design a new sample that is not very different from the previous one, ideally localizing the MH sampler in a high-probability region of the sample space. However, if a proposal distribution can generate samples that are fairly close to the target distribution without needing to refer to prior samples, independence chain MH can be effective, and can pay dividends in simplicity. Our conditional distribution for ϕ_k is just such a case. As a proposal distribution for ϕ_k samples, we simply use the mixture component density in Equation 5.8. Accordingly, the mixture component portions of the proposal and target distributions cancel in the acceptance ratio, which is simply the leftover trajectory probability:

$$a = \frac{\dfrac{\Gamma(\alpha_0 \beta^{*'}_{mn} + c_{mn})}{\Gamma(\alpha_0 \beta^{*'}_{mn})}}{\dfrac{\Gamma(\alpha_0 \beta^{*}_{mn} + c_{mn})}{\Gamma(\alpha_0 \beta^{*}_{mn})}},$$

where $\beta^{*'}_{mn}$ refers to the modified base measure proportion as computed with the proposed ϕ_k. In practical applications, alterations to ϕ_k seldom effect much change to the β^{*}_{mn}, or on the trajectory probability; this acceptance ratio is almost always very close to 1.

5.3.4 Observation model parameters (θ)

Our strategy for resampling observation model parameters $\boldsymbol{\theta}$ is virtually identical to the one for drawing new $\boldsymbol{\phi}$ values just described. Once again, independence chain Metropolis-Hastings is used to sample each individual state's emission model parameters θ_m one by one, and the proposal density is what would be the conditional distribution for these parameters in an ordinary HMM:

$$p_{\text{prop}}(\theta_m \mid \boldsymbol{\phi}, z_m, \boldsymbol{v}, \boldsymbol{y}) \propto p(\theta_m \mid \phi_{z_m}) \prod_{\{t:\, v_t = m\}} p(y_t \mid \theta_m). \tag{5.10}$$

As the target density, we use the actual θ_m conditional in the SS-BD-IHMM:

$$p(\theta_m \mid \boldsymbol{\phi}, z_m, \boldsymbol{v}, \beta, \rho, \alpha_0, \xi) \propto \\ p(\theta_m \mid \phi_{z_m}) \left(\prod_{\{t:\, v_t = m\}} p(y_t \mid \theta_m) \right) \left(\prod_m^M \prod_n^M \frac{\Gamma(\alpha_0 \beta^*_{mn} + c_{mn})}{\Gamma(\alpha_0 \beta^*_{mn})} \right). \quad (5.11)$$

The acceptance ratio works out to:

$$a = \frac{\displaystyle\prod_m^M \prod_n^M \frac{\Gamma(\alpha_0 \beta^{*\prime}_{mn} + c_{mn})}{\Gamma(\alpha_0 \beta^{*\prime}_{mn})}}{\displaystyle\prod_m^M \prod_n^M \frac{\Gamma(\alpha_0 \beta^{*}_{mn} + c_{mn})}{\Gamma(\alpha_0 \beta^{*}_{mn})}}.$$

Practical implementations of this sampling strategy should take advantage of opportunities to cancel out identical terms on both sides of this ratio.

In contrast with the $\boldsymbol{\phi}$ resampling, the configuration of individual observation model parameters θ_m can have a more profound effect on the hidden state trajectory probabilities that appear in this acceptance ratio. Accordingly, this acceptance ratio has considerably more variance than the one for ϕ_k, and more iterations of the sampler are sometimes necessary to ensure adequate exploration of the parameter space.

5.3.5 Preparing additional "unused" hidden states

Section 3.4.4 describes a procedure for instantiating additional states in the BD-IHMM. A similar step is necessary in SS-BD-IHMM inference as well, but the additional dependencies present in the newer model require a slightly more complex procedure. In particular, since observation model values $\boldsymbol{\theta}$ now affect the transition probabilities $\boldsymbol{\pi}$, we cannot sample these independently of the block labels $\boldsymbol{z}$ and base measure probability masses $\boldsymbol{\beta}$, which also affect those probabilities (as they did in the BD-IHMM). Moreover, the block labels $\boldsymbol{z}$ also determine the observation model generating distributions $\boldsymbol{\phi}$ from which particular observation models in $\boldsymbol{\theta}$ are sampled.

While we start just as we did in §3.4.4 and break off additional $\boldsymbol{\beta}$ proportions via the stick-breaking process (but see discussion under **Future approaches** in that subsection), the more tightly interleaved dependency structure of the SS-BD-IHMM makes sampling block labels and observation models for the newly-instantiated hidden states more complex. With most useful modification functions $\mathrm{Sm}(\cdot;\cdot)$, all $\boldsymbol{z}$ and $\boldsymbol{\theta}$ values can jointly have an effect on the likelihood function, which is the probability of the inferred hidden state trajectory. We therefore employ a Gibbs sampling scheme, iteratively resampling each novel $\boldsymbol{z}$ and $\boldsymbol{\theta}$ value so as to draw from their joint conditional distribution. We draw single z_m values as

$$z_m \,|\, \boldsymbol{v}, \boldsymbol{z}_{\setminus m}, \boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\rho}, \boldsymbol{\phi}, \alpha_0, \xi \;\sim\; \rho_{z_m} \cdot p(\theta_m \,|\, \phi_{z_m}) \cdot P(\boldsymbol{v} \,|\, \boldsymbol{\beta}, \boldsymbol{z}, \boldsymbol{\theta}, \alpha_0 \xi).$$

In most settings, computing the marginal probability of assigning z_m any hitherto-unused block label is a very difficult integration problem. It is much easier to give up on marginalizing altogether and approximate the existence of an infinite number of hitherto-unused blocks (and corresponding observation model generating distributions) via a truncated approximation of this arrangement, one that simply substitutes a finite collection of such blocks for the actual infinite set. Instantiating a collection of unused blocks is straightforward: update $\boldsymbol{\rho}$ by subdividing ρ_{new} according to the stick-breaking process, and draw corresponding observation model generating distributions directly from ϕ_0. In contrast to hidden state instantiation, the dependency structure of the model only requires these quantities to be sampled once.

Meanwhile, we draw single θ_m values as

$$\theta_m \,|\, \boldsymbol{v}, \boldsymbol{z}, \boldsymbol{\beta}, \boldsymbol{\theta}_{\setminus m}, \boldsymbol{\phi}, \alpha_0, \xi \;\sim\; p(\theta_m \,|\, \phi_{z_m}) \cdot P(\boldsymbol{v} \,|\, \boldsymbol{\beta}, \boldsymbol{z}, \boldsymbol{\theta}, \alpha_0 \xi),$$

applying the same Metropolis-Hastings technique described in §5.3.4.

5.3.6 Hidden state trajectory ($\boldsymbol{v}$)

As with the BD-IHMM, we use a two-method strategy for inferring hidden state trajectories in the SS-BD-IHMM. This strategy is virtually unchanged in the newer model; in fact, the description for the second method in §3.4.6 requires no amendments. The first method (§3.4.5) would also be applicable verbatim if it weren't for the expected sum computation in Equation 3.15. (Equation 3.16 remains the same.) We replace it with its SS-BD-IHMM equivalent as

$$\begin{aligned}\beta^{*[\text{in}]}_{m\,\text{new}} = \frac{\rho_{z_m}\beta_{\text{new}}}{1+\xi} \\ \cdot \left(1 + \frac{\xi \cdot \rho_{z_m} \int \mathrm{Sm}(\theta;\theta_m)\, dP(\theta \,|\, \phi_{z_m})}{\beta_{\text{new}}\rho_{z_m} \int \mathrm{Sm}(\theta;\theta_m)\, dP(\theta \,|\, \phi_{z_m}) + \sum_{k=1}^{M} \beta_k \cdot \mathrm{Sm}(\theta_k;\theta_m)\delta(z_m = z_k)}\right).\end{aligned} \tag{5.12}$$

5.3.7 Base measure probability masses (β) and “seesaw sampling”

Virtually all of the background discussion on $\boldsymbol{\beta}$ inference in §3.4.3 could be reproduced verbatim here, and readers are encouraged to refer back to that section for a refresher if necessary. The specifics of the SS-BD-IHMM diverge most notably from those of the BD-IHMM just before Equation 3.8, so we will pick things up from there. As then, the conditional density for $\boldsymbol{\beta}$ is not Dirichlet—rather, it can be expressed as a product of what looks like a Dirichlet density and extra terms relating to the base measure modification:

$$\begin{aligned} p(\beta_1, \beta_2, \ldots, \beta_M, \beta_{\text{new}} \,|\, \boldsymbol{q}, \boldsymbol{z}, \alpha_0, \xi) \;\propto\; & \operatorname{Dir}(q_{\cdot 1}, q_{\cdot 2}, \ldots, q_{\cdot M}, \gamma) \\ \cdot \prod_m^M \prod_n^M \Bigg(1 + & \frac{\xi \operatorname{Sm}(\theta_n; \theta_m) \delta(z_m = z_n)}{\beta_{\text{new}} \rho_{z_m} \int \operatorname{Sm}(\theta; \theta_m)\, dP(\theta \,|\, \phi_{z_m}) + \sum_{k=1}^M \beta_k \cdot \operatorname{Sm}(\theta_k; \theta_m) \delta(z_m = z_k)} \Bigg)^{q_{mn}} . \end{aligned} \tag{5.13}$$

In the BD-IHMM, it was possible to factor this proportional expression so that straightforward draws from Dirichlet distributions handled much of the work. More general SM-IHMM $\boldsymbol{\beta}$ conditional densities, including this one, do not afford this convenience. Instead, we are faced with a probability distribution expressed over a high-dimensional simplex that lacks an obvious simple sampling strategy.

We have used Metropolis-Hastings before to sample tough distributions, and a proposal distribution resembling the Dirichlet portion of Equation 5.13 might seem like a good place to start. In practice, a considerable majority of proposals generated this way are rejected, particularly if M is large. The reason for this may be attributable to a need for fidelity in a proposal distribution that increases with the dimensionality of the problem: consider that each degree of freedom in the problem gives the proposal distribution another chance to make a “bad” suggestion, and that an otherwise sterling proposal will very likely be rejected if even a single aspect of it is improbable according to the target distribution.

There are a number of techniques that adapt Metropolis-Hastings to high-dimensional problems. Multiple-try Metropolis [103] has the proposal distribution generate multiple samples at each iteration, essentially yielding several options for an encompassing proposal mechanism to choose from. This approach may improve the odds of a successful proposal, but as degrees of freedom increase, we may expect that fewer and fewer proposals in the ensemble will be free from faults. Meanwhile, Hessian-based Metropolis-Hastings attacks high dimensionality by using the curvature of the target distribution to customize the shape of the proposal distribution [104]. This approach is not ideal for our

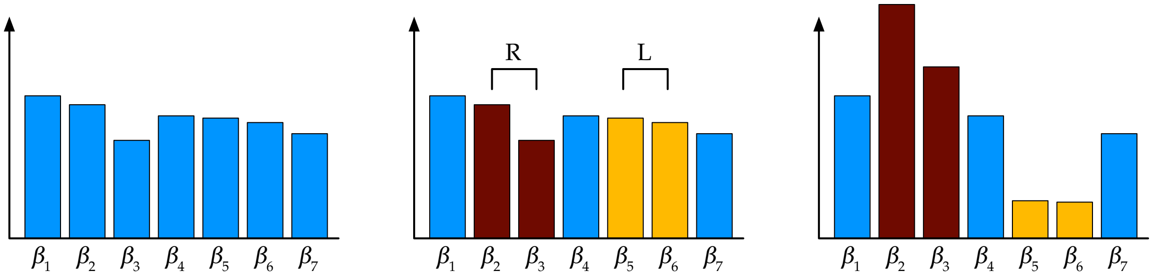


FIGURE 5.5: Graphical depiction of seesaw sampling. From left to right: the original distribution, then the designated "right" (R) and "left" variable sets. The variables in each set are randomly rescaled by a scalar factor such that the mass in both sets is conserved, and thus so the entire $\boldsymbol{\beta}$ sample sums to 1. The probability masses in the variable sets rise and fall like the ends of a seesaw, hence the name "seesaw sampling."

setting, where the mathematics of the base measure modification selected by the analyst, to say nothing of the general form of the modification itself (c.f. 5.3), or of the distribution's algebraically vexing residence on a simplex, can make it highly problematic to find the gradient for $\boldsymbol{\beta}$ analytically.

Instead of these options, we present a technique that takes advantage of the fact that the conditional density is expressed on a simplex, a technique that mitigates the challenge of high dimensionality through a coarse-to-fine sampling strategy. We refer to this method as "seesaw sampling" in light of its method of action, which involves repeatedly isolating two sets of variables within the $\boldsymbol{\beta}$ vector and rescaling their masses relative to each other by some scalar factor. This rescaling is done in a way that ensures that $\boldsymbol{\beta}$ always sums to 1. Visualized graphically, as in Figure 5.5, these ensembles of variables rise and fall like the ends of a seesaw.

To draw a $\boldsymbol{\beta}$ sample, the seesaw sampler iterates over a sequence of variable arrangements. Each arrangement partitions the variables into three sets: those that go on one side of the seesaw (let us arbitrarily call it the "left" side), those that go on the other, "right" side, and those whose masses will not change (since they are not on the seesaw, we may call them "spectators"). The seesaw sampler can only draw a valid $\boldsymbol{\beta}$ sample if the sequence of variable arrangements as an ensemble are admissible: in this case, if there is a way to seesaw mass around from any one configuration of $\boldsymbol{\beta}$ values to any other. Not all sequences are this way. First, consider the following valid set of arrangements for sampling a collection of six proportions, where L places a variable in the left set, R places a variable into the right set, and $\cdot$ makes the variable a spectator.

β_1	β_2	β_3	β_4	β_5	β_6
L	R	·	·	·	·
L	·	R	·	·	·
L	·	·	R	·	·
L	·	·	·	R	·
L	·	·	·	·	R

It is not hard to demonstrate that any one configuration of $\boldsymbol{\beta}$ values could be transformed to any other configuration of $\boldsymbol{\beta}$ values after repeated iteration through this sequence of partitions. In fact, in principle, only two iterations are necessary: in the first, all the mass is transferred from each right side element to the one left side element; then in the second pass, mass is portioned back into the right side elements as desired. Of course, it would be unusual if our sampler did shift mass around in exactly this way, but other, less serendipitous ways for the mass to diffuse could also happen, and in any case we have demonstrated that there is no "can't get there from here" situation implied by the set sequence. Now consider a second sequence:

β_1	β_2	β_3	β_4	β_5	β_6
L	R	·	·	·	·
L	·	R	·	·	·
L	L	R	·	·	·
·	·	·	L	R	·
·	·	·	L	·	R
·	·	·	L	L	R

It is clear that there is no way for mass from the first three variables to work its way to the second three variables. This set sequence is not admissible, and so it cannot be used.

While admissibility is necessary for a viable seesaw sampler sample set sequence, it is not enough to ensure an efficient sampler. In the valid set sequence we considered previously, β_1 takes on the role as a kind of depot for probability mass, a tight aperture through which all mass must pass on the way to its destination. The chances of this actually occurring in any reasonable amount of time are slim, particularly if the $\boldsymbol{\beta}$ conditional makes it more probable that β_1 should be small. In practice, a more effective schedule for the sampler looks something like this:

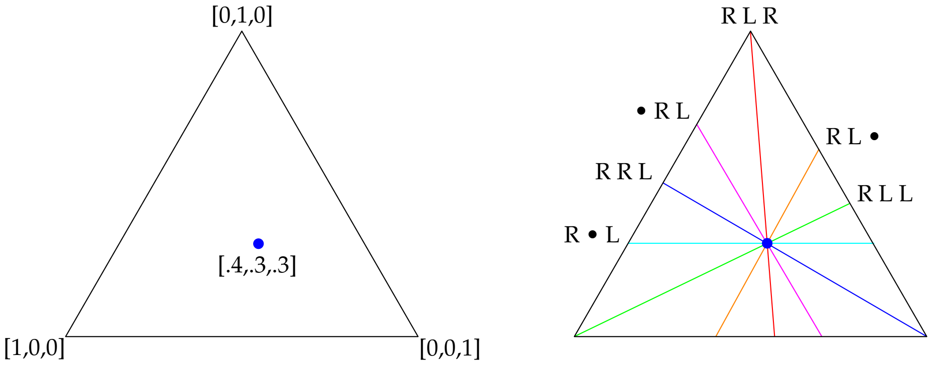


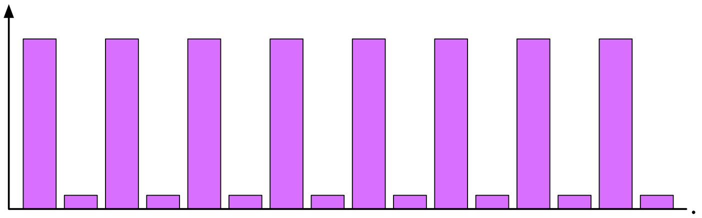
FIGURE 5.6: Left: a point on the 2-simplex. Right: the paths within the simplex created by starting at the point and seesawing in any of the six possible, meaningfully distinct seesaw configurations.

β_1	β_2	β_3	β_4	β_5	β_6
L	L	L	R	R	R
L	L	R	R	·	·
·	L	L	R	R	·
·	·	L	L	R	R
L	R	·	·	·	·
·	L	R	·	·	·
·	·	L	R	·	·
·	·	·	L	R	·
·	·	·	·	L	R

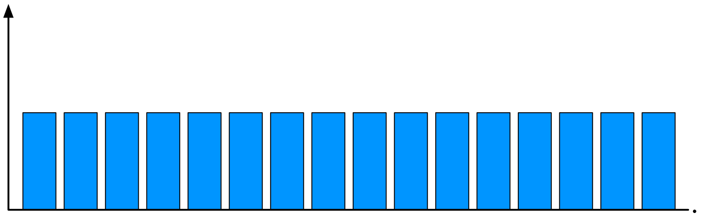
Certainly there are now more ways for probability mass to move around in this new scheme: it can slosh *en masse* according to the first arrangement or trickle from place to place while the sampler iterates through the final five. This added flexibility gives the seesaw sampler more ways to explore the simplex and locate modes of the target distribution, though of course further arrangements are possible. “Ways” is an apt word to use, since in fact any possible arrangement, combined with an initial point on the simplex, describes a line within the simplex. Figure 5.6 shows the lines corresponding to all possible seesaw set arrangements in the 2-simplex.

Constrained to follow lines like the ones in Figure 5.6, the seesaw sampler can only go in a finite set of directions from any given point on the simplex. This said, the number of lines increases combinatorially with each additional $\boldsymbol{\beta}$ dimension,[3] which gives us hope that the finite nature of this set seldom will not constrain us too much. Indeed, before too long, the naive impulse to simply enumerate all the lines and choose one that looks promising soon becomes frustrated by the sheer number of possible directions.

In light of this, our approach sacrifices one naive approach for a slightly less naive one: the set schedules just described. This approach presumes that we can create a collection of directions to search that generally give the sampler a decent opportunity to progress toward a mode of the target distribution. In practice, this appears to be the case, provided that we have some knowledge beforehand about which variables are likely to have similar values in the target distribution. A thought experiment that illustrates the importance of this knowledge can be illustrated. Consider a circumstance where the target distribution centers most of its mass tightly around the following arrangement of proportions:

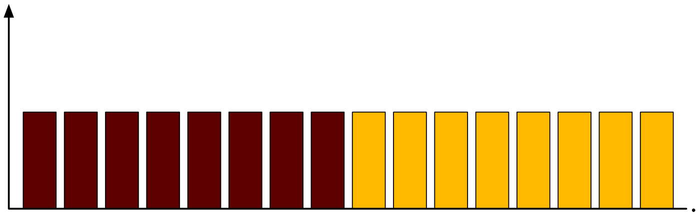


Lacking knowledge about the target distribution, we might initialize the sampler with the uniform distribution:

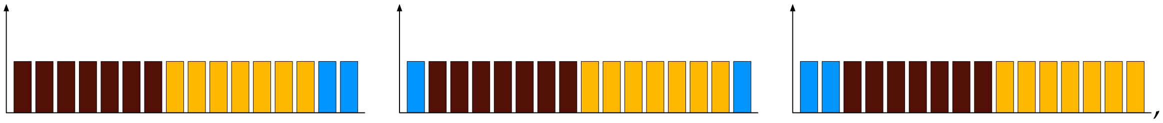


We then embark on a seesaw sampling schedule in the same pattern as the "effective schedule" discussed previously. The first arrangement balances the first half of the variables against the second half:

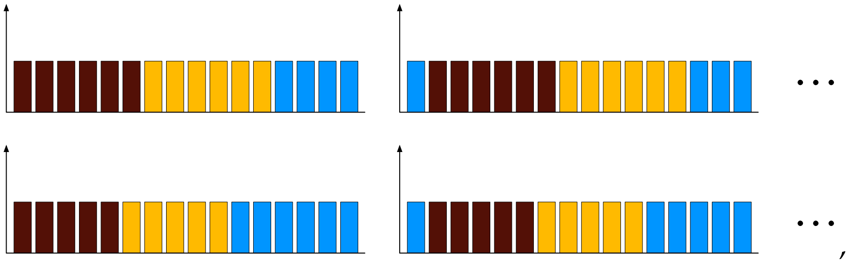
[3] In fact, the number of lines for the K-simplex can be shown to be $S(K+2,3)$, where $S(a,b)$ is the Stirling number of the second kind for arguments a and b. From $K=1$, this sequence increases rapidly: 1, 6, 25, 90, 301, 966, 3,025, 9,330, 28,501, and so on.



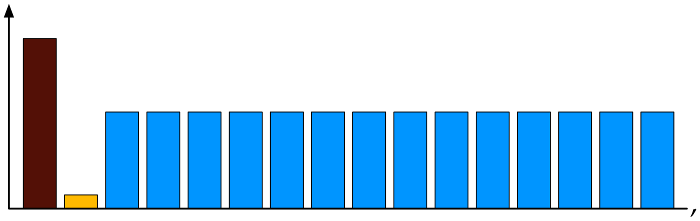
Unfortunately, within the target distribution, the total masses in both sets are very similar, even though adjacent variables within the sets have very distinct masses. This situation repeats within the next three sample set arrangements:



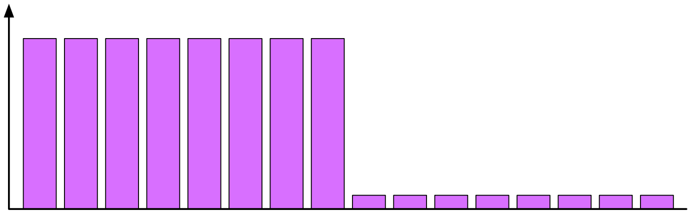
where variables whose masses are colored blue are placed in the spectator set. Subsequent arrangements, which place more variables in the spectator set, are similarly unable to advance the sample any closer to the mode:



and so on. Only when we get to the last set of arrangements, where the left and right sets are singletons, does the structure of the density allow us to make some progress:



and onward from there. Now, if we had reordered the variables in the beginning so that more similar variables were next to each other, the target distribution would have looked like this:


,

and the sampler would have arrived at the mode with the first sample set arrangement. The takeaway message from this demonstration is that the similarities and differences between the masses that variables take on in the density has a significant effect on the usefulness of different portions of the schedule.

In practice, for $\boldsymbol{\beta}$ densities associated with the BD-IHMM and the SS-BD-IHMM, most pairings of small seesaw sets are sufficiently decoupled to allow some degree of seesawing. Pairs of randomly-selected larger sets, by contrast, will nearly always have a similar arrangement of large and small values, making it unlikely to be the case that most elements in one set need more mass while most elements in the other set need less.

To solve this problem and make all parts of the "effective schedule" useful, we simply change the order of the elements in the $\boldsymbol{\beta}$ vector so that their corresponding $q_{\cdot m}$ values are sorted. Unless an extreme modification function $\mathrm{Sm}(\cdot;\cdot)$ is used or the ξ hyperparameter is exceedingly large, the Dirichlet factor of Equation 5.13 will require that $\boldsymbol{\beta}$ values and $q_{\cdot m}$ values tend to have the same relative proportions. Rearranging the $\boldsymbol{\beta}$ values according to the $q_{\cdot m}$ sort order therefore makes it much more likely that the large left and right seesaw sets at the beginning of the schedule contain collections of $\boldsymbol{\beta}$ values that, in the conditional density, have similar masses within the sets and distinct masses between the sets—and thus can seesaw effectively. In the end, $\boldsymbol{\beta}$ reordering and the effective schedule induce a coarse-to-fine strategy in the inference: the sampler makes broad, rough adjustments with the large-set arrangements that come first, then progressively finer adjustments with the increasingly smaller sets.

Because rearranging $\boldsymbol{\beta}$ changes the set of available "paths" along which the seesaw sampler may explore the $\boldsymbol{\beta}$ posterior, readers may wonder whether different $\boldsymbol{\beta}$ orderings might cause the $\boldsymbol{\beta}$ sampler to yield different sampling outcomes. We should hope this is not the case: the $\boldsymbol{\beta}$ sampler is meant to draw exclusively from its posterior. As we will describe, though, $\boldsymbol{\beta}$ reordering and the effective schedule are simply used to make useful proposals in Metropolis-Hastings samples from the actual $\boldsymbol{\beta}$ posterior. Different $\boldsymbol{\beta}$ orderings and schedules may effect the rate at which the sampler converges on and explores probable regions of the posterior, but they do not change the fact that the posterior is the distribution that gets sampled in the end.

The mathematics of seesaw sampling rely on a variable transformation similar to the one we saw in §3.4.3. If the original $\boldsymbol{\beta}$ values that we confront in actual sampling are

$$\boldsymbol{\beta} = \{\beta_1, \beta_2, \ldots, \beta_M, \beta_{\text{new}}\},$$

we consider a derived set of values comprising something like

$$\boldsymbol{G} = \{g_{S1}, g_{S2}, \ldots, G_L, g_{Li}, g_{Lj}, \ldots, G_R, g_{Rk}, g_{Rl}, \ldots, g_{SM}, g_{S\text{new}}\}.$$

Those values with an L subscript are associated with the left seesaw set, while those with an R subscript are associated with the right subset. Within $\boldsymbol{G}$, the variable transformation substitutes these values for the $\boldsymbol{\beta}$ values assigned to those sets. All $\boldsymbol{\beta}$ values in the third "spectator" set, are copied directly from $\boldsymbol{\beta}$ to $\boldsymbol{G}$. These are symbolized above with g_{S1}, g_{S2}, and g_{SM}.

G_L and G_R are the total masses of the left and right seesaw sets:

$$G_L = \sum_{i \in L} \beta_i, \qquad G_R = \sum_{i \in R} \beta_i.$$

Each g_{Li} or g_{Ri} is the relative proportion of its corresponding β_i value in the left or right seesaw sets, or

$$g_{Li} = \frac{\beta_i}{G_L}, \qquad g_{Ri} = \frac{\beta_i}{G_R},$$

while $g_{Si} = \beta_i$ as mentioned above. The objective of seesaw sampling is to draw new values for G_L and G_R by sampling a proportional value $c \in (0, 1)$:

$$G'_L = c(G_L + G_R), \qquad G'_R = (1 - c)(G_L + G_R).$$

The conditional density for c, derived from Equation 5.13, is

$$p(c \mid \boldsymbol{q}, \boldsymbol{G}, \boldsymbol{z}, \alpha_0, \xi) \propto \text{Beta}(c\,;\, \textstyle\sum_{i \in L} q_i, \sum_{i \in R} q_i) \\ \cdot \prod_m^M \prod_n^M \left(1 + \frac{\xi\,\text{Sm}(\theta_n; \theta_m)\delta(z_m = z_n)}{g_{S\text{new}}\rho_{z_m} \int \text{Sm}(\theta; \theta_m)\, dP(\theta \mid \phi_{z_m}) + \text{SmSum}(m)}\right)^{q_{mn}}. \quad (5.14)$$

SmSum is a shorthand we will use here (and only here) to fit our math within the page. It expands to

$$\begin{aligned}\mathrm{SmSum}(m) \;=\; & \quad c\;(G_L+G_R)\sum_{k\in L} g_{Lk}\cdot \mathrm{Sm}(\theta_k;\theta_m)\delta(z_k{=}z_m)\\ & +(1-c)(G_L+G_R)\sum_{k\in R} g_{Rk}\cdot \mathrm{Sm}(\theta_k;\theta_m)\delta(z_k{=}z_m)\\ & + \sum_{k\notin R\cup L} g_{Sk}\cdot \mathrm{Sm}(\theta_k;\theta_m)\delta(z_k{=}z_m),\end{aligned} \tag{5.15}$$

with the three terms corresponding to the left seesaw set, the right seesaw set, and the spectator set respectively.

We apply Metropolis-Hastings sampling to the one-dimensional c density in Equation 5.14. The distribution that proposes new values c' is a beta distribution whose mean is (usually) the original c value and whose variance σ_c^2 is a user specified parameter:

$$p(c' \mid c) = \mathrm{Beta}(a_c, b_c),$$

where we specify that

$$\begin{aligned} a_c &= \begin{cases} \dfrac{c^2 - c^3 - c\sigma_c^2}{\sigma_c^2} & \text{if } c - c^2 > \sigma_c^2,\\ 1 & \text{otherwise; and}\end{cases}\\ b_c &= \begin{cases} \dfrac{c - 2c^2 + c^3 - \sigma_c^2 + c\sigma_c^2}{\sigma_c^2} & \text{if } c - c^2 > \sigma_c^2,\\ 1 & \text{otherwise.}\end{cases}\end{aligned} \tag{5.16}$$

For some values of c, there is no beta distribution with the user-specified variance σ_c^2. For these, as coded in (5.16), we fall back on a uniform distribution for c' proposals.

Applying the seesaw sampler to other densities on the simplex besides Equation 5.13 involves a similar procedure: for each seesaw set arrangement, transform the density to a form that uses the G_L, g_{Li}, G_R, g_{Ri} parameterization, then use Metropolis-Hastings to sample the c factor that determines the new seesaw set proportions G'_L and G'_R. Transformed densities are not difficult to obtain—given some density on the $M-1$ simplex,

$$p(\boldsymbol{\beta}) = h(\beta_1, \beta_2, \ldots, \beta_M),$$

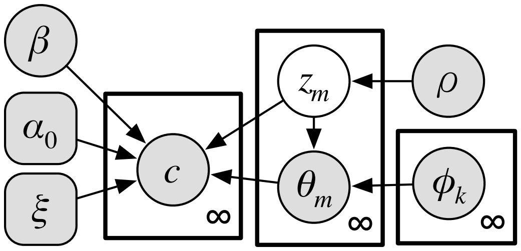


FIGURE 5.7: A graphical model, simplified from the SS-BD-IHMM graphical model in Figure 5.4, showing only the variables relevant to sampling the hidden state block labels $\boldsymbol{z}$ via the methods of §5.3.8. Compare with the analogous BD-IHMM model in Figure 4.1.

the transformed version will take a form like

$$p(\boldsymbol{G}) = G_L^{|L|-1} \cdot G_R^{|R|-1} \cdot h(g_{S1}, g_{S2}, \ldots, G_L g_{Li}, G_L g_{Lj}, \ldots, G_R g_{Rk}, G_R g_{Rl}, \ldots, g_{SM})$$

depending on which states are in which seesaw sets. The notationally vague third term simply states that the $\boldsymbol{\beta} \to \boldsymbol{G}$ transformation is "reversed" within the invocation of the density function h. This general form is the starting point for implementing seesaw samplers for the BD-IHMM's block proportion density in Equation 3.12

That about does it for seesaw sampling. To sum it up, we have tackled a troublesome probability distribution (the $\boldsymbol{\beta}$ conditional density) with a general purpose method for efficiently sampling densities defined over simplexes. This method does not require the user to supply much information besides the density, and in practical applications it has been applied to distributions with over 2,500 dimensions.

5.3.8 Hidden state block labels ($\boldsymbol{z}$)

In the SS-BD-IHMM, hidden state block label inference adapts the auxiliary variable-based approach described in Chapter 4. The main novelty is that the choice of a block label for a particular hidden state now also determines which observation model generating distribution gave rise to the hidden state's observation model. Figure 5.7 depicts the dependency relationships between the variables involved in $\boldsymbol{z}$ resampling, including the new observation model dependency just mentioned.

A few other important differences exist in addition to the new dependency structure. While the auxiliary variable intuition described in §4.2 and §4.3 still applies, the computations determining how the auxiliary variables guide the block label inference must be revised to reflect the more generalized structural modification present in the SS-BD-IHMM. Recall how for the BD-IHMM, §4.2 expands the probability of the hidden state trajectory transition counts matrix $\boldsymbol{c}$ into the product of per-transition probability terms like this (c.f. Equations 4.2 and 4.3):

$$\frac{c_{m_t\cdot,t-1} + \frac{\alpha_0\beta_{n_t}}{1+\xi} + \delta(z_{m_t} = z_{n_t}) \cdot \frac{\alpha_0\beta_{n_t}}{1+\xi} \cdot \frac{\xi}{\sum_k \beta_k \cdot \delta(z_{m_t} = z_k)}}{c_{m_t\cdot,t-1} + \alpha_0},$$

where t reflects the order in which each individual transition tallied in $\boldsymbol{c}$ is considered. In the third term of the nominator of this expression, we see the extra bias favoring transitions between hidden states with the same block label. In the SS-BD-IHMM, this bias is still present, but now it is modulated by the modification function $\mathrm{Sm}(\cdot;\cdot)$:

$$\frac{c_{m_t\cdot,t-1} + \frac{\alpha_0\beta_{n_t}}{1+\xi} + \delta(z_{m_t} = z_{n_t}) \cdot \frac{\alpha_0\beta_{n_t}}{1+\xi} \cdot \frac{\xi\,\mathrm{Sm}(\theta_{n_t};\theta_{m_t})}{\sum_k \beta_k \cdot \mathrm{Sm}(\theta_k;\theta_{m_t})\delta(z_{m_t} = z_k)}}{c_{m_t\cdot,t-1} + \alpha_0}. \tag{5.17}$$

Building on the same logic detailed in Chapter 4, we can derive an auxiliary variable-based sampler that "bonds" the block labels of two hidden states together by asserting that a transition between the states would not have happened if it hadn't been for the extra probability mass contributed by the third numerator term. First, the complete augmented likelihood for the block labels, corresponding to Equation 4.7 in the original sampler, is

$$\begin{aligned} P(\boldsymbol{c},\boldsymbol{\theta}\,|\,\boldsymbol{q},\boldsymbol{r},\boldsymbol{z},\boldsymbol{\beta},\alpha_0,\xi) &= \left(\prod_{m=1}^{M} p(\theta_m\,|\,\phi_{z_m})\right) \\ &\cdot \prod_{t=1}^{T} \Theta\left(\frac{c_{m_t n_t,t-1} + \frac{\alpha_0\beta_{n_t}}{1+\xi} + \delta(z_{m_t} = z_{n_t}) \cdot \frac{\alpha_0\beta_{n_t}}{1+\xi} \cdot \frac{\xi\,\mathrm{Sm}(\theta_{n_t};\theta_{m_t})}{\sum_k \beta_k\cdot\mathrm{Sm}(\theta_k;\theta_{m_t})\delta(z_{m_t}=z_k)}}{c_{m_t\cdot,t-1} + \alpha_0} - q_t \right)^{r_t} \\ &\cdot \left(\frac{c_{m_t n_t,t-1} + \frac{\alpha_0\beta_{n_t}}{1+\xi} + \delta(z_{m_t} = z_{n_t}) \cdot \frac{\alpha_0\beta_{n_t}}{1+\xi} \cdot \frac{\xi\,\mathrm{Sm}(\theta_{n_t};\theta_{m_t})}{\sum_k \beta_k\cdot\mathrm{Sm}(\theta_k;\theta_{m_t})\delta(z_{m_t}=z_k)}}{c_{m_t\cdot,t-1} + \alpha_0} \right)^{1-r_t}. \end{aligned} \tag{5.18}$$

See Chapter 4 for a full explanation of Θ and the auxiliary variables q_t and r_t. Note that the likelihood also now accounts for drawing the hidden state observation models from the $\boldsymbol{\phi}$

distributions designated by the block labels z.

Our sampler computes the probability of binding hidden states together given a collection of transitions between those states. Based on these computations, it sets the r_t auxiliary variables to improve sampling performance (c.f. discussion in §4.4). For SS-BD-IHMM block inference, we specify a formula for computing binding probabilities analogous to Equation 4.8 as

$$P(\mathrm{Bond}_{mn} \,|\, \tau) = \frac{\Gamma\left(c_{mn} + \frac{\alpha_0 \beta_n}{1+\xi}\right) \Gamma\left(\frac{\alpha_0 \beta_n}{1+\xi} + \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi \, \mathrm{Sm}(\theta_{n_t};\theta_{m_t})}{\sum_k \beta_k \cdot \mathrm{Sm}(\theta_k;\theta_{m_t})\delta(z_m = z_k)} + \tau - 1\right)}{\Gamma\left(c_{mn} + \frac{\alpha_0 \beta_n}{1+\xi} + \frac{\alpha_0 \beta_n}{1+\xi} \cdot \frac{\xi \, \mathrm{Sm}(\theta_{n_t};\theta_{m_t})}{\sum_k \beta_k \cdot \mathrm{Sm}(\theta_k;\theta_{m_t})\delta(z_m = z_k)}\right) \Gamma\left(\frac{\alpha_0 \beta_n}{1+\xi} + \tau - 1\right)}. \tag{5.19}$$

Most of the sampler algorithm described in §4.5, as well as the enhancements described in §4.6, can be adapted to SS-BD-IHMM inference in a straightforward way, substituting terms from the above two expressions for the places where terms from Equations 4.7 and 4.8 appear. Supplemental calculations will be necessary in pseudocode steps 4 and 5 to incorporate the observation model densities $p(\theta_m \,|\, \phi_{z_m})$, but these additions are also straightforward: these are factored into (or out of) the conditional at the same moments that transitions between corresponding hidden states are also included or discarded.

There is, however, one important remaining detail. In the BD-IHMM, for all hidden states m in the same block (call it j), and for all transitions from m to state n in block j, the base measure modification multiplies the corresponding base proportion β_n by the same factor, namely

$$\frac{\alpha_0}{1+\xi}\left(1 + \frac{\xi}{\sum_k \beta_k \cdot \delta(z_k = j)}\right).$$

This single factor enabled us to devise an efficient means of checking whether sampled bonds between hidden states would be violated by changes to z, in a very particular way, as described in the beginning of §4.5. The specific bond violation that this checking concerns itself with occurs when the sum $\sum_k \beta_k \cdot \delta(z_k = j)$ grows as a collection of states are merged into block j. As this sum grows, the modifying factor that "boosts" β_n shrinks. Under some circumstances, this shrinkage could be severe enough that an auxiliary variable value q_t would exceed the probability of the transition happening at all—a violation ruling out that particular z configuration. Thanks to the uniform base measure modification strategy used by the BD-IHMM, it was not necessary to check all bonds for this kind of violation—instead, it was only necessary to identify and store a single "most susceptible" bond for each set of bound-together hidden states. If any bond in the set were to be violated by a label change in the manner described above, this bond would always be violated first.

In the SS-BD-IHMM, by contrast, each transition between states m and n in the same block has its own unique factor for modifying β_n:

$$\frac{\alpha_0}{1+\xi}\left(1+\frac{\xi\,\mathrm{Sm}(\theta_n;\theta_m)}{\sum_k \beta_k\cdot\mathrm{Sm}(\theta_k;\theta_m)\delta(z_k=j)}\right).$$

For this reason, it is no longer possible to identify which transition is the "most susceptible" for a whole set of bound-together hidden states. It *is* possible, though, to identify which transition is the most violation-susceptible of all bound transitions emanating from a specific hidden state. This is because even though each β^*_{mn} has its own special modification factor, β^*_{mn} for a specific m and *all* n in the same block will all scale by the same factor as states are added or removed to the block. Referring to step 3(d)iiiC in the pseudocode algorithm in Chapter 4, we compute a specially normalized bond value $q^*_{mn,i}$ for each bonded state as

$$q^*_{mn,i}\leftarrow\frac{q_{mn,i}-(i-1)-\frac{\alpha_0\beta_n}{1+\xi}}{\frac{\alpha_0\beta_n}{1+\xi}\cdot\frac{\xi\,\mathrm{Sm}(\theta_n;\theta_m)}{\sum_k \beta_k\cdot\mathrm{Sm}(\theta_k;\theta_m)\delta(z_m=z_k)}}.$$

Next, we determine if this bond value is the most violation-susceptible of all bonded transitions emanating from state m:

$$q^*_m\leftarrow\max(q^*_m,q^*_{mn,i}).$$

These q^*_m values replace the $q^*_{s_m}$ values employed in the BD-IHMM block label sampler. Later, in this replacement for Chapter 4 pseudocode step 5(b)ii, we determine whether any of the violation-susceptible bonds have been violated:

```
**for each** hidden state $n$ in the union of set $m$ and block $j$,
    **if** $q^*_n > \frac{\sum_i \beta_i\cdot\mathrm{Sm}(\theta_i;\theta_n)\delta(z_i=j)}{\sum_i \beta_i\cdot\mathrm{Sm}(\theta_i;\theta_n)\max(\delta(z_i=j),\delta(s_i=m))}$ **then**
        $P_j=0$.
        **continue** to the next candidate block label.
```

Finally, as in Chapter 4, we refer frequently to the base measure modification term denominator sum in its theoretical formulation: $\sum_k \beta_k\cdot\mathrm{Sm}(\theta_k;\theta_{m_t})\delta(z_m=z_k)$. This sum must be approximated (c.f. §5.2.1 and the **Approximation of infinite sums** note in §4.5), and in most parts of the sampler, it suffices to substitute in

$$\beta_{\text{new}}\rho_{z_m}\int \text{Sm}(\theta;\theta_m)\,dP(\theta\,|\,\phi_{z_m}) + \sum_{k=1}^{M}\beta_k\cdot\text{Sm}(\theta_k;\theta_m)\delta(z_m = z_k).$$

5.4 In use

Sections §6.3 and §6.4 compares the performance of the SS-BD-IHMM with that of the BD-IHMM on an artificial data task and the object model learning task that motivates this thesis.

Chapter 6

Demonstrations and Experiments

The previous chapters have described structurally-modified Bayesian nonparametric hidden Markov models—and methods for doing inference in these models—in some detail. Here at last we put all of this infrastructure to work. We begin by applying the BD-IHMM to a series of problems involving both artificial and real-world data. Next, we confront the problem that motivated this thesis in the first place: organization of spontaneous visual object data. We examine how the SS-BD-IHMM really does achieve better object modeling performance—by using appearance cues to group object views, and by making more precise assumptions about the transition structure between object views in the absence of visual data. Finally, we examine an additional use for some of the BD-IHMM/SM-IHMM inference machinery.

6.1 Evaluation methods

The models presented in this thesis accomplish two complementary tasks. They discover sub-behavior structure in training data in an unsupervised way, and they construct a generative model that describes both the hidden dynamics underlying the data and the observations that arise from these dynamics. To determine how well our new models perform these tasks, we need principled performance measures. We describe these measures now.

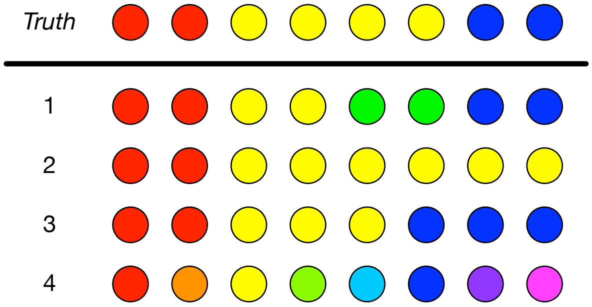


FIGURE 6.1: Different partitions of a set of eight items, with each column corresponding to a distinct item and each row corresponding to a partition. If the top partition is the "right" one, which of the other four is least wrong?

6.1.1 Evaluating structure identification: the adjusted Rand index

For a few of the problem settings we consider in this chapter, sub-behavior discovery evaluation is straightforward. Either the model does a good job of partitioning the training data into different behavioral regimes, or it fails utterly. In these cases, a visual review of the results reveals that it is sufficient to merely count the number of sub-behaviors the algorithm asserts is there. If it's the right number, the algorithm has found the correct answer.

In most cases, however, the situation is more ambiguous. Figure 6.1 summarizes the problem graphically: imagine that the top row is the correct way to label eight time steps of training data, and four competing algorithms have come up with the labelings on the following rows. Which of the partitions is closest to the truth? The four partitions of the training data could be ranked in different ways from "best" to "worst" depending on whether the ranker prefers to err on the side of overpartitioning, underpartitioning, or other criteria. Without knowing the needs or preference of the reader, we will fall back on a well-used method for partition comparison based on pairwise agreement between block assignments in both partitions. This method is known as the adjusted Rand index [105], a refinement of the Rand index that allows us to make meaningful comparisons of partition quality across multiple distinct datasets.

We begin a brief characterization of the adjusted Rand index by describing the original Rand index first. Consider two partitions $P1$ and $P2$, each assigning the N elements in some set of interest to distinct blocks. Considering each of the $\binom{N}{2}$ possible pairings of elements in the set, we can compute the following four statistics with respect to the partitions:

- A : the number of pairings where $P1$ and $P2$ *agree* that both elements are in the same block,
- B : the number of pairings where $P1$ and $P2$ *agree* that both elements are in different blocks,
- C : the number of pairings where $P1$ says the elements are in the same block, but $P2$ says they are in different blocks,
- D : the number of pairings where $P1$ says the elements are in different blocks, but $P2$ says they are in the same block.

The Rand index is simply the ratio of "agreement" pairs to all possible pairs, or

$$\text{Rand}(P1, P2) = \frac{A+B}{A+B+C+D} = \frac{A+B}{\binom{N}{2}},$$

a value that ranges in $[0, 1]$, with 1 corresponding to $P1$ and $P2$ being identical. It is perhaps easier to say that the Rand index is the ratio of the number of pairwise block assignment agreements to the number of possible element pairs, but breaking the index down into all four statistics helps illustrate how different aspects of similarity or difference between the partitions affect the outcome of the index.

The usefulness of the Rand index is limited in applications like ours, since different ground truth element labelings will tend to yield different ranges of Rand index, even if the quality of inferred labelings is similar in both circumstances. Because we randomly separate the data into training and test sets many times as we run our experiment, we are considering clustering performance over separate datasets with separate ground truth partitions. It is necessary to adjust the Rand index for these separate runs to attain useful, comparable statistics of temporal sub-behavior identification performance.

The adjusted Rand index was created to achieve these statistics. It computes the following:

$$\frac{\text{Rand}(P1, P2) - \text{Expected Rand index}}{\text{Maximum Rand index} - \text{Expected Rand index}},$$

where Maximum Rand index is 1, and Expected Rand index is the expected value of the Rand index with respect to a particular joint distribution over two partitions. Within this joint distribution, the first partition has the same number of blocks as $P1$; these blocks have the same number of elements as their counterparts in $P1$, though *which* elements they have may differ. The other partition in the joint distribution has the same arrangement with respect to $P2$. To draw a sample from this distribution, we go through each "position" in the first partition (e.g. "the third element of the fourth block") and draw elements from the dataset without replacement to fill those positions. After this first partition has been sampled, the dataset elements are put back in play and then similarly assigned without replacement to positions in the second partition. We thereby achieve two random partitions of the dataset elements.

With considerable patience, it would be possible to derive the adjusted Rand index all the way up from this generative characterization of the partition distribution, but it might be faster just be to have the expression for the adjusted Rand index, which after simplification amounts to:

$$\text{AdjRand}(P1, P2) = \frac{\sum_{i,j}\binom{n_{ij}}{2} - \frac{\sum_i\binom{n_{i\cdot}}{2}\sum_j\binom{n_{\cdot j}}{2}}{2}}{\frac{1}{2}\left(\sum_i\binom{n_{i\cdot}}{2} + \sum_j\binom{n_{\cdot j}}{2}\right) - \frac{\sum_i\binom{n_{i\cdot}}{2}\sum_j\binom{n_{\cdot j}}{2}}{2}},$$

where n_{ij} is an entry in the contingency table for the two partitions, or, the number of elements assigned to block i in $P1$ and block j in $P2$, and $n_{\cdot j}$ is the same kind of summing notation seen elsewhere in this thesis.

Intuitively, the adjusted Rand index subtracts the effects of a bias that two partitions have for similarity simply by virtue of the number and size of their blocks. For us, this adjustment permits more objective partition comparisons across multiple datasets. We note that the range of the adjusted index now includes values less than 0, but partition pairs that yield these values are, in a manner of speaking, "worse than random" and very seldom occur in practical settings.

Partition comparison continues to be an area of active research, and recent efforts appear drawn toward information theory-based methods. In [106], for example, Vinh et al. adapt a correction to a mutual information-based measure very similar to the adjustment in the Rand index. Nevertheless, the adjusted Rand index is well known and remains in good standing within the current pantheon of clustering comparison schemes.

To conclude this subsection, we should at least mention how we use the adjusted Rand index to measure the sub-behavior identification performance of the BD-IHMM and the SS-BD-IHMM. Given two models inferred by either algorithm, it is impossible to compare the partitions of their respective hidden state sets—the number of instantiated hidden states will often be different, and even if they are the same, individual states will almost certainly not be the same between the models. What is comparable, however, is the implied partition of the training data into separate sub-behaviors. In the notation of prior chapters, the hidden state block, i.e. sub-behavior, associated with a particular observation $\boldsymbol{y}_t$ is z_{v_t}; that is, the block label of the hidden state with which the observation has been matched. These labelings are the partitions we compare with a ground truth sub-behavior labeling using the adjusted Rand index.

6.1.2 Evaluating the predictive power of learned models

To determine how well the SS-BD-IHMM, BD-IHMM, or IHMM can infer a data generating process for a given time series, we adopt a standard machine learning approach: segregate data into training and test sets, infer a model given the data in the training set, and compute the likelihood of the test set with respect to the model. In a certain sense, we imagine ourself measuring to what extent the inferred model predicts that data like the test set data might occur.

As it happens, our test set likelihood computation does not compute the true likelihood of the test set given the training set data. To do so, we would need to find

$$p(\text{test set} \,|\, \text{training set}) = \int_{m \in \text{all models}} p(\text{test set} \,|\, m)\, p(m \,|\, \text{training set})\, dm$$

where "all models" is the set of all SS-BD-IHMM or BD-IHMM models using the modification function and the class of observation models designated for the problem. This marginalization cannot be expressed closed-form and is intractable to estimate numerically. A "next best thing" would be to draw a sample model from the posterior, or $m \sim p(m \,|\, \text{training set})$, but this too is problematic, since (conceptually) m contains a countably infinite number of parameters, and for those we inevitably choose to leave out so that we could represent the sample on a computer, further complicated marginalizations would be needed to account for their contributions to the likelihood. It is true that we do similar marginalizations while we perform model inference, but one thing that we never integrate away in inference is the training set hidden state trajectory $\boldsymbol{v}$. In contrast, likelihood computations for evaluating HMM performance on a test set *do* marginalize away the hidden state trajectory, another computation that would be impossible to perform closed-form in the (SS-)BD-IHMM setting and difficult to estimate numerically.

Thus, to keep things simple, we opt for what we believe is a sensible "third best" approach, which is to draw a sample m from the (SS-)BD-IHMM posterior, but "truncate" this sample to create a finite HMM that contains only the states that were visited by the inferred test set trajectory $\boldsymbol{v}$. The likelihood of the training set is computed with respect to this HMM. Besides being straightforward, we believe this approach reflects the way the SS-BD-IHMM and BD-IHMM might be used in practical settings—as a means of deriving from data a finite HMM with block-structured or even more elaborate dynamics, in settings where the number of necessary hidden states or blocks is not known. Finally, in most SS-BD-IHMM and BD-IHMM posterior samples, the probability of transitioning to any of the unused hidden states is typically fairly small, suggesting that unless the future data contains considerable amounts of data that is dissimilar to what appears in the training set, the visited states will be able to do most of the job of describing new information.

The derivation of the transition matrix $\boldsymbol{\pi}$ of the finite HMM is essentially the same as step 1 of the hidden state trajectory proposal procedure described in §3.4.5. In addition to what is described there, we set transition probability values π_{mn} for all n that are not visited in the inferred trajectory $\boldsymbol{v}$ to zero, then renormalize the rows of $\boldsymbol{\pi}$. Next, we compute the forward probability lattice as described in steps 2 and 3 of the proposal procedure, albeit for the test set, not the training set. This done, the test set likelihood can be computed by simply summing the final row $\boldsymbol{L}_t$ of the lattice.

6.2 Data organization feats of the BD-IHMM

We begin with the BD-IHMM, whose chief novelty over the IHMM is its ability to identify variable numbers of sub-behaviors in time-series data. We consider four different problem settings designed to test this capability in order of increasing difficulty.

6.2.1 Artificial data

We first consider an artificial data setting, where observations are 2-D samples from spherical Gaussian distributions ($\sigma = 1$) scattered on a plane. These distributions are emission models associated with states in a randomly-generated HMM. The HMM has anywhere from 2 to 4 sub-behaviors, and each sub-behavior has anywhere from 3 to 9 hidden states. Each emission model is drawn from a sub-behavior specific spherical Gaussian ($\sigma = 6$), whose mean in turn is drawn from another Gaussian ($\sigma = 4$). Transition probabilities between hidden states favor within-block transitions at a rate of 98%, so hidden state sequences remain in a sub-behavior with a “half life” of around 34 time steps.

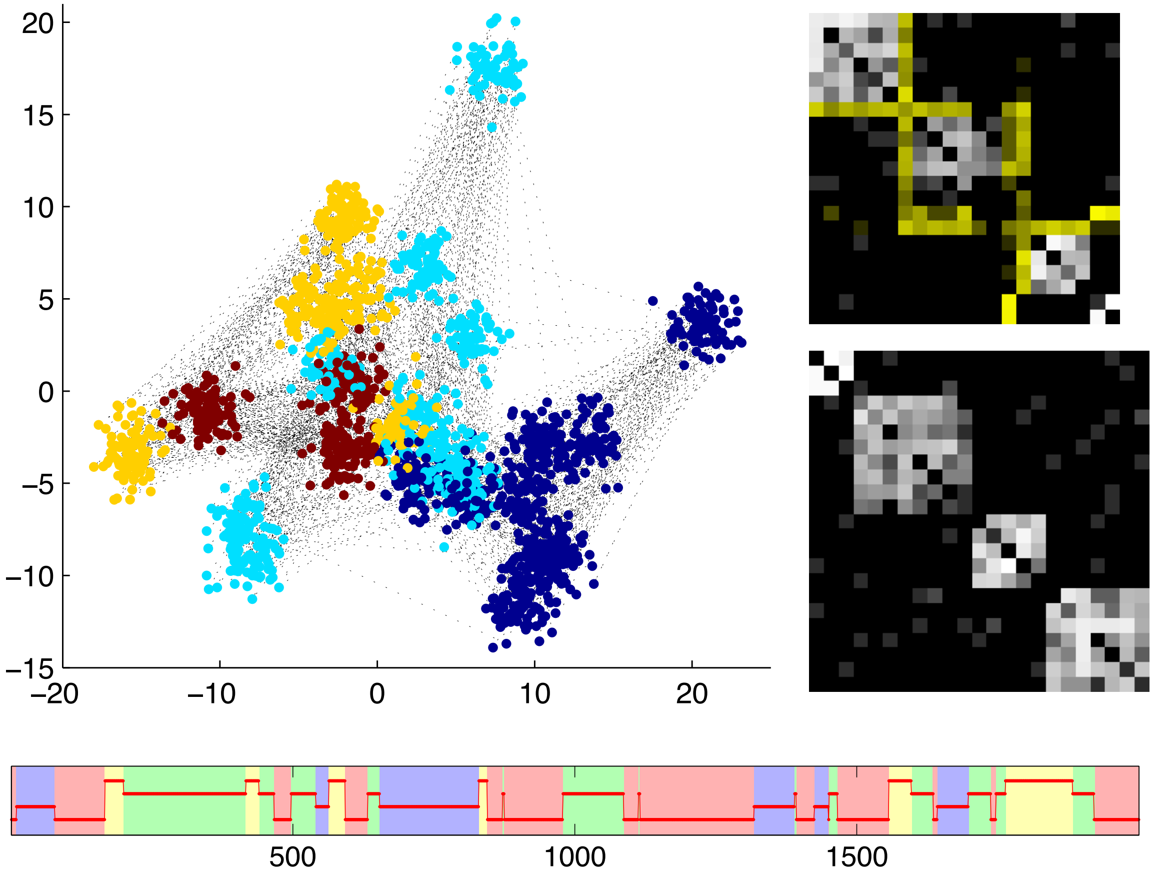


FIGURE 6.2: Top left: synthetic data for the BD-IHMM inference task in §6.2.1; colors mark the four sub-behaviors. At right, visualizations of the transition count matrices inferred for this data by the IHMM (top) and the BD-IHMM (bottom)—by recognizing sub-behaviors, the BD-IHMM correctly identifies separate states that the IHMM conflates (yellow). Below, a timeline relating the sub-behavior inferred for the training set sequence (red line) with the ground truth (background colors).

For each dataset, we draw 2,000 time steps of training data and 2,000 time steps of test-set data, an example of which appears in Figure 6.2. Both the IHMM and the BD-IHMM employ vague Gamma priors on all hyperparameters, along with spherical Gaussian emission models ($\sigma = 1$), the same as were used to create the data. Quantitative results appear in Figure 6.3. In this task, the IHMM and BD-IHMM tend to infer hidden Markov models of comparable quality—test-set log likelihoods are very similar (Wilcoxon signed-rank test, $p = 0.08$). Numerous adjusted Rand index scores near 1, however, indicate that the BD-IHMM is frequently coming up with the correct partitioning of the training data into sub-behaviors, a feat which the IHMM does not even attempt. It seems apparent that this extra capability does not require sacrifices in data prediction performance in this task.

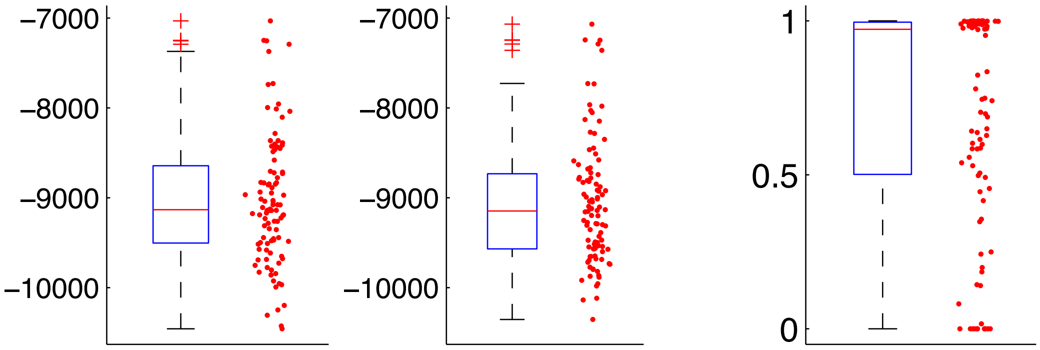


FIGURE 6.3: Left and center: box-and-whisker plots and scatter plots of test-set log likelihoods for (left) the BD-IHMM and (center) the IHMM for 40 runs of the artificial data task described in §6.2.1. Overall, test-set likelihood values are near-identical for both models (Wilcoxon signed-rank test, $p = 0.08$). At right, adjusted Rand index scores comparing sub-behavior labels inferred by the BD-IHMM for the training data with ground truth (c.f. the bottom of Figure 6.2). Over half of the runs yielded scores greater than 0.95. Scores of 0 typically correspond to complete undersegmentations, i.e. all data associated with just one sub-behavior.

Figure 6.2 shows an additional benefit of the BD-IHMM: in some cases, a model that explicitly accounts for sub-behaviors can disambiguate data where the emissions from two or more states are otherwise quite similar. In the example shown, multiple states have emission model distributions with considerable overlap. Only the BD-IHMM is able to identify these individual states, a feat it accomplishes by examining and accounting for differences in their patterns of inbound and outbound transitions.

6.2.2 Video gesture classification

In this task involving more challenging real-world data, we collected multiple video clips of a person executing four distinct gestures as they played a motion-activated video game.[1] The portions of the color video frames containing the person were downscaled to 21 × 19 pixels, then projected onto their first four principal components. The resulting sequence of four-dimensional observations make up the data used for IHMM and BD-IHMM training and testing. Examples of this data appear in Figure 6.4.

[1] *Wii Sports* on the Nintendo Wii.

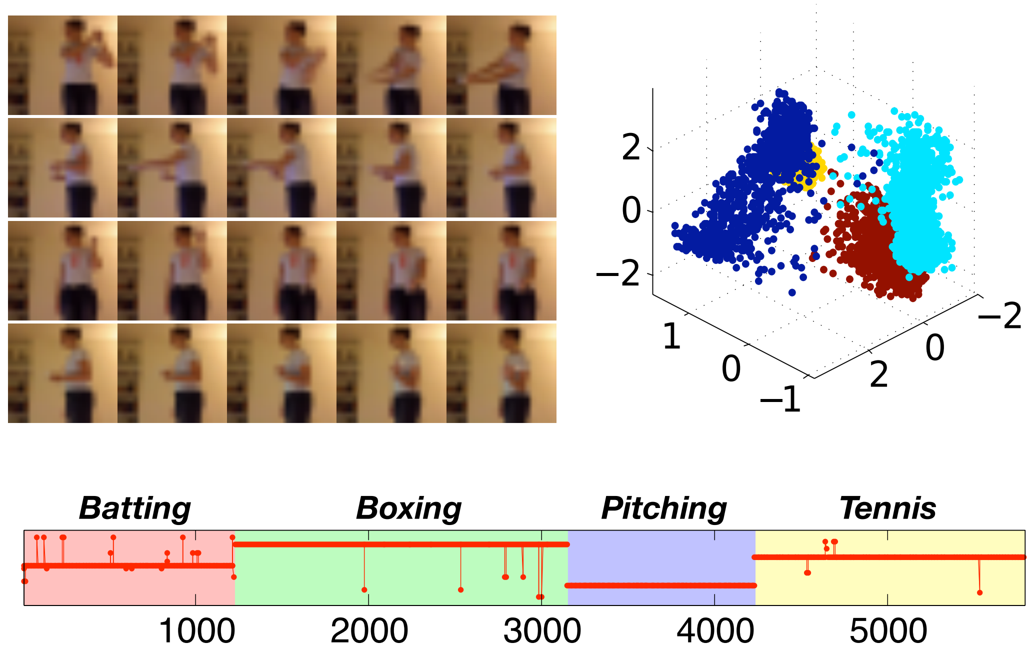


FIGURE 6.4: Top left: selected downscaled video frames for (top to bottom) batting, boxing, pitching, and tennis swing gestures. Top right: first three dimensions of video frames' PCA projections, colored by gesture type. Below: a timeline relating the sub-behavior inferred for the training set sequence (red line) with the ground truth (background colors). Labeled sub-behavior regions actually comprise numerous video clips of the same gesture.

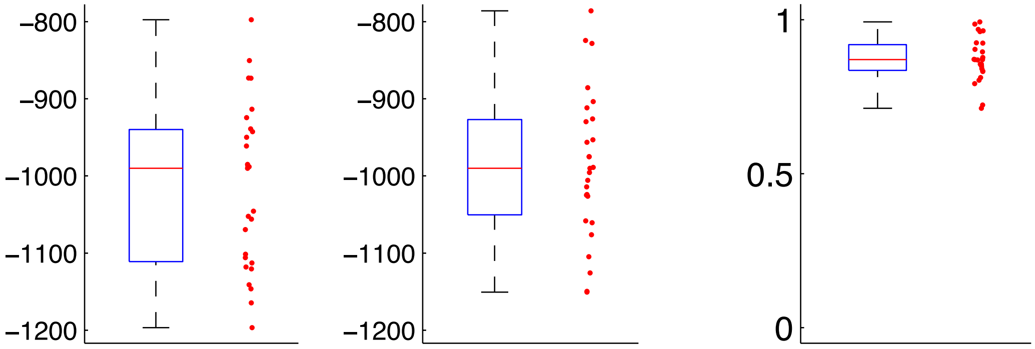


FIGURE 6.5: Left and center: box-and-whisker plots and scatter plots of test-set log likelihoods for (left) the BD-IHMM and (center) the IHMM for 40 runs of the video gesture data task described in §6.2.2. Overall, test-set likelihood values are slightly better for the IHMM (Wilcoxon signed-rank test, $p = 0.02$). At right, favorable adjusted Rand index scores comparing sub-behavior labels inferred for the training data with ground truth (c.f. the bottom of Figure 6.4).

For inference, parameters were similar to the artificial data experiment, except here the emission models were 4-D spherical Gaussians ($\sigma = 0.275$). We repeated a 9-way cross-validation scheme three times to collect results over multiple trials; training sets contained around 6,000 observations. Training-set sub-behavior labeling performance, as measured by comparing the inferred labels against human-generated ground truth and computing the adjusted Rand index, was favorable ($\mu = 0.87$, $\sigma = 0.07$), although IHMM-inferred models yielded slightly better test-set likelihood scores than BD-IHMM models on this task (Wilcoxon signed-rank test, $p = 0.02$). This performance difference is not overwhelmingly large: the mean ratio of BD-IHMM log likelihoods to IHMM log likelihoods over all of the experimental runs is 1.03 ($\sigma = 0.06$).

This task offers a convenient illustration of how sub-behaviors consist of multiple states; in most settings, individual states are too specialized for a specific range of observations to describe an entire sub-behavior on their own. For the video gesture data, both the BD-IHMM and the IHMM allocated around 45 hidden states to describe the training data (combined mean: 44.5, $\sigma = 5.0$); adjusted Rand indices comparing the IHMM's inferred trajectory through the states ($\mathbf{v}$) to the ground truth sub-behavior labeling are poor ($\mu = 0.28$, $\sigma = 0.036$).

6.2.3 A torture test: musical theme discovery in note-based data

The next task considers time series intended to push the limits of the BD-IHMM's sub-behavior identification ability. They contain fewer time steps than the tasks we've considered so far: this limits the statistical evidence for sub-behaviors. Furthermore, the progressions between states within sub-behaviors are highly regular, making the structure of the transition counts matrices—the chief statistic for sub-behavior discovery in the BD-IHMM—not very block-diagonal at all.

This task involves the identification of repeated themes in music, a well-known kind of time series. Western popular music often features repeated musical themes, usually corresponding to verses, choruses, and other structural elements of songs. In BD-IHMM terms, these themes are the sub-behaviors that we attempt to discover through inference. As hinted above, the discovery of this structure from the statistics of time step-to-time step hidden state succession is a deliberate challenge: purpose-built algorithms for theme discovery have access to autocorrelation measurements and other non-Markov statistics, along with specially-designed musical feature detectors which we do not employ [107].

¶ **Data and data preparation** — We begin with some background. There are two very broad strategies for representing music with computers. The first is "low-level": the computer stores something close to the actual signal that would be sent to the speakers to create sound. This strategy, used by WAV, MP3, and other popular formats, can represent sounds with very high fidelity, since one can choose to use as many samples or frequencies as they like to preserve the original character and nuance of the audio. Any kind of sound works with these representations—they are used for spoken dialogue as well as music, among other things. The elements of these representations, however—the individual samples or frequencies—have very little semantic value of their own. A good representation of a single piano note, for example, may use hundreds of superimposed frequencies or thousands of samples.

The second strategy does away with representations of actual sounds and instead codes individual "large-scale" audio events at specific times during audio playback. The most common realization of this strategy is MIDI, a representation whose elements denote events like "45.3 seconds into the song, a tuba plays an E♭ with loudness 5 for 2.2 seconds." Because they mean roughly the same thing as notes in a musical score, these elements have a much richer semantics than those of the first strategy. MIDI does not represent, however, the precise sound of specific notes—instead, it is assumed that the computer or electronic instrument knows how to generate the required sounds corresponding to high-level notions like *tuba*. For this reason, MIDI is generally not suitable for musical recordings, and it is mostly useless for representing non-musical information like speech. As a rough electronic analog to sheet music or player piano rolls, MIDI usually sees use as a means of controlling electronic instruments, from common devices in personal computers to expensive keyboards, sequencers, and drum sets used by performing artists.

In the following demonstration, we use the BD-IHMM to investigate structure in a collection of MIDI files. Table 6.1 lists the MIDI files used in all of the experiments in this section, which were chosen for their familiarity and discernible thematic variation. Each of these files were imported into MATLAB with the MATLAB MIDI toolbox developed by the University of Jyväskylä [108]. The note data was then converted from the toolbox representation to a "piano-roll" representation shown graphically in Figure 6.6. This representation describes the song as a sequence of binary vectors, each representing a two-beat interval in the music. Individual bits in the vectors correspond to unique notes played by individual synthesized instruments—thus, for example, a piano playing an E♭, a piano playing a D, and a French horn playing an E♭ would all be represented by different bits. A bit is set on if the corresponding note is sounding anytime in the interval represented

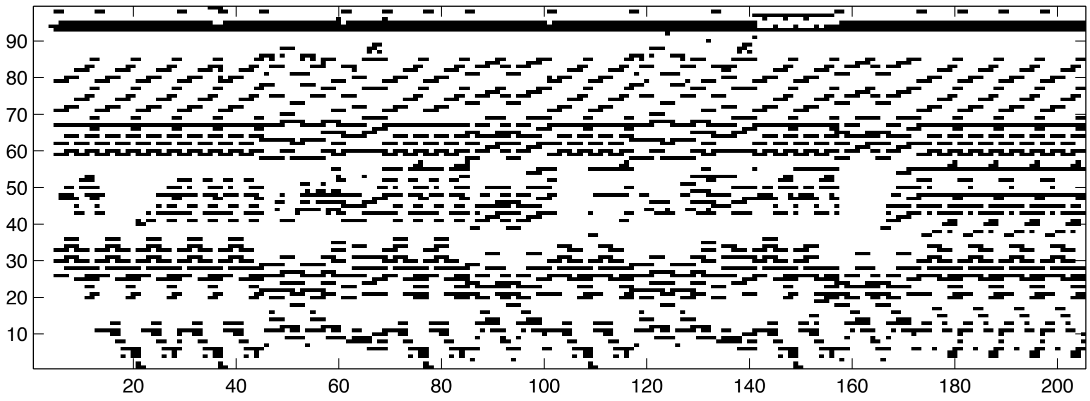


FIGURE 6.6: "Piano roll" representation of the MIDI file for *Uptown Girl* by Billy Joel. Each row represents a unique pitch played by one of several synthesized musical instruments; each column represents to a half-beat interval in the song. A row/column cell in the piano roll is "on" (black) if the corresponding instrument is playing a note in the time represented by the column. Displayed in this way, *Uptown Girl* shows hints of its thematic structure. Note that this entire sequence has only 205 time steps.

Artist	Song	Duration	Checksum
Billy Joel	*Uptown Girl*	3:24	60738
Cher	*Believe*	2:36	18601
Enya	*Orinoco Flow (Sail Away)*	4:20	21137
Harry Chapin	*Cat's in the Cradle*	4:00	62131
Johnny Cash	*Ring of Fire*	2:36	702
Meredith Brooks	*Bitch*	4:14	31548
Paul Simon	*You Can Call Me Al*	2:20	43621
Queen	*We Are the Champions*	2:44	9440
Rolling Stones	*Paint It, Black*	2:59	36020

TABLE 6.1: MIDI files used in the demonstration of §6.2.3. Because these files appear under different names in many different online MIDI file archives, the rightmost two columns list the duration and the BSD UNIX checksum of the file (as reported by the `sum` command) for identification purposes.

by the vector. For each song, we only employ bits corresponding to notes that are actually played in the song—that is, each row has at least one bit on, and rows corresponding to unused instruments or notes are discarded from the final song representation.

¶ **Observation models** — To adapt BD-IHMM techniques to the task of modeling "piano roll" note sequences like the one shown in Figure 6.6, we must select an observation model for states capable of generating bit vectors like the ones that make up the piano roll columns. The simplest such model is the "naive Bayes" approach, which dictates that the probability of any bit being on is conditionally independent of the state of other bits given knowledge of the state emitting the observation. Let $\boldsymbol{\theta}_m$ be the parameters of the observation model associated with the mth hidden state. These parameters are the probabilities of the bits in observations drawn from the model being on. In accordance with established notation, we restate this as

$$y_{t,i} \sim \text{Bernoulli}(\theta_{v_t,i}), \tag{6.1}$$

where v_t identifies the hidden state associated with the tth time step of the data sequence. The prior for observation models is even simpler:

$$\theta_{m,i} \sim \begin{cases} 0.99 & \text{with probability } 0.5, \\ 0.01 & \text{with probability } 0.5. \end{cases} \tag{6.2}$$

Restricting $\theta_{m,i}$ values to 0.99 or 0.01 yields a high but not absolute correlation between bits in the columns in the data ($\boldsymbol{y}_t$) and the observation models to which they are assigned ($\boldsymbol{\theta}_{v_t}$). This in turn motivates the allocation of many states to describe the data, since it is difficult for any one state to generalize over extended portions of a song.

¶ **Evaluation notes** — In contrast with most of the experiments and demonstrations in this chapter, this music task and the one in the next subsection lack bona-fide ground truth. Instead, we created our own collection of labels for the music by listening to the MIDI songs and labeling the data by hand according to what we perceived to be distinct themes. This judgment is naturally subjective, and despite our good intentions, we experimenters cannot be trusted to act without bias in our labeling—hence, it may be best to consider this task a “demonstration” rather than an “experiment”. Nevertheless, we expect that the vast majority of our labeling choices would not be controversial. Most of the time, our labels reflect the verse/chorus/bridge structure of the songs, but in some pieces (e.g. *Uptown Girl*, with a highly modulated melody complemented by a rich and dynamic polyphonic harmony) additional labels mark times when the music just started to “sound different” to our avid but untrained ear. For this same reason, lengthy transitional interludes between different sections of some songs sometimes received their own labels as well.

In any case, it is this labeling that we compare with the block labeling implied by the inferred hidden state trajectory, using the adjusted Rand index (§6.1.1).

¶ **Results** — Overall adjusted Rand index performance on the nine songs in Table 6.1 appears in Figure 6.7. The scores reflect the difficulty of the theme discovery task; this

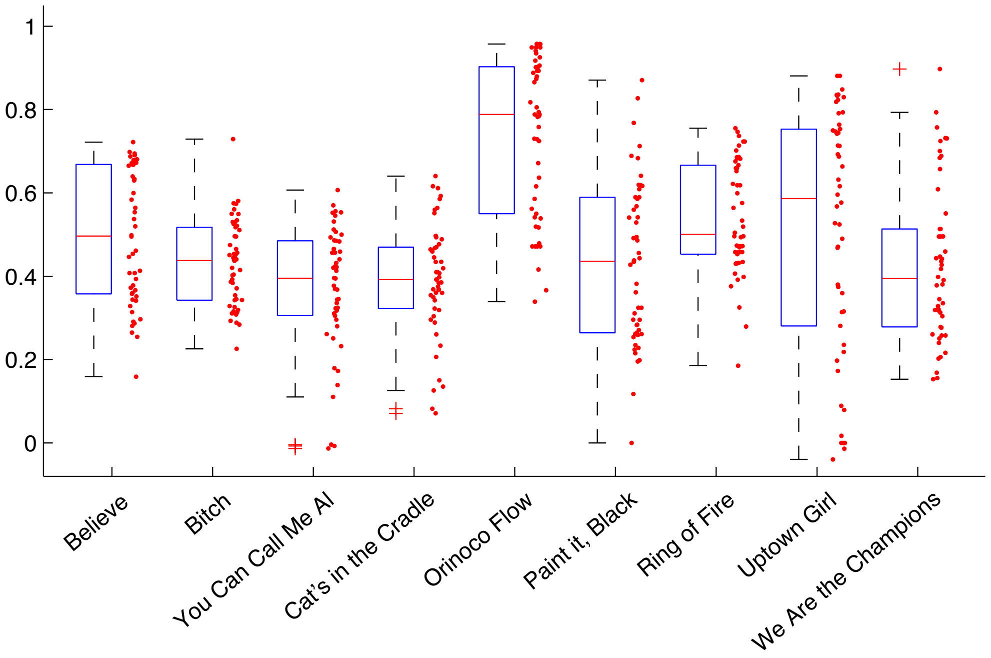


FIGURE 6.7: Box plots and scatter plots of adjusted Rand index scores for all nine songs listed in Table 6.1. Scores are compiled from 50 runs of BD-IHMM inference for each song. The mean adjusted Rand index over all tasks was 0.48 ($\sigma = 0.21$).

said, it is evident that the BD-IHMM was able to discover meaningful structure in the data.

To make more sense of the scores, however, we examine individual labeling results in Figure 6.8. Here we find that although there is room for improvement in the adjusted Rand index scoring, in many cases the BD-IHMM is managing to identify sensible structure in the data. Occasionally, this structure does not square with our musically-informed labeling, but nevertheless reflects significant changes in the MIDI song data—for example, the "incorrect" sub-behavior transition around $t = 18$ in *You Can Call Me Al* corresponds to the injection of arpeggiated bass chords, and hence motivate the allocation of a completely new set of BD-IHMM hidden states. Overpartitioning of the hidden states into too many sub-behaviors is a problem in these results, but may be understandable given the limited amount of data in each song.

To underscore this point, Figure 6.9 shows the kinds of transition counts matrices inferred by the BD-IHMM for this task. Ordinarily, for tasks like these where transitions between sub-behaviors are seldom, block structure is at least somewhat apparent when the matrix rows are shuffled to appear in the order in which their corresponding states were first

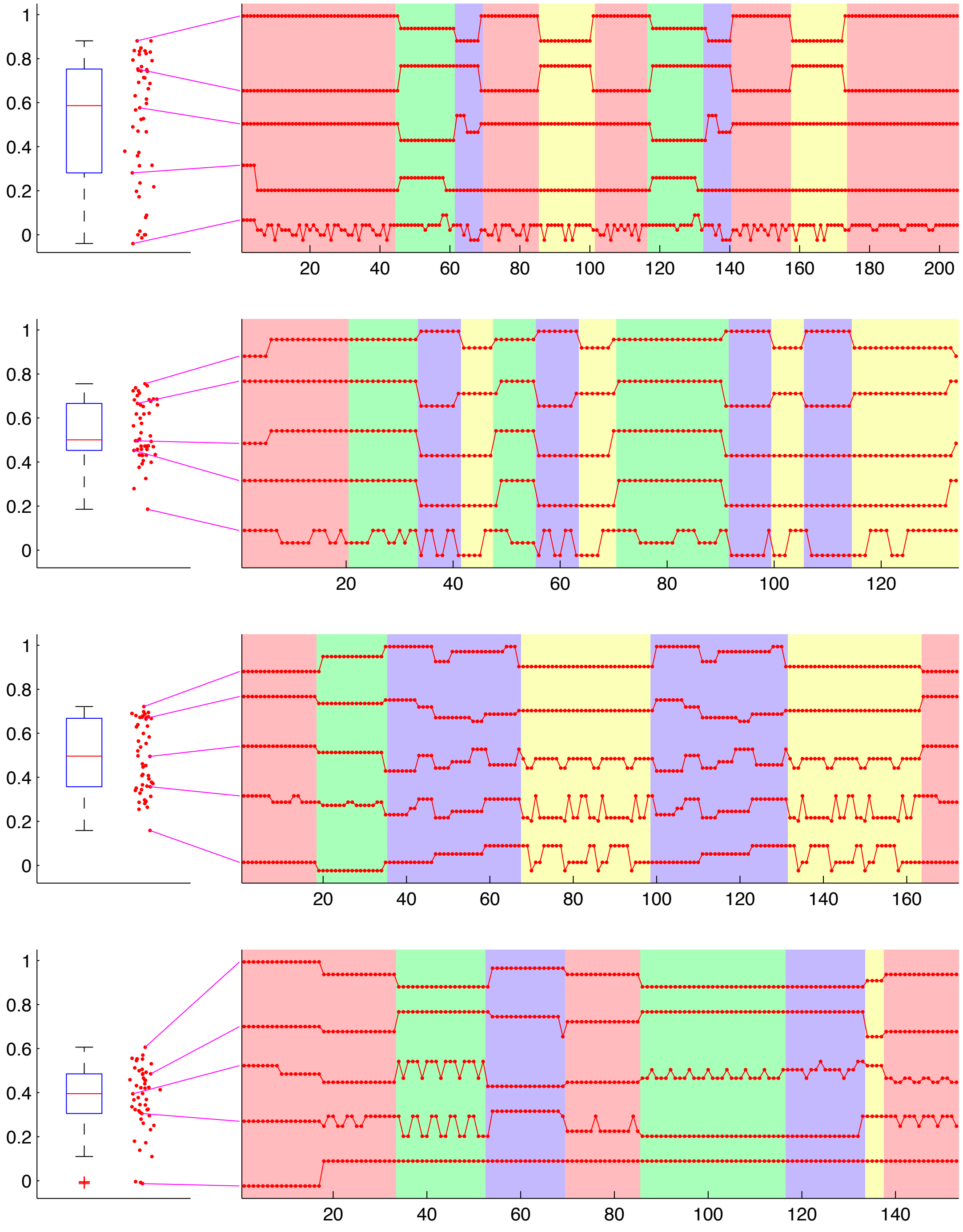


FIGURE 6.8: Examples of musical theme labelings for (top to bottom) *Uptown Girl*, *Ring of Fire*, *Bitch*, and *You Can Call Me Al*. The latter two songs were more difficult for the BD-IHMM. Examples show the best, worst, and 25th, 50th, and 75th percentile labelings for each song as determined by the adjusted Rand index; red lines correspond to inferred labelings, while background colors indicate ground truth. In several cases, labelings with modest adjusted Rand index scores still identify meaningful structure in the song.

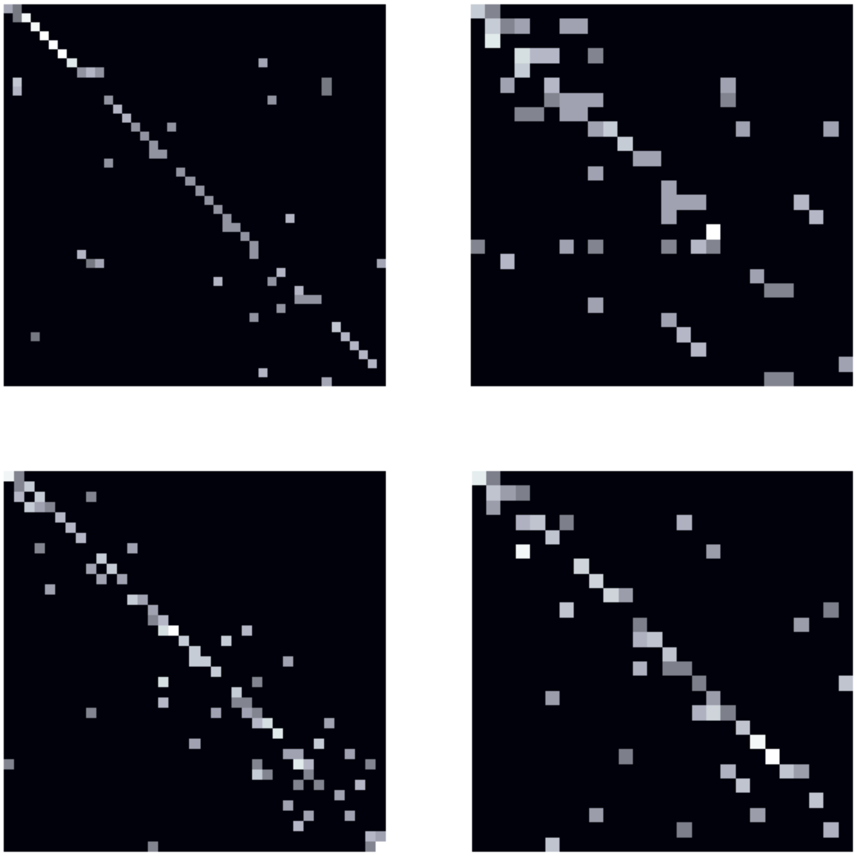


FIGURE 6.9: Transition counts matrices sampled by one of the BD-IHMM inference runs for the songs (top left) *Uptown Girl*, (top right) *Ring of Fire*, (bottom left) *Bitch*, or (bottom right) *You Can Call Me Al*. Rows are permuted to be in the order in which states are first encountered in the inferred hidden state trajectory, an ordering usually conducive to revealing block structure in counts matrices. There is not very much block structure to see here.

visited in the hidden state trajectory. The very sparse matrices in the figure show little block structure indeed—compare with the considerably more robust structure apparent in Figure 6.2.

In data-poor circumstances like these, sampling-based approaches to BD-IHMM inference like the one evolved in this thesis have an additional utility. More than one sub-behavior labeling might be plausibly applied to the data; multiple BD-IHMM posterior samples provide the analyst with a range of such labelings.

As a final note, the BD-IHMM conference paper explores application of BD-IHMM to musical theme discovery with audio-based representations of musical data [109]. Results are slightly worse than those shown here, reflecting the additional challenge of distilling the semantically relevant musical information from recorded audio samples.

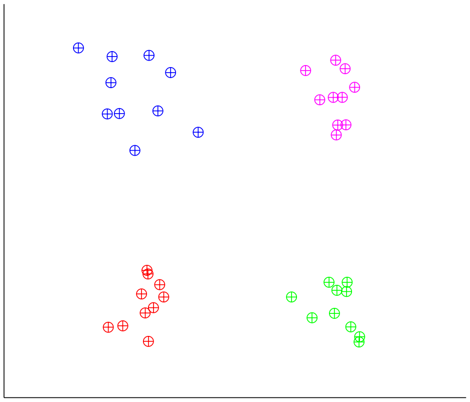
6.3 New capabilities: the SS-BD-IHMM

The previous section has demonstrated the BD-IHMM's ability to identify sub-behavior structure in time series. Chapter 5 introduces the SS-BD-IHMM, a generalization of the BD-IHMM with two improvements inspired by the problem of learning view-based visual object models from video data. In this section we consider two artificial tasks designed to highlight each of these improvements by themselves. In both problems, the new capabilities of the SS-BD-IHMM yield significant improvements in partitioning the training set data and predicting the test set data.

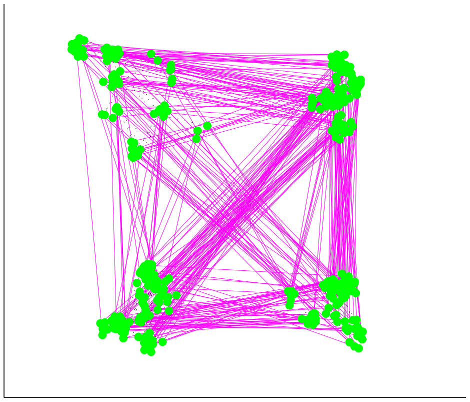
6.3.1 "Four mounds"

Our first task offers a basic exploration of how SS-BD-IHMM inference can use appearance similarity to arrange hidden states into groups. This capability was motivated in §5.1.2 by situations in which an object has been viewed from at least two separate sets of viewpoints, but thus far there has been no meaningful visual observation of the object transitioning between these two sets of viewpoints. The BD-IHMM would have no reason to associate these separate viewpoints with the same object, but the SS-BD-IHMM can use appearance similarity cues to determine that they are in fact experiences of the same thing.

This appearance-based grouping is supplemental to the way in which both BD-IHMM and SS-BD-IHMM inference use the inferred transitions between hidden states to determine likely groupings. In fact, for this task, transition dynamics offer no clues about which hidden states belong together. We consider a circumstance where observations are 2-D real vectors drawn from spherical multivariate normal observation models associated with a Markov process operating over 40 hidden states. These observation models are arranged in four separate clusters as shown in the example below, hence the mildly more vivid "four mounds" name used for this section:



Ten hidden states appear in each cluster. Transition probabilities between each of the hidden states do not reflect the clustering arrangement. Below, the green dots show a sequence of 400 observations sampled from the same example model whose observation model means are shown above. Both magenta and black dotted lines connect the samples in sequence:



In the above, magenta lines mark transitions between hidden states in different clusters, while dotted black lines mark transitions between states in the same cluster. The latter are difficult to make out, partly due to the nearness of states in the same cluster and partly due to the considerable number of inter-cluster transitions in the data. As hinted earlier, the dynamic structure in this data is utterly independent of the cluster membership of the hidden states, so the frequent inter-cluster transitions is not surprising.

This subsection uses the word “cluster” to refer to semantically useful groupings of hidden states. The term “block”, used in most other parts of the thesis, means exactly the same thing. We use the new term here to emphasize how the primary evidence for grouping in this section—the spatial clustering of hidden states—is different from the dynamics-based evidence we’ve considered in the past.

As an analogy for object learning, this task could be said to represent an extreme case of the imagined circumstance at the beginning of this subsection, where no meaningful transition is observed between collections of views of an object. Each 2-D observation gives a glimpse of what one of the “mounds” (c.f. objects) is like, but since the next observation is equally likely to come from any mound, the succession of inferred “mound views” (c.v. object views) is uninformative about the underlying grouping arrangement. In this sense, there are no meaningful transitions to observe. Four mounds datasets are thus akin to viewing a randomly shuffled collection of pictures of four different objects—the sequence of the photos offers no information about which photos belong to the same object.

Since the BD-IHMM can only use dynamics-based evidence to identify hidden state groupings, it is not surprising to learn that this model never manages to correctly identify the four hidden state clusters in “four mounds” datasets. Nevertheless, there is another point in comparing the BD-IHMM and the SS-BD-IHMM at this task. By expressing a more structure-rich prior on the observation model parameters—the only functional difference between the BD-IHMM and the SS-BD-IHMM in this task—the SS-BD-IHMM may be able to infer observation models that do a superior job of describing test set data. We evaluate this hypothesis in the following experiment.

¶ **Data generation** — The artificial data generating process for this task involves sampling a Markov model, then sampling a sequence of observations from that model. The Markov model has 40 states, ten in each block. Observation model means for the first ten states are drawn from a spherical multivariate normal distribution ($\sigma = 5$) with mean $[0,0]^\top$. Observation model means for the next three blocks of states are drawn from this same distribution shifted to $[0,40]^\top$, $[40,0]^\top$, and $[40,40]^\top$ respectively. As mentioned before, observation models themselves are unit spherical multivariate normal distributions centered at these sampled means.

To generate the 40 rows $\boldsymbol{\pi}_m$ of the transition matrix $\boldsymbol{\pi}$, we draw from a Dirichlet distribution with uniform parameters:

$$\boldsymbol{\pi}_m \sim \mathrm{Dir}\left(\frac{\alpha}{40}, \frac{\alpha}{40}, \ldots, \frac{\alpha}{40}\right),$$

with $\alpha = 4$ for this experiment.

Finally, we draw hidden state trajectories and observations in the usual way for HMMs given a transition matrix and observation models. Each experimental run executed this entire generative procedure anew to control for variation in artificially generated data.

¶ **Evaluation methods** — We compare BD-IHMM and SS-BD-IHMM inference results in two ways: by computing the likelihood of test set data given a finite HMM derived from posterior SS-BD-IHMM and BD-IHMM samples, and by observing how well both models segregate hidden states into blocks. For this problem, the hidden state partitioning results are clear enough to obviate the need for sophisticated analytical techniques.

¶ **Experiment** — We compare the SS-BD-IHMM and BD-IHMM on 264 separate datasets.[2] For each dataset, we sampled transition matrices using the above procedure and generated a sequence of 11,000 observations. The first 1,000 observations were used as training set data; the final 10,000 were used to evaluate trained models in the manner described above. For the BD-IHMM and SS-BD-IHMM, our inference sampled means for unit spherical multivariate normal observation models. The SS-BD-IHMM also sampled means for spherical multivariate normal observation model generating distributions ($\sigma = 5$). For these, we specified a spherical multivariate normal prior centered at $[20, 20]^\top$, $\sigma = 100$. In the BD-IHMM, meanwhile, all observation models were drawn from a fixed, shared, vague spherical multivariate normal observation model generating distribution centered at $[20, 20]^\top$, $\sigma = \sqrt{100^2 + 5^2}$. This σ parameter comes from convolving the two normal distributions involved in SS-BD-IHMM observation model generation, which we believe yields the fairest comparison between both models; this said, both 100 and $\sqrt{100^2 + 5^2}$ are really quite close to one another.

¶ **Results** — The test-set likelihood results for each of the 264 datasets appear in Figure 6.10. A majority of runs (157 out of 264) show a higher test-set likelihood for models inferred by the SS-BD-IHMM than by the BD-IHMM. This improvement is statistically significant (Wilcoxon signed-rank test, $p = 1.4 \times 10^{-5}$), albeit faint: the mean ratio of BD-IHMM to SS-BD-IHMM test-set log likelihood scores is 1.003 ($\sigma = 0.011$); the median ratio is only slightly smaller. Nevertheless, it seems safe to assume that overall, the more elaborate observation model generation structure in the SS-BD-IHMM can infer slightly more accurate models than the BD-IHMM can.

[2] We had originally planned to use fewer datasets, but added additional runs to achieve statistical significance in the test-set likelihood comparison. There is no significance to our using 264 additional runs, other than this was the number completed when we examined the new results.

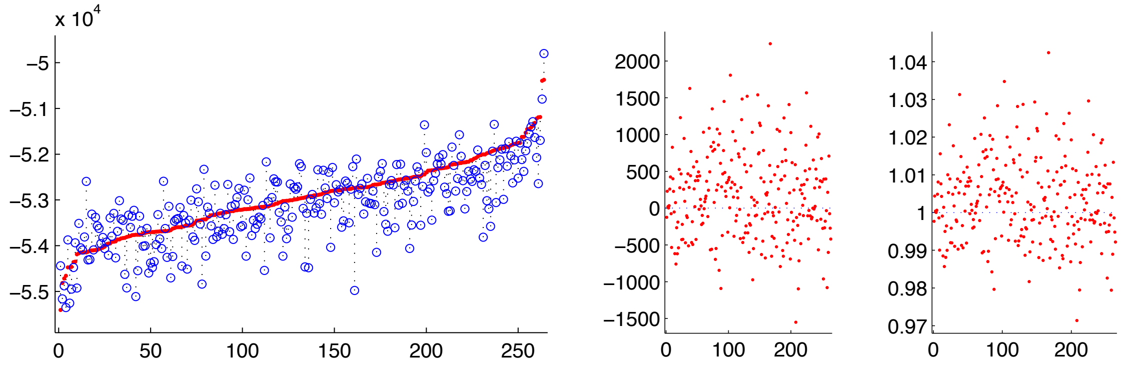


FIGURE 6.10: Test-set log likelihood results for the 264-dataset "four mounds" experiment of §6.3.1. At left, test-set log likelihoods for the SS-BD-IHMM (red dots) and the BD-IHMM (blue circles), sorted in increasing order of SS-BD-IHMM log likelihood for clarity. Black dotted lines connect SS-BD-IHMM and BD-IHMM results for the same dataset. At center, the difference between the SS-BD-IHMM and the BD-IHMM test-set log likelihood values for the 264 datasets (unsorted). Values above zero indicate better performance for the SS-BD-IHMM. At right, the ratio of BD-IHMM test-set log likelihood values to those from the SS-BD-IHMM for the 264 datasets (unsorted). Values above 1.0 indicate better performance for the SS-BD-IHMM.

Figure 6.11 shows histograms of the number of hidden state partitions inferred by the SS-BD-IHMM and the BD-IHMM for the 264 experimental datasets. A visual inspection of the individual SS-BD-IHMM partitionings (not shown) reveal that those identifying four clusters of hidden states did so by correctly associating each partition with one of the four mounds; by contrast, those few BD-IHMM samples with more than one partition showed no meaningful relationship between partition structure and the hidden state mound arrangement.

So far, the description of this experiment has not mentioned the base measure modification function specified for the SS-BD-IHMM prior. We did specify a modification function for this task—in fact, the same one used in the next experiment (§6.3.2)—but since there is no structure in the hidden state transitions, we should expect the modification function to be deemed more or less irrelevant during inference. Satisfyingly, this seems to be the case: the median ξ value inferred for SS-BD-IHMM models is 0.001, while the largest is 0.15.

6.3.2 "Twin waffles"

Our second task is designed to show benefits from using the more general base measure modification in the SS-BD-IHMM to closely match the dynamic structure of a particular problem setting. We consider a circumstance where observations are 2-D real vectors

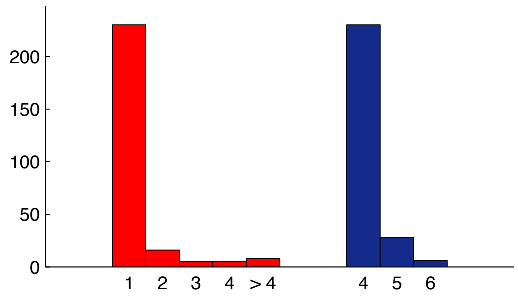
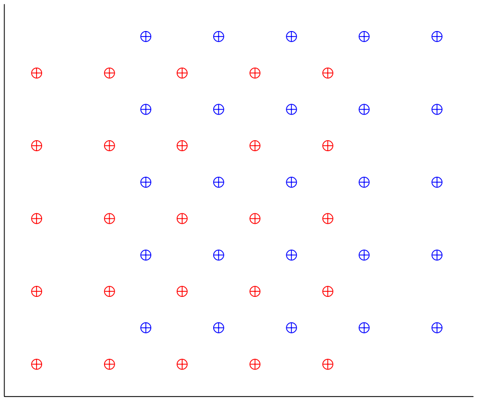
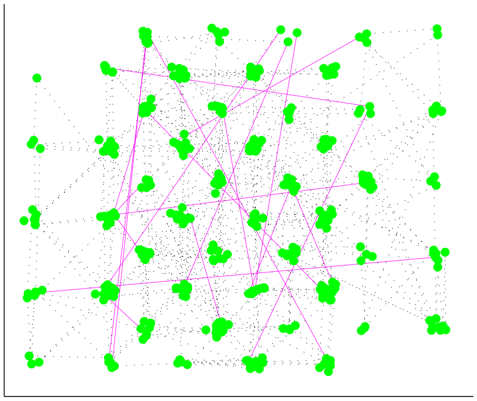


FIGURE 6.11: Histograms of the number of hidden state partitions inferred for the 264 "four mounds" experiment (§6.3.1) datasets for BD-IHMM (left, red) and SS-BD-IHMM (right, blue) models. The correct number of partitions is four; the BD-IHMM typically "undersegments" the hidden states since it cannot use the spatial layout of the states to make partition decisions.

drawn from two grid-like arrangements of spherical multivariate unit normal distributions whose means are laid out in the pattern shown below:



States whose observation model means are red belong to one block; those with blue means belong to the other. The two grid-like patterns inspire the name "twin waffles" for this task. Below, the green dots show a sequence of 400 observations sampled from the model. Both magenta and black dotted lines connect the samples in sequence:



Here, black dotted lines show transitions between states in the same block; magenta lines show transitions between the two blocks. The dotted black lines are much more numerous, indicating that the process generating the data does indeed have block structured dynamics. What may be less obvious is that within-block transitions tend to occur between nearby states; jumps from one side of a "waffle" to another are very rare. Thus, in addition to the block-structured dynamics just mentioned, the within-block transitions exhibit a proximity-based dynamic structure. The SS-BD-IHMM can express this new structure; the BD-IHMM cannot.

In terms of object model inference, the point of this experiment is to test a prior for HMMs that explicitly encodes the smoothness property described in the beginning of Chapter 5, which we related there to the Gestalt principle of continuity. The fact that an observation on one side of a "waffle" seldom follows an observation on the other side resembles the way our visual perception of an object is also continuous—we seldom see an object rotate 180° or undergo other drastic appearance changes in an instant. Nevertheless, our visual system's comprehension of how objects change over time allows us to derive a unified understanding of how an object looks, even when two object appearances are usually separated in time by numerous intermediate appearances. We shall see that a similar comprehension is necessary to infer an accurate model of "twin waffles" data.

In the last section, we learned that the more elaborate block-specific observation model generating distribution scheme in the SS-BD-IHMM can confer a slight advantage in model inference over the simpler approach used in the BD-IHMM. To control for this effect and evaluate the benefits of added structure in transition probability priors exclusively, we introduce in this section the BD-IHMM*, an augmentation of the BD-IHMM that adds the SS-BD-IHMM's block-specific observation model generating distributions to the BD-IHMM, but leaves the transition structure prior the same. Equivalently, the BD-IHMM* can be said to be an instantiation of the SS-BD-IHMM with a very simple similarity function:

$$\mathrm{Sm}(\theta_m; \theta_n) = 1.$$

Despite of this generous gesture to the SS-BD-IHMM's competitor, the substantial overlap between the "waffles" make it unlikely that block-specific observation model generating distributions will be especially helpful to either approach.

¶ **Data generation** — Let us briefly specify the artificial data generating process for this task. There are 50 states in total, 25 in each block. Their observation models are spherical unit multivariate normal distributions. Their means are provided explicitly: for the first 25 states,

$$\theta_i = \begin{bmatrix} \theta_{i,x} \\ \theta_{i,y} \end{bmatrix} = \begin{bmatrix} 15((i-1) \bmod 5) \\ 15\lfloor (i-1)/5 \rfloor \end{bmatrix}.$$

The second 25 states are shifted copies of the first 25:

$$\theta_i = \begin{bmatrix} \theta_{i,x} + 1.5 \cdot 15 \\ \theta_{i,y} + 0.5 \cdot 15 \end{bmatrix}.$$

Block labels z assign hidden states to the two blocks as described above:

$$z_i = \begin{cases} 1 & \text{if } i \leq 25, \\ 2 & \text{if } i > 25. \end{cases}$$

Subsequently, we sample a transition matrix for transitions between the hidden states using a procedure that resembles the structure of an SS-BD-IHMM generative process. We start with a uniform $\boldsymbol{\beta}$ vector: $\beta_1 = \beta_2 = \ldots = \beta_{50} = 1/50$, then specify a modification function based on the proximity of argument observation model means:

$$\mathrm{Sm}(\theta_n; \theta_m) = \exp\left(-\frac{1}{2} \frac{(\theta_{n,x} - \theta_{m,x})^2 + (\theta_{n,y} - \theta_{m,y})^2}{18^2} \right). \tag{6.3}$$

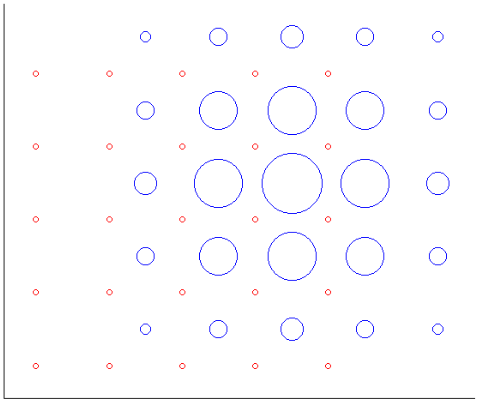
Probabilities $\boldsymbol{\pi}_m$ for transitions emanating from row m are drawn as

$$\boldsymbol{\pi}_m \sim \mathrm{Dir}(\alpha_0 \beta^*_{m1}, \alpha_0 \beta^*_{m2}, \ldots, \alpha_0 \beta^*_{m50}),$$

where the modified $\boldsymbol{\beta}^*_m$ values are computed as

$$\beta^*_{mn} = \frac{\beta_n}{1+\xi}\left(1 + \frac{\xi\,\mathrm{Sm}(\theta_n;\theta_m)\delta(z_m = z_n)}{\sum_{j=1}^{50}\beta_j\,\mathrm{Sm}(\theta_j;\theta_m)\delta(z_m = z_j)}\right).$$

We use parameter values $\alpha_0 = 4$ and $\xi = 10$ when generating data for our experiments. This graphic depicts the modified $\boldsymbol{\beta}^*_{38}$ vector for transitions out of state 38 (the center of the upper right "waffle"); larger circles correspond to larger values in the $\boldsymbol{\beta}^*_{38}$ vector. As described, nearby states within the same block are favored.



Finally, hidden state trajectories and observations are drawn in the usual way for HMMs given $\boldsymbol{\pi}$ and $\boldsymbol{\theta}$. Each experimental run executes this entire generative procedure anew to control for variation in artificially generated data.

We stress that the variables in the expressions listed above are used strictly to generate the input data for experiments and are not supplied to the BD-IHMM* or SS-BD-IHMM inference machinery in any way. This said, the similarity function $\mathrm{Sm}(\cdot;\cdot)$ in Equation 6.3 is considered to be known *a priori* and is therefore used directly within the SS-BD-IHMM. This thesis does not explore methods for inferring attributes of SM-IHMM modification functions; we defer study of this worthwhile capability to future work.

¶ **Evaluation methods** — We employ the same evaluation methods used in the prior experiment (§6.3.1).

¶ **Experiment** — We employ the same multiple dataset, 1,000 time step training set/10,000 time step test set experimental schedule used in the prior experiment (§6.3.1). In this case, we achieved statistical significance with 40 datasets.

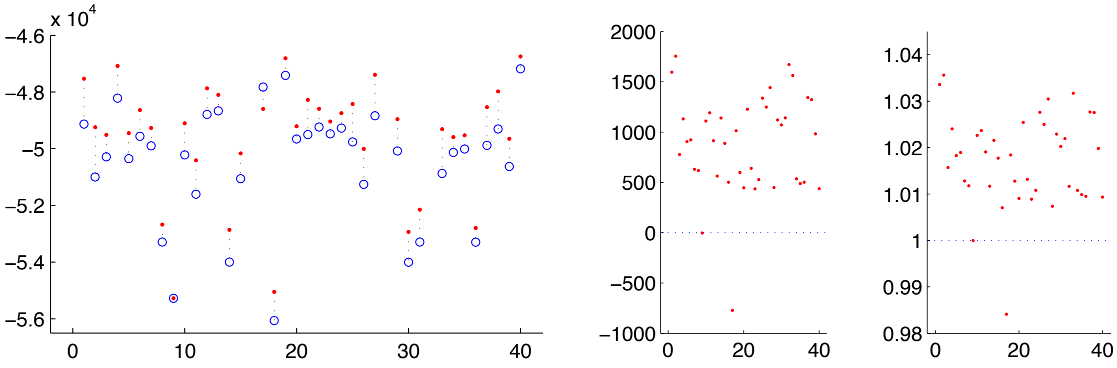


FIGURE 6.12: Test-set log likelihood results for the 40-dataset "twin waffles" experiment of §6.3.2. At left, test-set log likelihoods for the SS-BD-IHMM (red dots) and the BD-IHMM* (blue circles). Black dotted lines connect SS-BD-IHMM and BD-IHMM* results for the same dataset. Two results for "difficult" datasets with very low log likelihoods are not shown, which prevents awkward graph scaling; no results are missing from the other two plots. At center, the difference between the SS-BD-IHMM and the BD-IHMM* test-set log likelihood values for the 40 datasets. Values above the dotted blue zero line indicate better performance for the SS-BD-IHMM. At right, the ratio of BD-IHMM* test-set log likelihood values to those from the SS-BD-IHMM for the 40 datasets. Values above the dotted blue 1.0 line indicate better performance for the SS-BD-IHMM.

¶ **Results** — The test-set likelihood results for each of the 40 datasets appear in Figure 6.12. For all but two, the SS-BD-IHMM performed better, that is, the test-set likelihood was higher. Perhaps unsurprisingly, this improvement is statistically significant (Wilcoxon signed-rank test, $p = 3.8 \times 10^{-10}$).

Given the similarities between this task's data generating process and the internals of the SS-BD-IHMM, perhaps it is no great shock that this model performs well in comparison to the BD-IHMM*. What may be more surprising is that both models diagnose different types of block structure in the data. As shown in Figure 6.13, the SS-BD-IHMM tends to infer the correct structure, while the BD-IHMM* oversegments the blocks—it partitions the hidden states into more than two sets. There is a sensible reason for this: breaking the "waffles" into distinct subregions is the closest thing the BD-IHMM* can do to creating a model with a genuine tendency for transitions to occur between states with near-adjacent observation models. Nevertheless, these oversegmented blocks are not the partitioning answer we are looking for.

Figure 6.14 shows histograms of the number of hidden state partitions inferred for SS-BD-IHMM and BD-IHMM* models. The SS-BD-IHMM rarely fails to produce the correct hidden state partition; all of the SS-BD-IHMM posterior samples that allocated two blocks for inferred hidden states inferred the "twin waffles" arrangement used to generate the

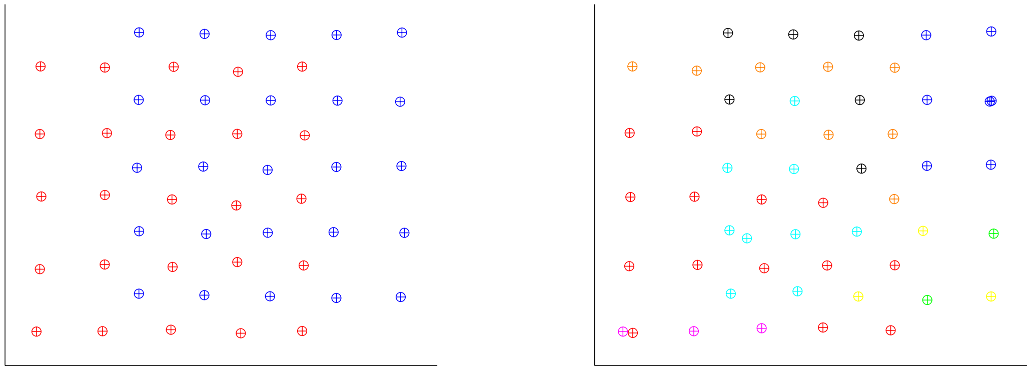


FIGURE 6.13: Hidden state observation model means inferred by the SS-BD-IHMM (left) and the BD-IHMM* (right) for one of the 40 datasets in the "twin waffles" task of §6.3.2. Marker colors indicate the block membership of the hidden states. The BD-IHMM* must "oversegment" the hidden states to approximate the data generating process's tendency to sample transitions between nearby hidden states, but the SS-BD-IHMM, which encodes this tendency in its prior, partitions the states correctly.

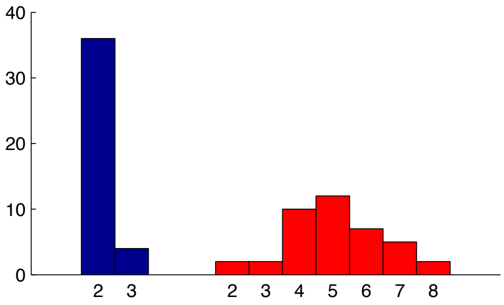


FIGURE 6.14: Histograms of the number of hidden state partitions inferred for the 40 "twin waffles" experiment (§6.3.2) datasets for SS-BD-IHMM (left, blue) and BD-IHMM* (right, red) models. The correct number of partitions is two; the BD-IHMM* typically "oversegments" the hidden states.

data. We conclude that the additional "smoothness" structure incorporated into the prior we used with the SS-BD-IHMM in this task confers a decisive advantage in the inference of sub-behavior structure.

6.4 Visual object data

At long last we approach the problem that motivated the entire development of the models described in this dissertation: unsupervised learning of view based object models from

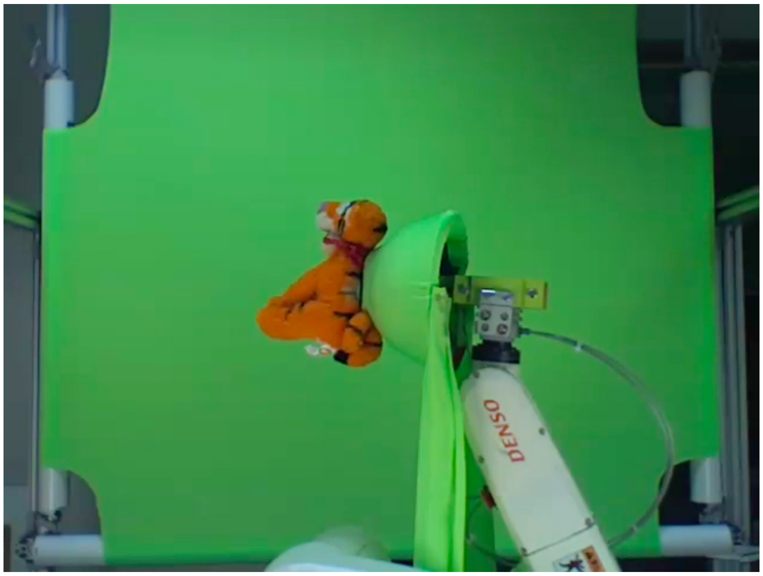


FIGURE 6.15: 640 × 480 pixel still image from the raw video data collected for the visual object task dataset. The object, a stuffed Tigger doll in this case, attaches to the six degree-of-freedom robot (bottom right) via a green fabric-clad mount and apron, which helps in frontal views to correctly limit the object mask region derived by color keying. During data collection, the robot moves objects smoothly between a series of random orientations, which can vary in pitch, roll, and yaw. More details appear in §6.4.1.

videos of moving objects.

6.4.1 Data preparation

We perceive our visual world as containing numerous varied, independent, freely moving objects. To produce a dataset at least somewhat relatable to our experience of these objects, we collected a number of common household items and placed them on a six degree-of-freedom industrial robot arm positioned in front of a green fabric screen (Figure 6.15). The robot arm moved the objects through a wide range of angular orientations, with considerable variation in pitch and yaw (rotation about the vertical and lateral axes about the object; angular ranges were about 320° and 90° respectively) and moderate variation in roll (rotation about the longitudinal axis; angular range was between 30° and 90° depending on how the object was mounted). These orientations were achieved by locking the number 2 ("shoulder"), number 3 ("elbow"), and number 4 ("forearm") joints of the arm in a "crouch" pose (see figure) and varying the angles of the number 1 ("base pivot") and number 5 and 6 ("wrist") joints. Some objects were held from the side as shown in the figure; others were supported underneath in a manner similar to a waiter supporting a tray of food.

The orientation trajectory for the object was generated by sampling points at random within the three dimensional space of permissible base pivot and wrist joint angles. The robot control software directed the robot to rotate the joints smoothly and linearly between these angles in sequence, where "linearly" means that the joint angles of the robot configuration trajectory between the sampled orientations were determined via linear interpolation. Rotational speed and acceleration were not linear and were determined in part by the robot control software, though a unitless speed control parameter was adjusted to produce rotational speeds ranging between around 30° and 120° per second.

Lacking statistical data on the orientation behavior of everyday objects in natural visual experience, and constrained in part by the kinematic limitations of the robot arm, we resigned ourselves to designing a simple orientation trajectory sampling procedure that yielded a plausible distribution of orientations and rotational motions. For each point, the base pivot joint angle was sampled uniformly from the range of valid joint angles within 70° of the previous base pivot angle. The wrist joint angle that most significantly determined object pitch, meanwhile, was sampled uniformly from the valid angle range for that joint. Finally, because we tend to see or interact with many objects while they are in an upright orientation, the wrist joint angle that most significantly determined object roll was sampled such that near-upright orientations were most likely. For objects supported from underneath, the angle was drawn from a distribution derived from a mixture of normal distributions, in which one component's standard deviation was half the valid angular range and the other's was 20% of this range, and where both were summed with weights $[0.5, 0.5]$ before the resulting mixture was truncated to the valid angular range. Objects supported from behind are naturally biased toward upright orientations due to the physical layout of the robot, where true roll can only occur through the combined actions of both wrist joints; thus, roll angles under this circumstance were drawn from a normal distribution whose standard distribution was half the valid angular range, truncated to this angular range.

Lastly, to avoid overrepresenting slow or near-stationary motions in our dataset, new orientations were only added to the trajectory if the combined pitch and yaw angular distance from the last orientation exceeded 10°. The slower motions that would arise from smaller changes in orientation are already well-represented by the slow motions in the acceleration and deceleration phases at the beginning and end of each rotational move between orientation trajectory points.

We recorded five-minute videos of each object in our collection undergoing the random robotic rotation related above. Our recording apparatus was a small video camera built into an Apple MacBook Pro laptop computer manufactured in November of 2006. This camera produced 640x480 pixel color videos at variable frame rates near 30 frames per second. A third party software program[3] was installed to clamp the camera white balance and exposure settings to a preset value; review of videos post-recording, however, revealed that this program allowed occasional automatic, inadvertent changes in these settings throughout our recording days. These changes were seldom enough that they did not affect most of the recorded object videos; those that were adversely affected by abrupt setting changes were discarded from the dataset.

When recording was complete, we processed the videos in Apple, Inc.'s Final Cut Express, a commercial video editing software package. We first re-encoded the videos at the NTSC-standard fixed frame rate of 29.97 frames per second, then used the built-in color-keying functionality of the software to create a second "matte video" with white pixels in locations in the original video whose color diverged from the green screen background, and black pixels everywhere else. Since items besides the target object (e.g. the robot) also show up in the matte video, we touched up the matte video with a custom program that isolated the single connected region of white matte pixels in each frame that overlaid the actual object. Portions of video where this was not possible (e.g. where the object occluded the robot, causing their matte regions to merge) were discarded from the dataset.

Our final data preparation step combined the information from the object and matte videos to create a collection of low-resolution "input movies" suitable for use as input data for the inference system. The process of creating a frame of an input movie is as follows. First, both the object movie and matte movie frames are cropped such that the object is centered in the smallest possible square frame that contains all of the object pixels. The cropped frames are then rescaled to 30×30 pixels, and pixels in the object movie frame that are not white in the matte movie frame are set to black. These processed frames are combined into a single four-channel input movie frame, with the first three channels the red, green, and blue color channels, and the fourth channel an alpha channel containing the matte data. Finally, we reduce the frame rate in input movies to allow the algorithm to process

[3] iGlasses by Ecamm Network, LLC, `http://www.ecamm.com/mac/iglasses/`.

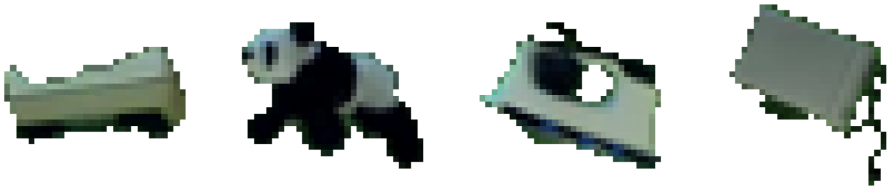


FIGURE 6.16: Input movie stills from four separate input movies. From left: a Scotch tape dispenser, a stuffed toy panda, an iron, and a toaster. The alpha channel information in the input movie has been applied so that only object pixels are shown.

more input data, and to eliminate temporally adjacent frames which, at the 30 × 30 pixel resolution, are largely redundant. The experiments described in this subsection employ 2 frame per second input movies. Examples of input movie frames appear in Figure 6.16.

Once converted into this format, input movies are typically broken at random into clips that are roughly 15 seconds in length. It is small unlabeled collections of these clips, intended to simulate momentary, chance encounters with objects, that make up the input data for inference.

6.4.2 Observation model-related distributions

Summarizing from the previous subsection, our input data for this visual object data organization task is a collection of short 2 frame-per-second input movie clips, where individual frames are $30 \times 30 \times 4$ element arrays. These arrays correspond to 30×30 pixel RGB color images with an alpha (or opacity) channel, and are the observations we aim to account for with HMMs inferred via the techniques of this thesis.

Among contemporary applications of HMM methods, $30 \times 30 \times 4 = 3,600$ dimensional observations are rather sizable, and naive approaches to inferring observation models, or even computing the likelihood of an observation given an observation model, can be computationally taxing. For this reason, rather than drawing observation models from a distribution over a continuous space (e.g. drawing a 3,600 dimensional multivariate normal observation model mean from a 3,600 dimensional multivariate normal posterior), we maintain a finite library of candidate observation models derived from the input data. Observation models associated with hidden states during inference are therefore sampled from a distribution over all of the models in the library.

This approach has one very compelling advantage: costly computations like observation likelihoods can be computed beforehand and stored in large tables—inference code need only look up or sum the appropriate table values to compute likelihoods, marginals, and so forth. A secondary advantage is that the relationship between the observations and the observation models is extremely simple to understand, in contrast to other tractable but more sophisticated representations (e.g. linear or non-linear projections of the image data onto a low-dimensional space)—this simplicity makes it easier to reason about the degree to which the SS-BD-IHMM and the other tested approaches contribute to the results described below. There are disadvantages as well. First, there is no assurance that any one of the candidate observation models in the library is the "best" for describing a given collection of input movie frames (indeed, the "best" model is almost certainly *not* in the library). Next, having a finite collection of candidate observation models does not seem to jibe well with the idea of building HMMs with (conceptually) countably infinite hidden states. Eventually, some states must draw the same observation model from the library. In practice, however, the finite collection of hidden states associated with the input data is much smaller than the number of candidate observation models in the library, and circumstances where any of these states have the same observation model are rare. Furthermore, even if two states do have the same observation model, it is not the case that they have identical semantics. The states could be in different blocks, for example; more subtly, differing patterns of transitions into and out of the states can help the model encode specific dynamic behaviors that would be lost if the states were somehow merged together. In any case, we are confident that this approach can yield useful, descriptive models, and that the considerable computational efficiencies it affords are worth pursuing.

In this task, candidate observation models in the library are rather straightforward, being 3,600 dimensional multivariate normal distributions with a fixed spherical covariance. Expressing this algebraically, the probability of observing a particular input movie frame $\boldsymbol{y}_t$ given parameters of an observation model $\boldsymbol{\theta}_m$ copied from the library collection is

$$p(\boldsymbol{y}_t \,|\, \boldsymbol{\theta}_m) = \prod_{i=1}^{3{,}600} \frac{1}{\sqrt{2\pi\sigma_\theta^2}} \exp\left(-\frac{(y_{t,i} - \theta_{m,i})^2}{2\sigma_\theta^2} \right),$$

written here as the equivalent product of 3,600 independent univariate normals, with the standard deviation σ_θ shared among all dimensions and among all observation models. This model ignores important traits of the observations: in our data, for example, pixel values never take on values outside the range $[0, 1]$, while the support of this model spans

FIGURE 6.17: Input movie still frame of an iron (left) blurred by a Gaussian kernel ($\sigma = 4$) to produce a candidate observation model mean (right). The large black blob at the center of the iron is the power cord, which has been secured beneath the handle for the iron's ride on the robot.

$\mathbb{R}^{3,600}$. What's more, the model does not attempt to capture any correlations between image pixels, neither obvious ones (e.g. the correlation in R, G, and B intensity values for the same pixel) nor the richer and more complex inter-pixel correlations that are the subject of much current research (e.g. [110]). On the bright side, these distributions are simple to work with, and the logarithm of the above expression can be interpreted as a scaled, shifted sum squared difference between the observation and the observation model mean. Moreover, since the observation models are always selected from a fixed library of models derived from the data, there is no chance that the generous support of these models will result in a $\boldsymbol{\theta}_m$ sample whose mean contains out-of-range or otherwise bizarre pixel values—all $\boldsymbol{\theta}_m$ will "look like" object views.

To generate the candidate set of observation models, a large number of frames are selected at random from the input movies. Each of the color/alpha channels is subjected to a Gaussian blur operation employing an isometric kernel with a fixed standard deviation (c.f. Figure 6.17). The resulting blurred image becomes the mean of one of the candidate observation models.

As mentioned previously, once the candidate observation models have been generated, it is easy to create a lookup table containing $p(\boldsymbol{y}_t \mid \boldsymbol{\theta}_m^{[c]})$ for any observation $\boldsymbol{y}_t$ and any candidate observation model $\boldsymbol{\theta}_m^{[c]}$ (the [c] denoting *candidate*). Computing marginal observation probabilities is also straightforward, with

$$p(\boldsymbol{y}_t \mid \boldsymbol{\phi}_k) = \sum_m p(\boldsymbol{y}_t \mid \boldsymbol{\theta}_m^{[c]}) \cdot P(\boldsymbol{\theta}_m^{[c]} \mid \boldsymbol{\phi}_k),$$

with $\boldsymbol{\phi}_k$ some observation model generating distribution, on which topic we will now proceed.

In general, observation model generating distributions reflect some commonality among the observation models they generate. In the first artificial data task (§6.3.1), these distributions indicated that the parameters of observation models for states in the same block tended to cluster around specific locations in $\mathbb{R}^2$. Observation models describing different views of the same object certainly have many things in common—color, shape, texture, and so forth—but mainly for simplicity's sake we elected to have our observation model generating distributions deal exclusively with color. Different views of the same object should, according to these distributions, have the same color; put slightly differently, given an object, its object model generating distribution indicates what color will tend to predominate in specific object views. Thus, although it is only one of the general object traits that we might have chosen, overall color is a useful tidbit of global information about an object, and one that could conceivably be helpful for object model inference.

The "candidate library" approach to observation models makes the necessary calculations for observation model generating distributions relatively straightforward. First, for each of the candidate observation models in the library, we derive an RGB "global observation model color" as a weighted average of the colors of the 30 × 30 pixels in the observation model mean. The weights for averaging come from the corresponding observation model alpha channel values; in this way, pixels that are not part of the object image do not contribute to the color associated with the observation model.

Let $\theta_{m,\bar{R}}$, $\theta_{m,\bar{G}}$, and $\theta_{m,\bar{B}}$ refer to the weighted average red, green, and blue intensity values for a particular observation model $\boldsymbol{\theta}_m$. An observation model generating distribution $\boldsymbol{\phi}_k$ has three parameters: $\phi_{k,\bar{R}}$, $\phi_{k,\bar{G}}$, and $\phi_{k,\bar{B}}$, and has the following proportions:

$$P(\boldsymbol{\theta}_m \mid \boldsymbol{\phi}_k) \propto \left(\frac{1}{\sqrt{2\pi\sigma_\phi^2}}\right)^3 \exp\left(-\frac{(\theta_{m,\bar{R}} - \phi_{k,\bar{R}})^2 + (\theta_{m,\bar{G}} - \phi_{k,\bar{G}})^2 + (\theta_{m,\bar{B}} - \phi_{k,\bar{B}})^2}{2\sigma_\phi^2}\right), \tag{6.4}$$

or, in other words, a 3-dimensional spherical multivariate normal distribution with standard deviation σ_ϕ^2, centered at the parameters. We specify a proportion because the equation above is normalized over all observation model candidates in the library; that is, $\sum_m P(\boldsymbol{\theta}_m^{[c]} \mid \boldsymbol{\phi}_k) = 1$, with m iterating over all the candidates.

Although there was not as pressing a need for it in this circumstance, we chose to adopt the "finite library" approach to observation model generating distributions as well. We generated our library by drawing many random RGB triplets from the unit cube—usually around 2,000 of them. We specified a uniform prior over these triplets. Once again, this approach offered efficiencies by allowing us to cache probabilistic computations needed for inference, such as $P(\boldsymbol{\theta}_m^{[c]} \mid \boldsymbol{\phi}_k^{[c]})$, and recover their values later through rapid table lookups.

6.4.3 Modification function

Unsurprisingly, we specify a spherical Gaussian kernel for the intra-block transition similarity function $\mathrm{Sm}(\boldsymbol{\theta}_n; \boldsymbol{\theta}_m)$:

$$\mathrm{Sm}(\boldsymbol{\theta}_n; \boldsymbol{\theta}_m) = \prod_{i=1}^{3{,}600} \exp\left(-\frac{(\theta_{n,i} - \theta_{m,i})^2}{2\sigma_{\mathrm{mod}}^2}\right). \tag{6.5}$$

Here, too, we achieve efficiencies by precomputing a lookup table containing $\mathrm{Sm}(\boldsymbol{\theta}_n^{[c]}; \boldsymbol{\theta}_m^{[c]})$ for all pairs of candidate observation models. Our similarity function in this case is symmetric, so it is not necessary to store the entire table.

6.4.4 Experiment 1: few objects, many runs

Our first experiment compares the relative performances of four separate models related to the BD-IHMM on the problem of learning view-based object models from short video clips. These models are

1. The BD-IHMM (Chapter 3),
2. The full SS-BD-IHMM (Chapter 5),
3. The BD-IHMM*, a variant of the BD-IHMM with a hierarchically-structured prior on observation models borrowed from the SS-BD-IHMM (§6.3.2),
4. A fourth model with the structured transition dynamics prior of the SS-BD-IHMM, but the simpler "single-layer" prior on observation models borrowed from the BD-IHMM; in other words, the prior on observation model parameters is shared across all hidden state blocks.

These four approaches determine what effect, if any, the two enhancements of the SS-BD-IHMM over the BD-IHMM have on the discovery of object models in object video clips, separately or together. To be able to discern such effects with statistical certainty,

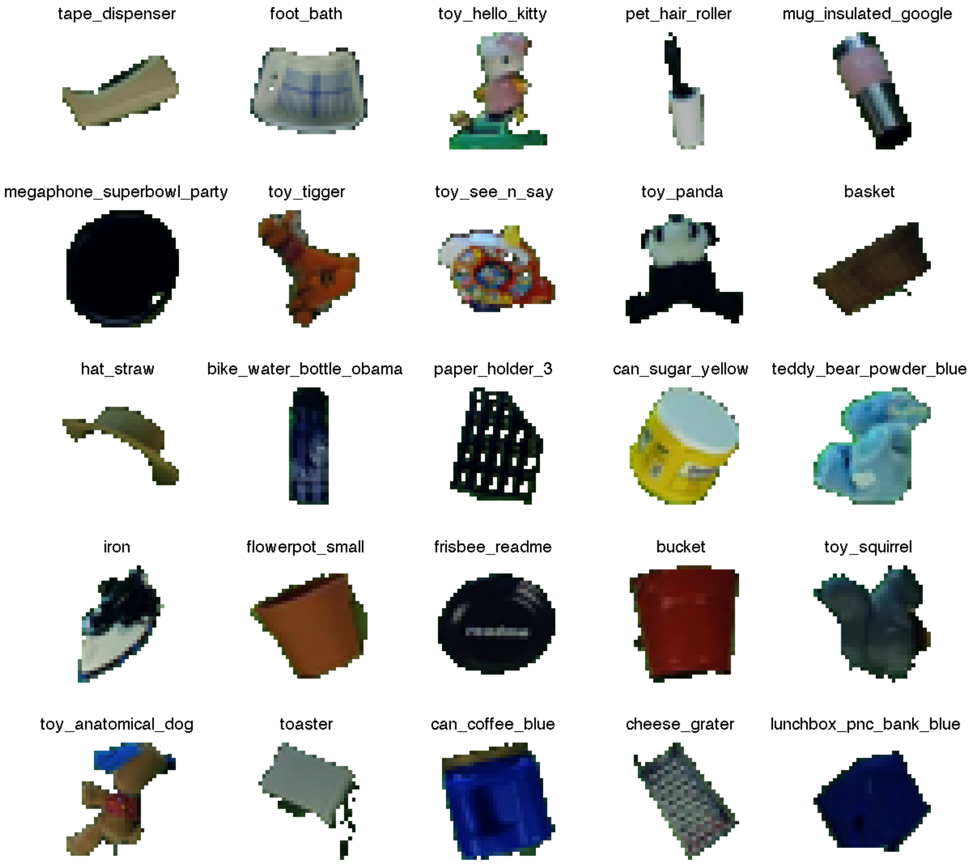


FIGURE 6.18: Still images from input movies taken for the 25 objects used in the object model learning experiment in §6.4. A number of objects are visually quite similar at certain aspects, including `frisbee_readme` and `megaphone_superbowl_party` (a toy megaphone, here viewed down the "barrel") in these images, along with `flowerpot_small` and `bucket`, and other pairings under different viewing circumstances.

we must perform object model learning several times over different datasets; to achieve the necessary statistics in a limited time frame, we therefore restrict these datasets to only three objects. These three are drawn randomly from a library of 25 objects shown in Figure 6.18.

To save time on software implementation and minimize the chances of creating new software errors, we elected not to implement the four models considered in this experiment themselves but instead to make very close approximations of the models by adjusting the settings of the various SS-BD-IHMM parameters described in §6.4.2 and §6.4.3. As the σ_{mod}

Model	σ_θ	σ_ϕ	σ_{mod}
BD-IHMM	4.0	100	1000
SS-BD-IHMM	4.0	0.25	15
BD-IHMM*	4.0	0.25	1000
"Model 4"	4.0	100	15

TABLE 6.2: Observation model hierarchy and modification function parameters for the SS-BD-IHMM in the experiments of §6.4.4. Large values (100 and greater) effectively "disable" the related functionality of the SS-BD-IHMM in this task, allowing this model to emulate the other three models listed in the left-hand column. There is no significance to large values being either 100 or 1000; mainly, they reflect when we got tired of typing zeros. Other values were derived manually from limited experiments on a fixed dataset comprising clips of the first three objects shown in Figure 6.18; inference of parameters like these remains a subject for future investigation.

value increases, for example, the similarity function $\text{Sm}(\boldsymbol{\theta}_n; \boldsymbol{\theta}_m)$ in Equation 6.5 approaches the $\text{Sm}(\boldsymbol{\theta}_n; \boldsymbol{\theta}_m) = 1$ function entailed by the BD-IHMM. Likewise, a large σ_ϕ value reduces draws from the observation model generating distributions described by Equation 6.4 to the uniform draws from the RGB triplet library (c.f. §6.4.2) that the BD-IHMM would use. Table 6.2 shows the SS-BD-IHMM observation model hierarchy and modification function parameters used to approximate the four models targeted by this experiment.

6.4.5 Examining an object learning result

Here we take a look inside one of the SS-BD-IHMM object model learning results that we will be compiling by the hundreds in our experimental performance evaluation in §6.4.6. We ran SS-BD-IHMM parameter inference on a dataset identical to those used in the §6.4.6 experiments; that is, one comprising a small number of short video clips of three objects. In this case, the objects happened to be a Hello Kitty toy, a flowerpot, and a Tigger stuffed animal. These objects are fairly distinct, so the SS-BD-IHMM has no problem telling them apart; fortunately, we test more challenging datasets in the actual experiments.

Figure 6.19 shows an inferred hidden state transition matrix and the mean of the observation models (or, object views) associated with each hidden state. In a practical sense, these are the object models learned from the data. To help convince ourselves that these models are encoding meaningful dynamic relationships between the object views (that is, short of actual testing), we show in Figure 6.20 the most probable trajectories through each object's hidden states. Identifying this path is an instance of the classic traveling salesman problem, but with nine states to consider at most, complexity is not an issue. The trajectories in

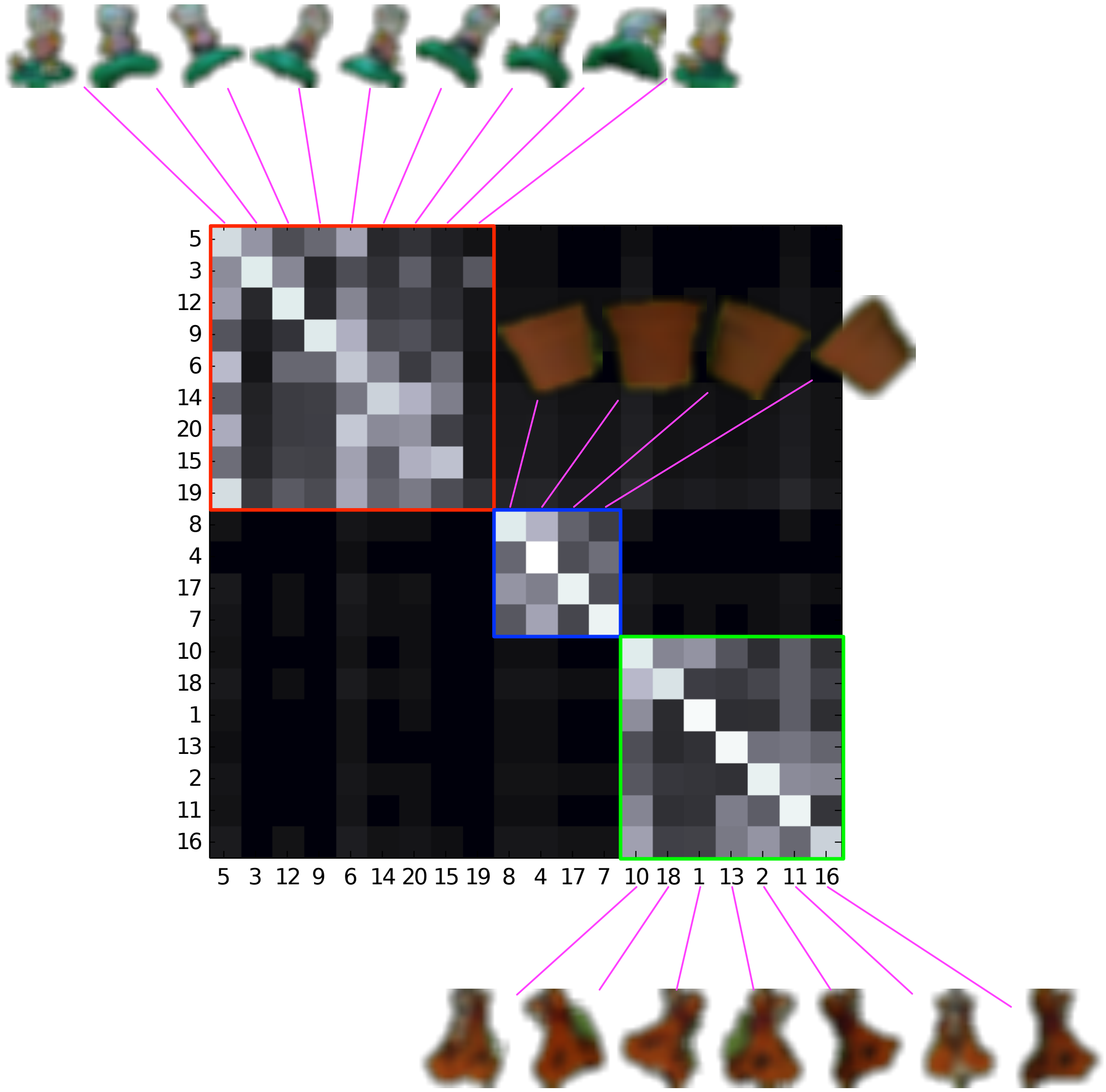


Figure 6.19: A visualization of a truncated (c.f. §6.1.2) transition matrix derived from inferred SS-BD-IHMM parameters for the model examined in §6.4.5. Rows are sorted in the order in which hidden states are first visited in the inferred training set hidden state sequence v. Colored squares highlight transitions between hidden states with the same block assignment; there are three blocks here, corresponding to the three objects in the training set data. Observation model means for each state are also shown. This transition matrix shows some of the same structured preference for transitions between similar-looking object views built into the prior, particularly in the high probability of self-transitions.

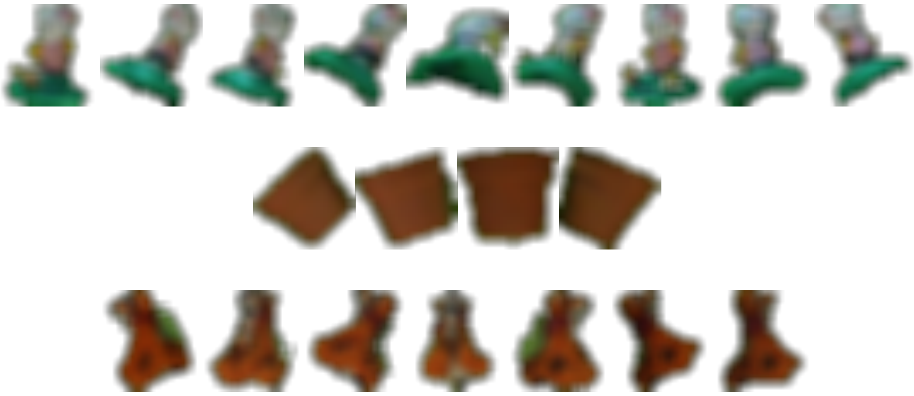


FIGURE 6.20: A visualization of the highest probability trajectories through the hidden states that make up each object in the model examined in §6.4.5. Each trajectory, shown in order from left to right, transitions smoothly through its object's hidden states without jumps between visually disparate object views. At bottom, the three leftmost views have the Tigger toy facing left; the three rightmost views have it facing right.

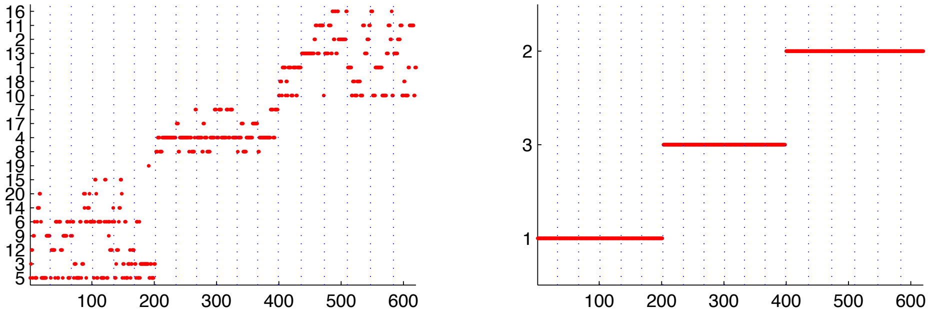


FIGURE 6.21: Left: inferred hidden state trajectory (v) of the model examined in §6.4.5. Blue dotted lines show boundaries between video clips. The order of the clips is not significant—they are treated as separate inputs to the inference system. That said, the first six clips contain video of the Hello Kitty toy, the next six show a flowerpot, and the remaining six show a stuffed Tigger doll. For this dataset, the inference system has correctly allocated distinct object views for each object—that is, no single view is assigned to data from more than one object. At right, the block (i.e. object) index assigned to each frame, as derived from the inferred hidden state trajectory v and hidden state block labels z. This labeling of the training set data happens to be perfect.

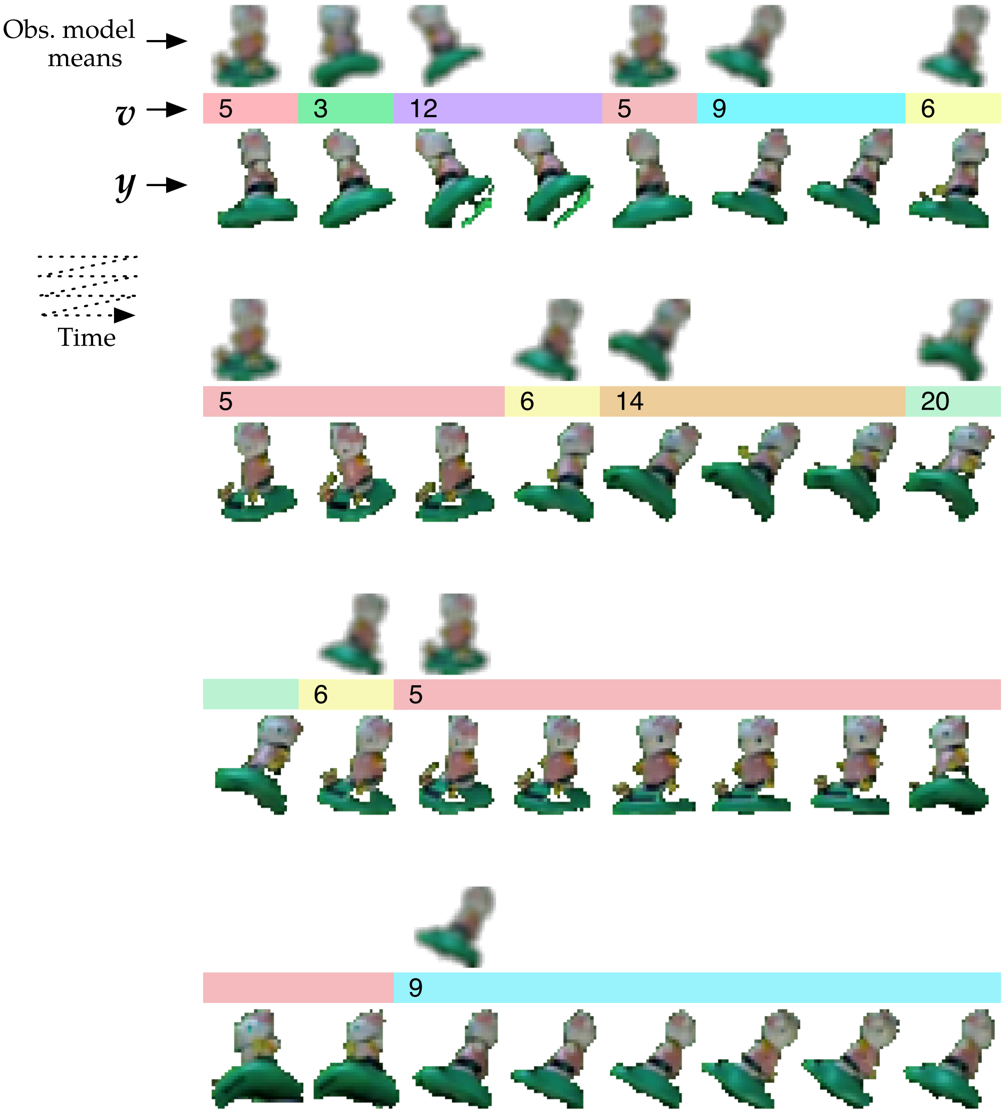


FIGURE 6.22: A 16-second, 32-frame excerpt of one of the input data sequences ($\boldsymbol{y}$) used as training data for the model examined in §6.4.5, along with the inferred hidden state trajectory ($\boldsymbol{v}$) and the observation model means associated with each hidden state. There are 32 observations in total, shown in row-major order from left to right, top to bottom. For ease of comparison, observation model means are displayed again whenever a hidden state recurs in the trajectory.

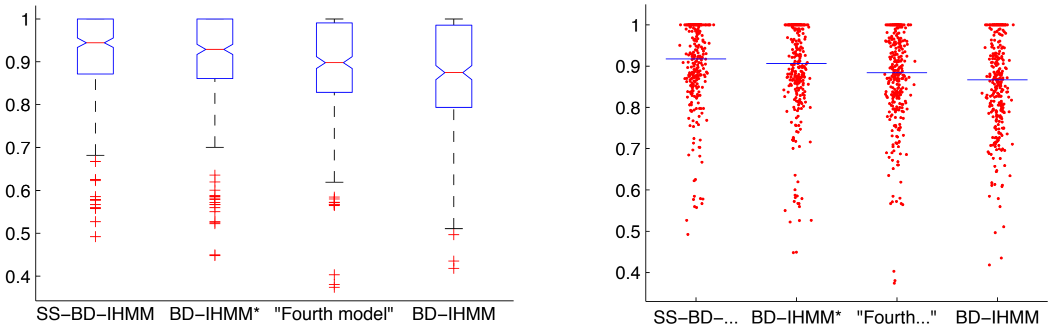


FIGURE 6.23: Box and whisker plots (left) and scatter plots (right) with mean values (blue lines) of adjusted Rand indices for the four test conditions in the first object model learning experiment (§6.4.4). As detailed in the text, SS-BD-IHMM object organization performance is significantly better than the "fourth model" and the BD-IHMM.

Figure 6.20 show smooth transitions through the hidden states corresponding to plausible physical object motions.

Further visualizations appear in Figure 6.21 and 6.22, which respectively show the inferred hidden state trajectory and block labels for the training set and a detail of an inferred hidden state trajectory respectively. These figures are not intended to make any particular point, but instead to offer further visual guidance to the reader on the nature of the inference task.

6.4.6 Results of Experiment 1

We conducted 336 runs of the experiment as described, with each of the three objects' training data comprising six input movie clips of about fifteen seconds in length.[4] These clips made up 30 percent of the available input movie data for each object; remaining data were used for the test set.

Figure 6.23 uses the adjusted Rand index to compare the four models' abilities to organize the training data into object models. The SS-BD-IHMM, with a median adjusted Rand index of 0.944, performed significantly better than the "fourth model" (Wilcoxon signed-rank test, $p = 1.7 \times 10^{-12}$) and the BD-IHMM ($p = 1.6 \times 10^{-18}$) on the same training data. Performance was not significantly better than the BD-IHMM* ($p = 0.09$), although we suspect that further runs might yet yield this result. Both the BD-IHMM* and the "fourth model"

[4] There is no particular significance to conducting 336 runs—the experiment was merely left running unattended during the Christmas holiday.

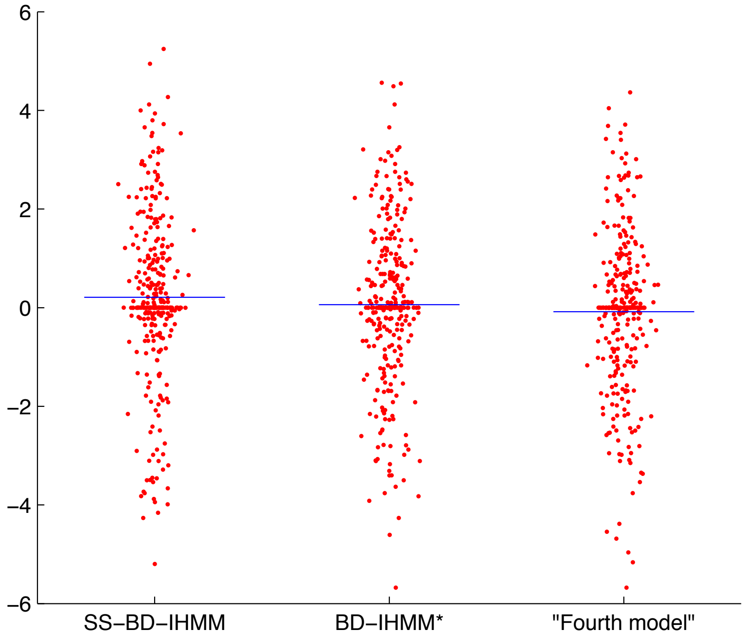


FIGURE 6.24: A visual comparison of the test set likelihood performance of the SS-BD-IHMM, the BD-IHMM*, and the "fourth model" against the BD-IHMM. **To interpret:** Points above 0 indicate performance gains over the BD-IHMM. If y is the y-coordinate of a point and p is the likelihood of the test set data according to the corresponding complex model (not shown), then the BD-IHMM likelihood for the test set data is $p^{\exp(y)}$. Grand differences in likelihood are not uncommon in HMM settings. **Blue lines** indicate the scatter plot means: 0.2113, 0.0616, and -0.0813 from left to right. As detailed in the text, SS-BD-IHMM model test set data likelihood performance is significantly better than the "fourth model" and the BD-IHMM.

were also significantly better than the BD-IHMM ($p = 3.1 \times 10^{-13}$, $p = 0.004$ respectively). We conclude that the additional structure in all three complex models yield better performance in unsupervised categorization of the training data, and that more structured transition dynamics priors ("fourth model") and and inclusion of appearance-based grouping cues (BD-IHMM*) independently yield improvements in this task.

Figure 6.24 compares the test-set likelihood performance of models inferred by the SS-BD-IHMM, the BD-IHMM*, and the "fourth model" with models inferred by the BD-IHMM. Here, too, the SS-BD-IHMM performed significantly better than the "fourth model" (Wilcoxon signed-rank test, $p = 0.006$) and the BD-IHMM ($p = 0.008$), but not significantly better than the BD-IHMM* ($p = 0.13$). No other model shows significant performance gains over the BD-IHMM, suggesting that appearance-based grouping and structured transition dynamics priors have complementary benefits in producing models that predict the future appearance and behavior of visual object data.

We interpret the outcomes of this object model learning experiment as reflecting the successful realization of the "prior we were waiting for" in §5.1.2. An approach that both explicitly captures global appearance traits of an object and uses view similarity to inform inference of its dynamic behavior creates better view-based object models than one that lacks these capabilities.

6.4.7 Experiment 2: many objects, few runs

Having evaluated the experimental performance the SS-BD-IHMM and related models on limited dataset, we now wish to place the model to the test for which it was originally designed. In this experiment, our input dataset makes use of all 25 objects at the same 30% training set/70% test set ratio employed in the previous experiment. Since there is considerably more data in this experiment, running times for inference are longer, and there was not time to collect the same quantity of performance statistics for this task that there was with only three objects in the training data. Our examination of the results of this experiment will necessarily be more qualitative in nature.

6.4.8 Results of Experiment 2

To offer some visual insight into the performance of the SS-BD-IHMM on this larger dataset, Figures 6.25, 6.26, 6.27, and 6.28 display various object model learning results from a single SS-BD-IHMM posterior sample. These visualizations show clear identification of object models and the relationships of their object views. A few mistaken judgments appear—the image captions describe them in more detail—but these can typically be attributed to several of the objects looking very similar.

To show that the performance in these four figures are not a lucky fluke, Figure 6.29 displays block label assignments for the training data in eight separate SS-BD-IHMM inference runs; seven in addition to the one shown in Figure 6.25. Given the similarity of these results, it seems safe to conclude that the SS-BD-IHMM can derive meaningful object models from the 25-object video dataset.

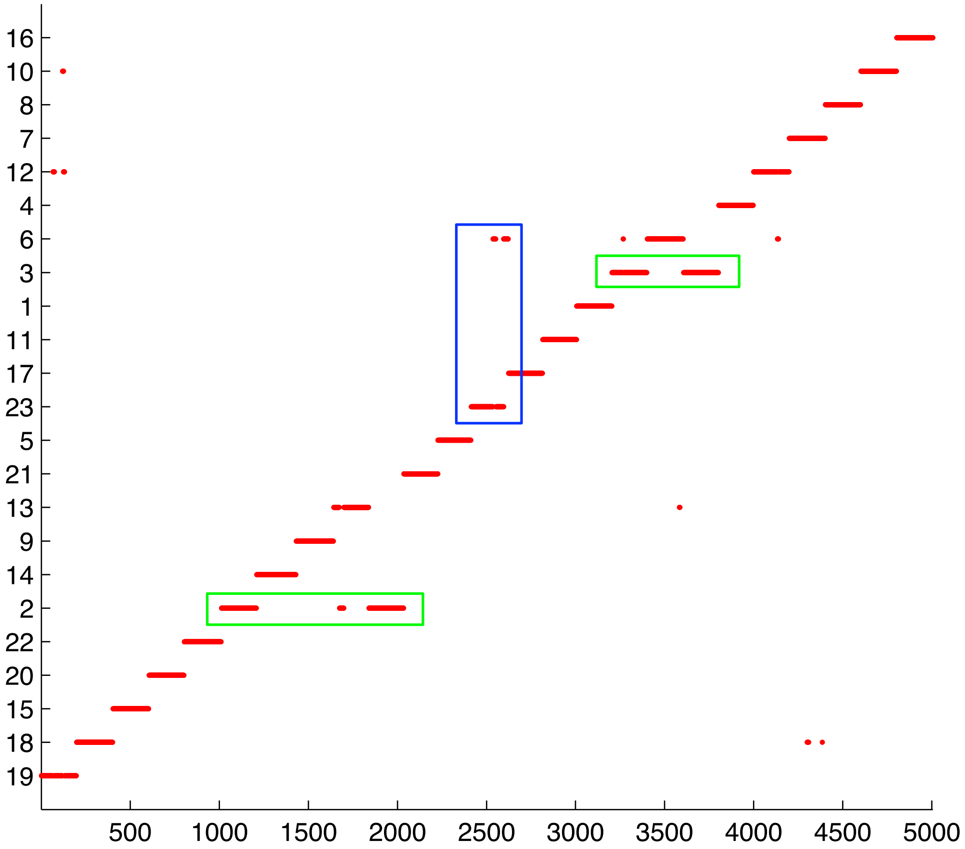


FIGURE 6.25: Trajectory of the input data through the inferred hidden state blocks for a particular run of the SS-BD-IHMM on all 25 objects (see §6.4.7). The input data consists of numerous short clips, each showing a single moving object. Clips themselves are independent of each other; they do not have an order. In this figure, however, they have been concatenated so that clips showing the same object are contiguous. Boundaries between separate clips are not shown. A perfect trajectory for this clip ordering would resemble a staircase with 25 distinct steps; this trajectory, while still quite good, has some flaws. Green squares indicate where two similar-looking objects were grouped together in the same object; the blue square shows where one object was modeled partly by "its own" states and partly by those inferred for another object. More visualization of this learning result appears in Figure 6.26. The adjusted Rand index for this partitioning of the input data is 0.88.

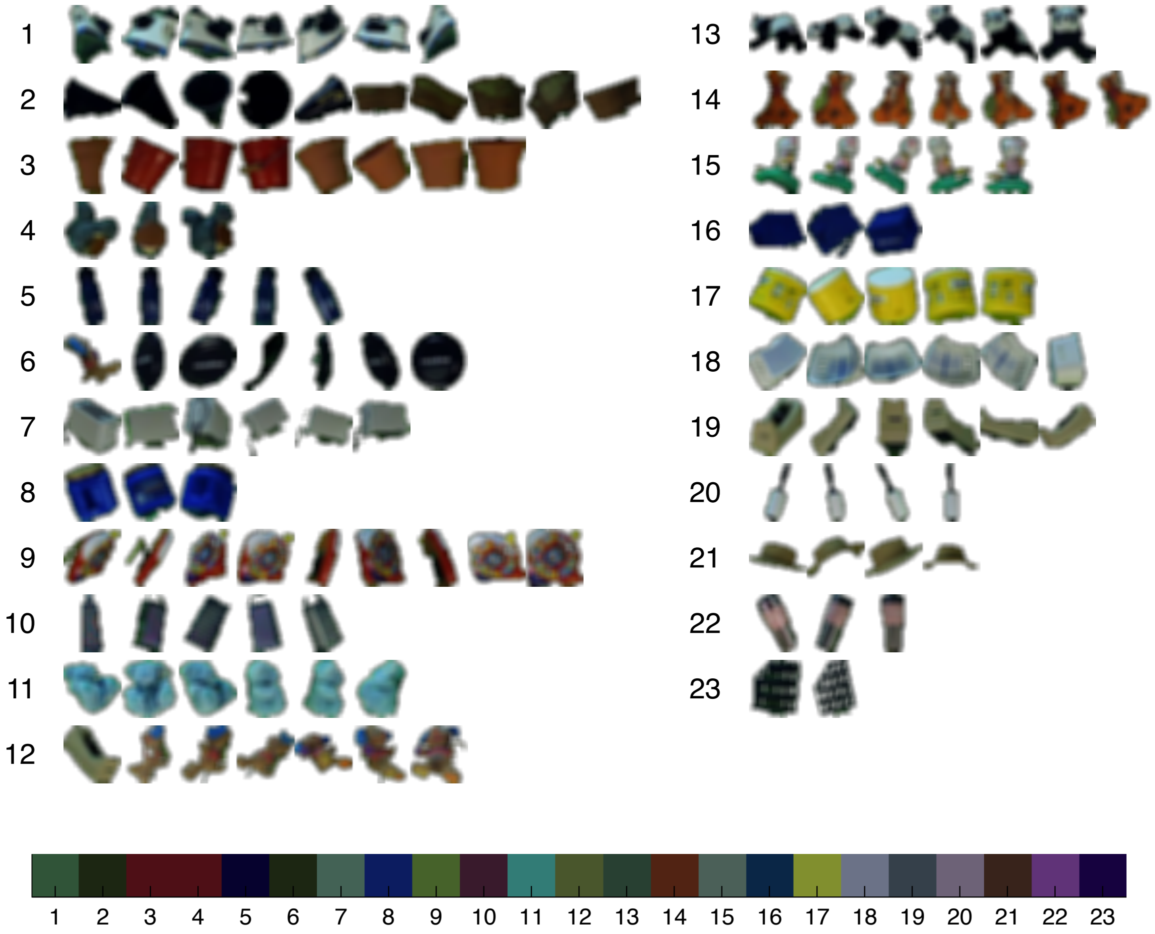


FIGURE 6.26: Means of the observation models associated with hidden states in each of the 23 objects inferred in the same run of the SS-BD-IHMM depicted in Figure 6.25. Means are presented in the "most probable trajectory" order described in §6.4.5 (also used in Figure 6.20). Overall, the model has done a decent job of identifying and capturing the visual variation of the different objects, with some obvious blunders. Objects 2 and 3 spuriously combine pairs of similar-looking objects; object 23 needs only two states because (as Figure 6.25 shows) the SS-BD-IHMM inference procedure chose to assign a significant number of observations of this paper holder to object 6. Finally, the stuffed animal captured by object 12 appears to be occasionally confused for other objects. **At bottom**, the inferred means of the object model generating distributions described in §6.4.2, showing the inferred global traits (i.e. color) associated with each object.

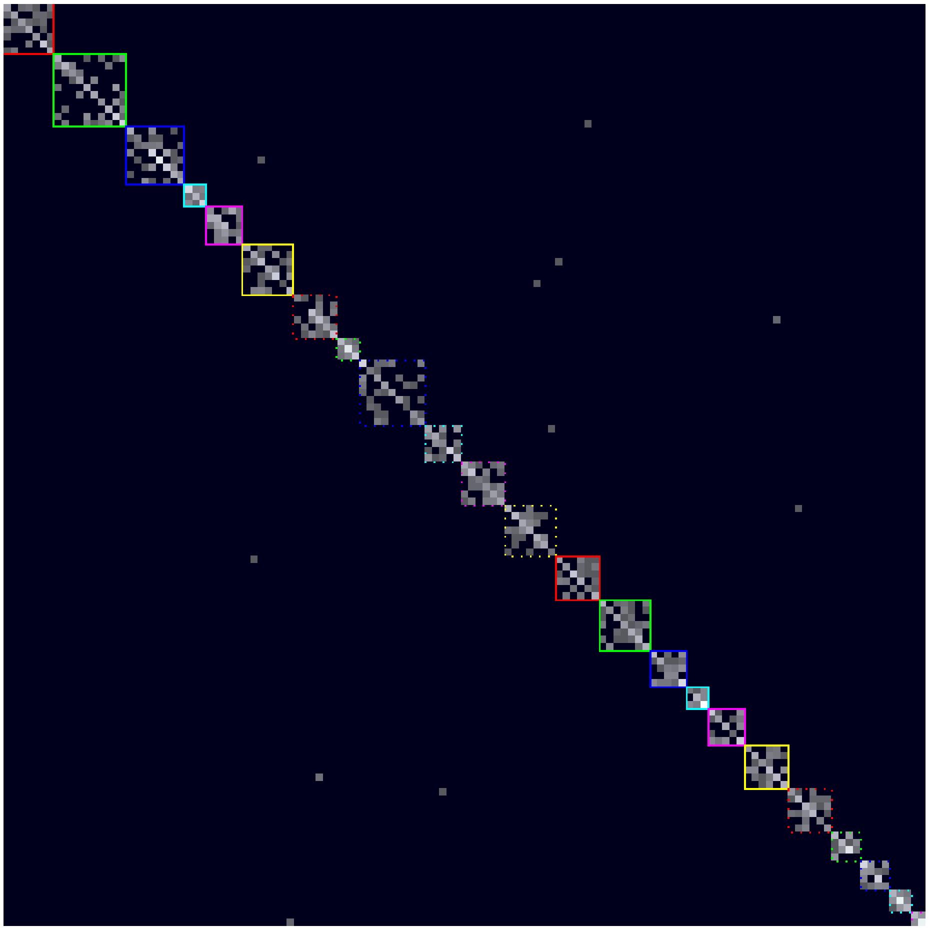


FIGURE 6.27: Visualization of the matrix of transition counts $\boldsymbol{c}$ derived from the hidden state trajectory $\boldsymbol{v}$ inferred in the 25-object SS-BD-IHMM run also examined by Figures 6.25 and 6.26. Transitions between states belonging to the same inferred object appear within solid or dotted-line colored boxes; a few wrongly inferred transitions between objects (such transitions never occur in the data) appear outside of the boxes.

6.4.9 Experiment 3: SS-BD-IHMM versus simpler hacks

This thesis takes great pains to develop a fairly complicated probabilistic model for describing time series. It is worthwhile to determine whether a simpler approach might do as well or better at a real-world task than a more complex system. To this end, we perform a new experiment nearly identical to the three-objects-at-a-time experiment in §6.4.4, but this time with the SS-BD-IHMM competing with several similar, simpler models. These models are cobbled together from existing techniques and have no principled basis for the way they integrate transition dynamics or appearance-based clustering into their inference decisions, but perhaps this extra effort is not really necessary.

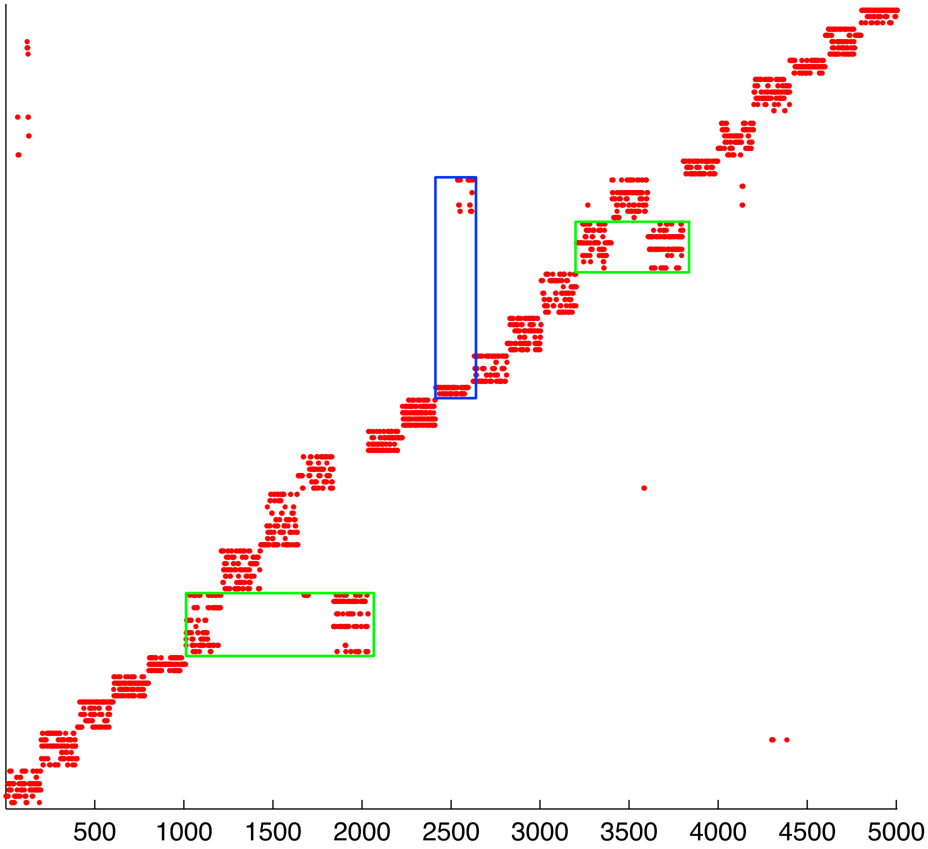


FIGURE 6.28: The hidden state trajectory for the 25-object input data, inferred in the same SS-BD-IHMM run also examined by Figures 6.25, 6.26, and 6.27. Clips are sorted as described in Figure 6.25, and the blue and green boxes identify the same errors described there. Off-block-diagonal dots indicate places where the inferred trajectory has incorrectly associated input data with hidden states in the wrong block. There are 127 hidden states in all.

¶ **Competing models** — We will consider six competing models that are simpler than the SS-BD-IHMM. These models are still somewhat sophisticated—for starters, all make use of Bayesian nonparametric methods, mainly so that they can have some means for inferring the number of hidden states and blocks necessary to describe the data.

Our first three competitor methods have a common beginning: rather than paying any attention at all to the dynamic nature of the data, they simply cluster the observations using a regular Dirichlet process mixture model (DPMM, §2.2.4). The mixture components and their parameters are exactly the same as the observation models in the SS-BD-IHMM, and

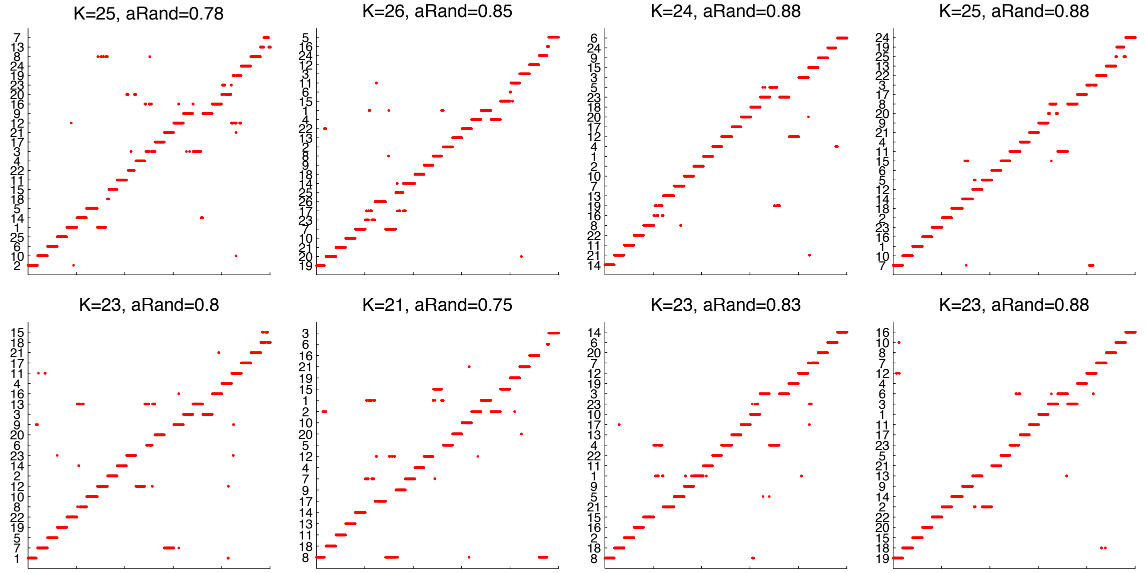


FIGURE 6.29: Hidden state block label trajectories for multiple SS-BD-IHMM runs on 25-object datasets. At bottom right, the same run as shown in Figure 6.28. Plot titles show K, the number of objects inferred from the data, and aRand, the adjusted Rand index for the depicted labeling of the training set data. While the technique was never able to tell the flowerpot_small object from the extremely similar bucket object (c.f. Figure 6.18), no other such failure to disambiguate is general to all of the runs.

the usual vague gamma hyperprior is expressed on the hyperparameter modulating clustering behavior (α). The Chinese restaurant process (§2.2.5) is used for DPMM inference, while hyperparameter inference takes place in the customary way (c.f. §3.4.8, [1]). Simply by assigning the observations to mixture components, the DPMM inference "infers" a hidden state trajectory, although this understanding of what are actually cluster assignments is applied *post hoc*.

With the hidden state trajectory and observation models inferred, our next step is to group the hidden states into blocks. We proceed in three ways, which yields the first three models:

- The first uses the DPMM to cluster states into blocks based on the similarity of their observation models. Here, the same color-based observation model generating distributions that were used in the SS-BD-IHMM and the BD-IHMM* (§6.4.2) are now used to cluster the inferred hidden states into blocks.
- In the second method, clustering is based on the transition dynamics—specifically, on the transition counts matrix $\boldsymbol{c}$ derived from the inferred hidden state trajectory.

This approach compares the patterns of state transitions into and out of individual states to determine which states have similar transition dynamics—and therefore deserve to be clustered together. This Chinese restaurant process-based approach is essentially the same as the "two-state-restricted Chinese restaurant process" that forms part of the proposal distribution for BD-IHMM split-merge trajectory sampling (§3.4.6; Equation 3.17), except now the number of available tables is unbounded, the Context operator simply returns counts of transitions into and out of individual states (i.e. for a given state m, it returns the vector sum of the mth row and mth column of the transition matrix) and the probabilities of hidden state observation models given their block assignments (that is, observation model clustering; c.f. the second term of Equation 3.17) are ignored.

- The third method combines the cues used in the prior two: transition dynamics and observation model similarity. The marginal probability of a state's observation model given those of other states present in a block is not ignored as in the second method, but is factored into the Chinese restaurant process probability for that block assignment.

The next three competing models also have a common beginning: instead of the Dirichlet process mixture model, the Infinite Hidden Markov Model (IHMM, §2.4) infers an HMM without blocks to describe the input data. In reality, as in the first experiment (§6.4.4), we do not implement the IHMM but instead approximate it via special SS-BD-IHMM parameter settings—specifically, adopting the same settings used for the emulated BD-IHMM in the first experiment, with the ξ hyperparameter fixed at the value 0 (preventing any modification function from having any effect on the base proportions $\boldsymbol{\beta}$) and hidden state block label ($\boldsymbol{z}$) sampling disabled for good measure.

From here, these models group inferred hidden states into groups, using the same three methods employed by the previous three models. In sum, two means of inferring hidden state trajectories and observation models, times three means of inferring hidden state blocks, yields six separate models.

6.4.10 Results of Experiment 3

For 158 randomly sampled three-object datasets, the same kind as those used in Experiment 1 (§6.4.6), we generated view-based object models using the SS-BD-IHMM and the six

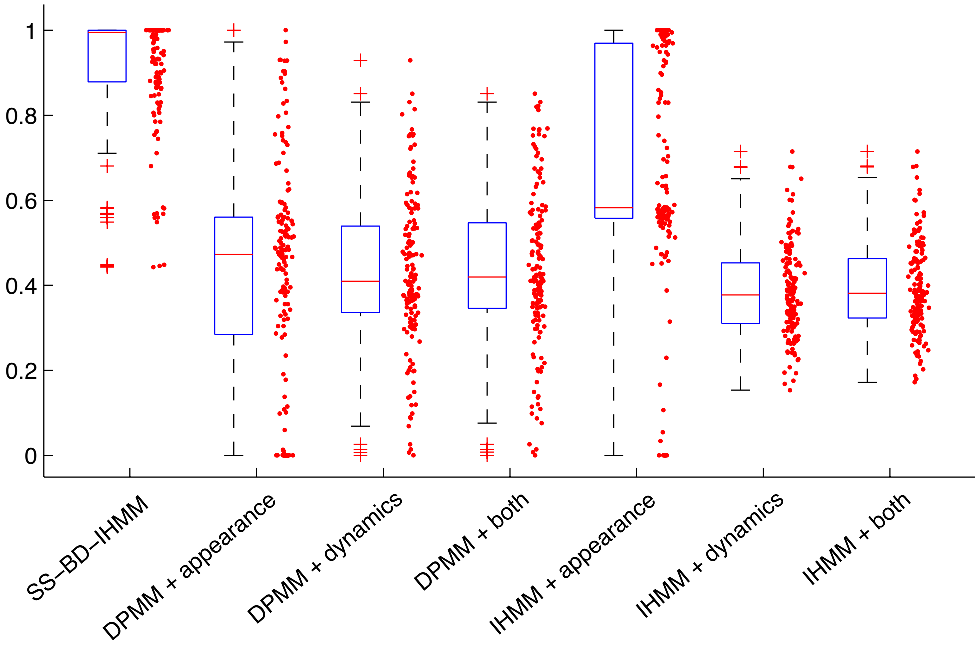


FIGURE 6.30: Box plots and scatter plots of adjusted Rand index scores for the 3-object model learning task achieved by the SS-BD-IHMM and the several DPMM and IHMM-derived "hack" models described in §6.4.9. Scores are compiled from 158 randomly drawn datasets. The SS-BD-IHMM has decisively better performance at partitioning the training set data than the other models.

"hacks" described above. Figure 6.30 reveals decisively better sub-behavior identification performance for the SS-BD-IHMM. The nearest competitor is the approach that uses the IHMM to allocate a set of hidden states to the data, then clusters the hidden states based on their appearance characteristics alone. This approach is labeled "DPMM + appearance" in Figure 6.30; the performance advantage for the SS-BD-IHMM is significant (Wilcoxon signed-rank test, $p = 4.4 \times 10^{-16}$). The scatter plot for this approach shows a large clump of adjusted Rand index scores around 0.57; curiously, a smaller clump appears in the SS-BD-IHMM plot, and indeed in the results of the first object learning experiment shown in Figure 6.23. For three-object datasets, adjusted Rand scores like these are associated with models that mistakenly group two of the objects together into a single object. A visual examination of the results reveals that of the 61 datasets where the "DPMM + appearance" model generated a training set labeling with an adjusted Rand score between 0.51 and 0.63, 53 clearly exhibited the mistake of combining two objects into one, with the others showing different kinds of labeling problems. By contrast, the SS-BD-IHMM only made this mistake six times for the same 61 datasets, with an additional 11 notably erroneous

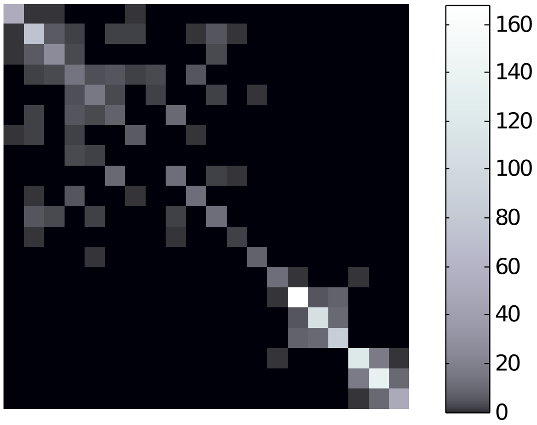


FIGURE 6.31: Symmetrized transition counts matrix (made by summing the transition matrix to its transpose) whose rows are used as clustering input in the "hack" (§6.4.9) approaches to dynamics-based sub-behavior discovery. The scale at right gives a sense of the counts value of each matrix entry. Rows of this matrix are too distinct for effective clustering using Dirichlet process mixture models with categorical distribution observation models.

labelings due to other problems (usually oversegmentation).

It seems fair to conclude from this that information about the dynamic relationships between the inferred hidden states is useful for inferring view-based object models from data—this is the advantage that the SS-BD-IHMM has over the "DPMM + appearance" approach. Four of the "hacks" described in §6.4.9 do try to group hidden states based on the information contained in the inferred transition matrices. Figure 6.30 makes it clear that this method is unhelpful,[5] and Figure 6.31 helps explain why: the rows are simply too diverse to support clustering via the proposed method; instead, massive oversegmentation of the hidden states occur. A variety of concentration parameter values were attempted for this clustering method, but none avoided overclustering.

One final training set labeling result that deserves elaboration is the performance differences between approaches that derive hidden states via DPMM clustering and approaches that use the IHMM for this. IHMM approaches tend to derive more hidden states, since the input observation dynamics provide a useful cue that discourages two similar-looking object views from being clustered together. Having more object views greatly decreases the chances that a single view will be associated with observations from multiple objects, which explains why the "IHMM + appearance" model beats the "DPMM + appearance"

[5] We SS-BD-IHMM inventors could perhaps be accused of coming up with a bad straw-man to pit against our fancy model; honestly, we thought it would do a lot better!

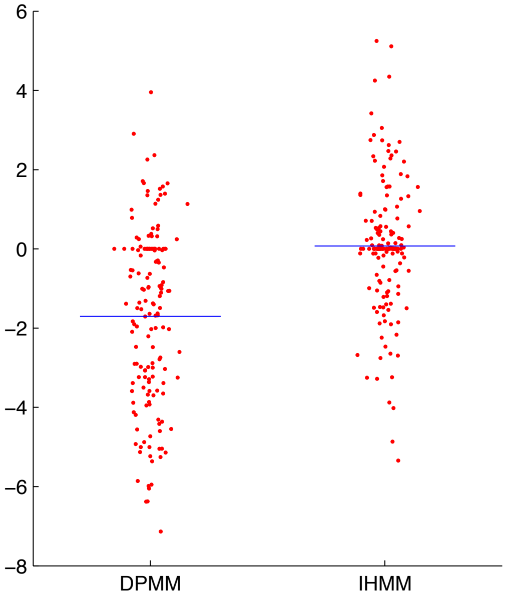


FIGURE 6.32: A visual comparison of the test set likelihood performance of the DPMM and IHMM-derived models against the SS-BD-IHMM. **To interpret:** Points about 0 indicate performance gains over the SS-BD-IHMM. If y is the y-coordinate of a point and p is the likelihood of the test set data according to the corresponding DPMM or IHMM-derived model, then the BD-IHMM likelihood for the test set data is $p^{\exp(y)}$. **Blue lines** indicate the scatter plot means: -1.70 and 0.074 from left to right. As detailed in the text, for these trials, the SS-BD-IHMM performs significantly better than the DPMM-derived model but not the IHMM-derived model.

approach. On the other hand, more object views increases the opportunities for oversegmentation based on the dynamics, which explains why “DPMM + dynamics” and “DPMM + both” beat “IHMM + dynamics” and “IHMM + both” respectively.

Figure 6.32 compares test set likelihood scores for models inferred by the SS-BD-IHMM with those created by DPMM and IHMM-based methods. These comparisons come from the same 158 datasets and models for which sub-behavior identification performance was measured and discussed above. Recall that all three DPMM-derived models start out the same way—they construct an HMM based on the DPMM clustering results—and so even before sub-behaviors are identified, the parts of the models relevant to predicting future data are already learned. The same can be said for the three IHMM-based approaches—just replace “DPMM” with “IHMM” in the previous statement. For this reason, we can group DPMM-derived models together and IHMM-derived models together in this comparison.

We see first that DPMM-derived models are vastly worse than SS-BD-IHMM-inferred models at predicting the test set data, a significant result (Wilcoxon signed-rank test, $p = 1.1 \times 10^{-16}$). IHMM-based models fare better than we might have expected—over the 158 datasets, there was no significant difference between SS-BD-IHMM test set log likelihood scores and those of the IHMM-based approaches (Wilcoxon signed-rank test, $p = 0.25$). To look at Figure 6.32, it might even be the case that the IHMM-based models are doing even slightly better.

Is this a concern for the SS-BD-IHMM? Probably not a great one. The SS-BD-IHMM is a considerably more complicated model than the IHMM, and perhaps it should not be entirely surprising that it could overfit the data just slightly. Another explanation could be that because the SS-BD-IHMM and the IHMM were given the same number of Gibbs sampling iterations to infer their parameters from the data, the simpler IHMM received more training relative to its own complexity than the SS-BD-IHMM did. Either way, the sub-behavior identification performance of the SS-BD-IHMM is far better than that of the IHMM-based method.

6.4.11 Experiment 4: grayscale data

So far, our view-based object model learning experiments have used color object video data. Color is a distinctive cue for object recognition, and while our results have already shown that the structural assumptions expressed in the SS-BD-IHMM prior are beneficial for this dataset, some readers may be concerned that color makes objects "too easy" to discriminate in our data. In this section we repeat our first object model learning experiment (§6.4.4) with a grayscale object video dataset created by averaging the color channels in our color object video dataset. Figure 6.33 shows still images from this new dataset.

The observation model schema and parameters (§6.4.2) are left unchanged from the experiments on color data, save for the fact that images and observation models in the new data are now 1800-dimensional (30 × 30 pixels of intensity data and alpha mask each), and observation model generating distributions are now single-dimensional (one dimension for intensity). The modification function parameter σ_{mod} (c.f. §6.4.3 and Equation 6.5) was changed to 12.8, the value we determined would make the modification function for the grayscale data be as similar as possible to the modification function for the color data. We arrived at this number by creating candidate observation models from frames in both datasets, then finding the σ_{mod} that minimizes the least squares difference between (a) the

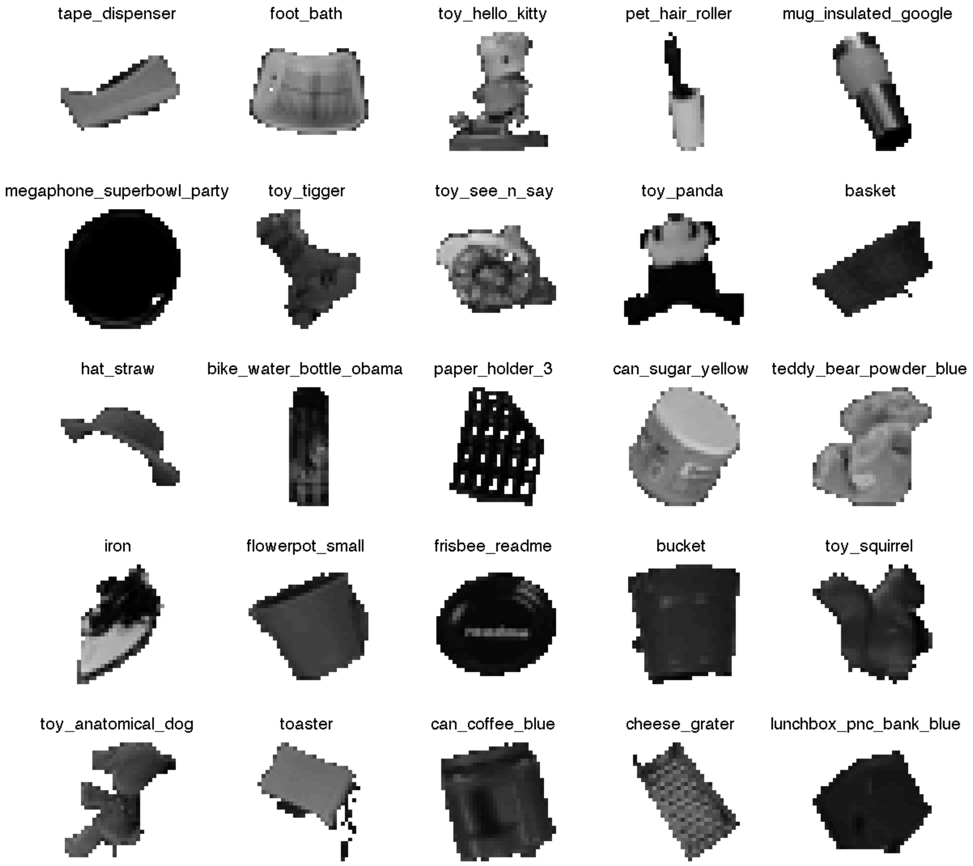


FIGURE 6.33: Still images from input movies taken for the 25 objects used in the grayscale object model learning experiment in §6.4.11. These movies were created by averaging the color channels in the data used for the other object model learning experiments—by removing color, the intention is to create a more difficult dataset, with objects that are more difficult to distinguish. Readers who are viewing this document in color may compare the still images above with their color counterparts in Figure 6.18.

sum of the exponents in Equation 6.5 for the color data and (b) the sum of those same exponents for the grayscale data. All other details of the experiment were left unchanged.

6.4.12 Results of Experiment 4

We conducted 335 runs of the experiment as described, employing the same training and evaluation procedure as Experiment 1 (§6.4.6).

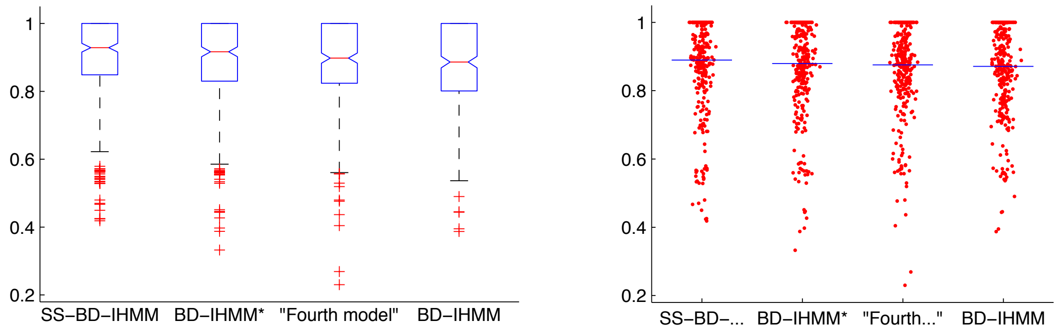


FIGURE 6.34: Box and whisker plots (left) and scatter plots (right) with mean values (blue lines) of adjusted Rand indices for the four test conditions in the fourth (grayscale) object model learning experiment (§6.4.11). As detailed in the text, SS-BD-IHMM object organization performance is significantly better than the other models. Two scores are out of the scale of these plots: on one trial, the SS-BD-IHMM assigned nearly all of the training data to only one object model, yielding an adjusted Rand index score of 0.005; on another trial, the BD-IHMM* assigned the training data to only one object model, yielding an adjusted Rand index score of 0. Note that the vertical axis scales of these plots are slightly different than those used in Figure 6.23 to present similar results from Experiment 1.

Figure 6.34 uses the adjusted Rand index to compare the four models' abilities to organize the training data into object models. The SS-BD-IHMM, with a median adjusted Rand index of 0.929, performed significantly better than the BD-IHMM* (Wilcoxon signed-rank test, $p = 0.024$), the ”fourth model” ($p = 8.5 \times 10^{-4}$), and the BD-IHMM ($p = 5.7 \times 10^{-5}$). The BD-IHMM* was also significantly better than the BD-IHMM ($p = 0.02$), but none of the other result pairings were determined to be significantly different after this number of trials. Nevertheless, we may still conclude that in the absence of a cue as strong as color, all of the structural assumptions of the SS-BD-IHMM prior allows for better results in view-based object model learning than models with only a subset of those assumptions.

Figure 6.35 compares the test-set likelihood performance of models by the SS-BD-IHMM, the BD-IHMM*, and the "fourth model" with models inferred by the BD-IHMM. In contrast to Experiment 1, none of the models showed performance that was significantly different from the BD-IHMM (Wilcoxon signed-rank test; SS-BD-IHMM vs. BD-IHMM $p = 0.94$, BD-IHMM* vs. BD-IHMM $p = 0.71$, "fourth model" vs. BD-IHMM $p = 0.64$). One reason for this result could be that because the grayscale appearances of objects are more similar than the color appearances, the consequences of using an inappropriate observation model to describe the appearance of an object in the video are not as severe for grayscale data as they are for color data. In terms of predicting the appearance and behavior of grayscale object video data, all four models appear to yield similar results.

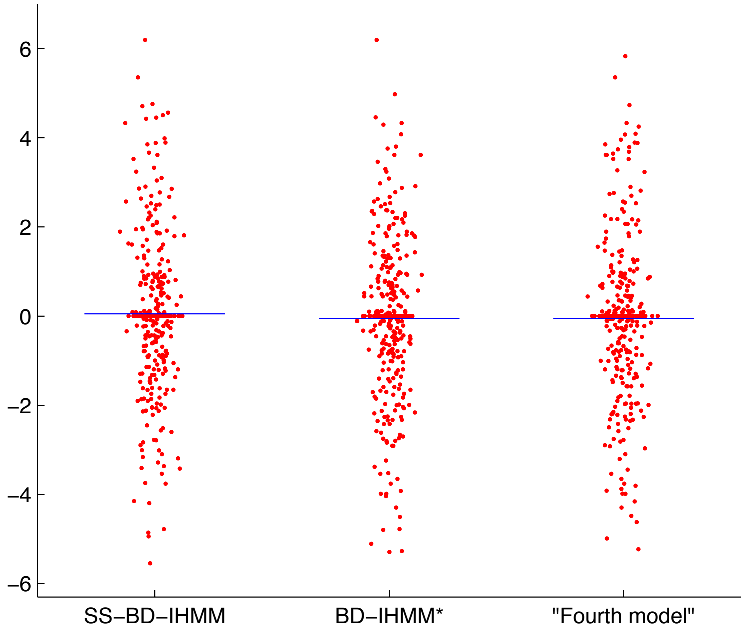


FIGURE 6.35: A visual comparison of the test set likelihood performance of the SS-BD-IHMM, the BD-IHMM*, and the "fourth model" against the BD-IHMM. in the fourth (grayscale) object model learning experiment (§6.4.11). **To interpret:** Points above 0 indicate performance gains over the BD-IHMM. If y is the y-coordinate of a point and p is the likelihood of the test set data according to the corresponding complex model (not shown), then the BD-IHMM likelihood for the test set data is $p^{\exp(y)}$. Grand differences in likelihood are not uncommon in HMM settings. **Blue lines** indicate the scatter plot means: 0.0501, -0.0525, and -0.0520 from left to right. As detailed in the text, none of these models were determined to show significant differences in test set log likelihood.

Despite this similarity, the SS-BD-IHMM's significantly better results for identifying the distinct objects within the video data, even in the absence of color, a very useful cue for this task, give us continued confidence that the additional prior structure within the SS-BD-IHMM is beneficial.

6.4.13 Conclusions

This section has explored the suitability of the SS-BD-IHMM for deriving view-based object models from videos of objects, the task for which the model was originally conceived.

In basic terms, we can deem the model a success: the experiments show that the SS-BD-IHMM does identify distinct objects in video, and that it can do so when there are numerous objects. There is a more subtle conclusion as well: with the experimental results in §6.4.6, it is evident that a model with a more finely structured prior than the BD-IHMM's basic preference for block-diagonal transition matrices achieves performance gains. It also seems plausible (although it was only demonstrated to statistical significance for grayscale data) that a model facilitating both visual similarity-based transition dynamics *and* grouping through appearance cues is better at delimiting objects in videos than a model with only one or neither of these properties. Finally, we determine that the SS-BD-IHMM is better at this same task than a number of obvious, simpler approaches.

These specific conclusions about the SS-BD-IHMM and other BD-IHMM-derived models suggest a few broader, more general notions. The first is that no matter how one might choose to derive object models from video, the idea of applying prior assumptions with all the structure you need and nothing more is powerful. The SS-BD-IHMM learns object models as well as it does because it omits constraining assumptions (e.g. there are exactly 20 objects) while keeping useful ones (e.g. the appearance of object views correlates with the probability of transitions between them). A second, associated notion is that because Bayesian nonparametric approaches to organizing information adopt so few constraining assumptions, they are worthy of serious consideration when engineering a useful, principled system for unsupervised learning of object models.

We should be careful not to claim that this section represents a complete investigation of the application of SS-BD-IHMM techniques to view-based object model learning. For starters, we have not tested how suitable the learned object models are for applications of object models, recognition being chief among these. Additional considerations that should be addressed for practical applications include the nature of the transition similarity function—should we stick with Equation 6.5, or try something new? Should we specify parameters like σ_{mod} beforehand or learn them? It is also likely that the image patch exemplar-based observation models described in §6.4.2 are not the most suitable for practical applications—what should we use instead?

Indeed, some considerations for practical object model learning are not even addressed in the SS-BD-IHMM framework at all. As described, the SS-BD-IHMM assumes that its entire input consists of properly-registered, non-occluded, valid observations of objects, one object at a time. Under the controlled conditions in which our data was acquired, this assumption is appropriate. In settings where one wishes to "mine" the objects from a video (c.f. [49]), such high-quality observations will tend to be an exception rather than the rule. If an object is somehow isolated from the video frame of a natural movie, we can depend on incorrectly delimited object boundaries, occlusions, sub-optimal lighting conditions, and so on. A system for learning object models in this setting must determine what of the incoming observational data is reliable, or better yet, work in concept with the system that derives object observations from video frames to incorporate object knowledge into the video interpretation process.

In spite of all of this, we feel that the development of the SS-BD-IHMM efforts shown here have some value for the vision research community. A system that learns the appearance of objects as humans and objects do will eventually need to face the problem that the SS-BD-IHMM tackles directly—that of organizing well-structured visual object models from limited, unstructured data giving examples of how the appearances of various objects change over time. Hopefully this section portrays a worthwhile new direction toward approaches that solve this problem.

6.5 Relational data

Our final demonstration brushes on an interesting new direction for some of the model machinery in the BD-IHMM. Recall that in the BD-IHMM, the likelihood function for hidden state block label inference, namely $p(\boldsymbol{z} \,|\, \boldsymbol{v}, \boldsymbol{\beta}, \boldsymbol{\rho}, \alpha_0, \xi)$, does not actually require the full hidden state trajectory $\boldsymbol{v}$: instead, it is sufficient to know only the number of transitions c_{mn} between pairs of hidden states m, n. These c_{mn} comprise a matrix $\boldsymbol{c}$ of non-negative integer values, and it is this matrix that the technique of Chapter 4 probes for block structure.

Non-negative integer matrices are hardly unique to hidden Markov models and time series analysis. A much broader (although still not encompassing) class of applications can be envisioned by imagining situations in which it is useful to count the number of times that some event involving two distinct entities occurs. In the hidden Markov model case, the entities are states, and the events are the transitions between them. A different application might consider people in a social network as the entities and count the number of emails or messages between pairs of individuals. In computer vision, we might be interested in the number of times that pairs of image features coincide within some region of interest. There can even be asymmetric sorts of countable relationships between different kinds of entities: we might count the number of times that a person listens to different songs in a music library. All of these entities exhibit some kind of countable relationship between

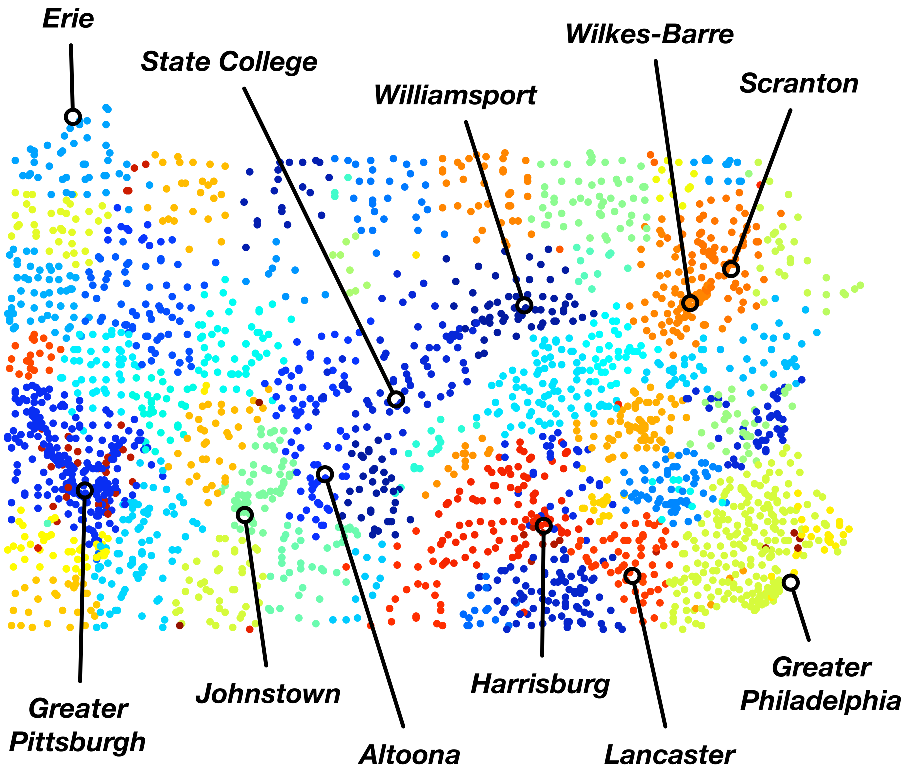


FIGURE 6.36: “Segmentsylvania” map showing locations of Pennsylvania municipalities in the U.S. Census Bureau’s 2000 daily commuter dataset [2]. Colors show vertex groupings achieved by applying the partitioning approach described in Chapter 4 to the commute graph. Nearly all groupings are spatially localized; some colors are reused or similar due to the large number of clusters. More details appear in §6.5.

each other—transitions, conversations, co-occurrence, listenings—which suggests that a technique for partitioning non-negative integer matrices might be a useful addition to the growing toolbox of methods for analyzing relational data, or data that describe relationships between pairs or larger sets of entities. This is an application area which has seen modest application of Bayesian nonparametric methods (e.g. [111]), though interest appears to be growing.

In this demonstration we consider a countable relationship between pairs of entities. The entities are some 2,580 municipalities throughout the state of Pennsylvania, and the relation counts c_{mn} are the number of commuters interviewed by the U.S. Census Bureau who travel between the municipalities indexed by m and n [2]. Figure 6.36 shows the counts matrix assembled by the Bureau, along with the partitioning results achieved on this matrix via the methods of Chapter 4. While we do not attempt to quantify here how “good” this partitioning result is, there is a considerable qualitative appeal—for just about every block in the partition, the municipalities belonging to that block are geographically localized and centered around middle- and large-sized urban centers. This occurs in spite of the fact that the algorithm makes no use of geographic information about the municipalities. Furthermore, there is a staggering disparity between the popularity of some places (many people commute to e.g. Philadelphia; far fewer commute to Scalp Level, PA[6]), and yet meaningful clusters emerge. We believe that this result should motivate further investigation into applications of BD-IHMM derived models toward relational data.

[6] An unincorporated community southeast of Johnstown, PA, in case you had wondered.

Chapter 7

Concluding Notes

Having commented on the outcome of our motivating experiment in §6.4.13, and having used up over two hundred pages besides, we think it best to conclude with just a few remarks on future directions.

First, as §6.5 suggests, the model and inference machinery we have developed for our unsupervised method for learning hidden Markov models may be useful for additional problem settings not related to time series. This may merit further investigation.

Next, although Chapter 5 introduces a more general framework for flexible structural modification of hierarchical Dirichlet process-derived hidden Markov models, we have not investigated the problem of inferring the specifics of any structural modification besides blocks of hidden states. The SS-BD-IHMM experiments in §6.3 and §6.4 use Gaussian kernel-based modification functions with fixed σ parameters, and these remain fixed throughout inference. A useful new direction would be to infer the correct σ value, perhaps even separate σ values for different blocks or states. More fundamentally, an investigation into inferring modification functions generally would likely increase the flexibility and descriptive power of our technique in useful ways.

Although in this thesis we adapt an approach based on the hierarchical Dirichlet process, there is not necessarily an inherent advantage to using this model as a foundation for flexible hidden Markov model inference over other Bayesian nonparametric methods. The recent explosion in new approaches for dependent random measures (c.f. §1.3) now offers a number of promising new models that beg to be adapted to this end. We find the technique described in [84] to show particular promise.

Finally, we believe the motivating problem we have confronted in this thesis—distilling arbitrary numbers of discrete object models from our visual experiences of objects over time—has not been satisfactorily addressed by the neural computation literature. Central to our concern is the lack of attention paid to how the visual system might deduce whether an object present in the visual field is novel or familiar—and thus whether the visual experience should be used to create a new object model or refine an existing one. We do not claim that Bayesian nonparametric methods are the secret to how the visual system solves this problem, but we propose that some of the flexible prior modeling assumptions that may be achieved through these methods have analogs in neural systems. Whatever the case, we eagerly await advancements in our understanding of how the brain organizes complex, high-level information.

Bibliography

[1] Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476): 1566–1581, 2006. ISSN 0162-1459. doi: 10.1198/016214506000000302. URL http://pubs.amstat.org/doi/abs/10.1198/016214506000000302.

[2] US Census Bureau Systems Support Division and Population Division. Differences between the 1990 and 2000 census questionnaires. http://www.census.gov/population/www/cen2000/commuting/mcdworkerflow.html, 2000. URL http://www.census.gov/population/www/cen2000/commuting/mcdworkerflow.html.

[3] Louise Hainline. The development of basic visual abilities. In *Perceptual Development: Visual, Auditory and Speech Perception in Infancy*, pages 1–50. Psychology Press, Sussex, 1998.

[4] Elizabeth S Spelke. Principles of object perception. *COGNITIVE SCIENCE*, 14: 29—56, 1990. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.95.507.

[5] Amy Needham. Object recognition and object segregation in 4.5-Month-Old infants. *Journal of Experimental Child Psychology*, 78:3–22, 2001. URL http://www.ingentaconnect.com/content/ap/ch/2001/00000078/00000001/art02598.

[6] Lincoln G. Craton. The development of perceptual completion abilities: Infants' perception of stationary, partially occluded objects. *Child Development*, 67(3):890–904, 1996. doi: 10.1111/j.1467-8624.1996.tb01771.x. URL http://dx.doi.org/10.1111/j.1467-8624.1996.tb01771.x.

[7] A Needham and R Baillargeon. Object segregation in 8-month-old infants. *Cognition*, 62(2):121–149, February 1997. ISSN 0010-0277. URL http://www.ncbi.nlm.nih.gov/pubmed/9141904. PMID: 9141904.

[8] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. ISSN 0018-9219.

[9] Leonard E. Baum, Ted Petrie, George Soules, and Norman Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The Annals of Mathematical Statistics*, 41(1):164–171, February 1970. ISSN 00034851. URL http://www.jstor.org/stable/2239727. ArticleType: primary_article / Full publication date: Feb., 1970 / Copyright © 1970 Institute of Mathematical Statistics.

[10] Hugo Steinhaus. Sur la division des corps matériels en parties. *Bull. Acad. Polon. Sci. Cl. III*, 4:801–804, 1957.

[11] Victor Hasselblad. Estimation of finite mixtures of distributions from the exponential family. *Journal of the American Statistical Association*, 64(328):1459–1471, December 1969. ISSN 01621459. URL http://www.jstor.org/stable/2286083. ArticleType: primary_article / Full publication date: Dec., 1969 / Copyright © 1969 American Statistical Association.

[12] 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 (Methodological)*, 39(1):1–38, 1977. ISSN 00359246. URL http://www.jstor.org/stable/2984875. ArticleType: primary_article / Full publication date: 1977 / Copyright © 1977 Royal Statistical Society.

[13] H. Akaike. A new look at the statistical model identification. *Automatic Control, IEEE Transactions on*, 19(6):716–723, 1974. ISSN 0018-9286.

[14] Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6 (2):461–464, 1978. URL http://projecteuclid.org/DPubS?service=UI&version=1.0&verb=Display&handle=euclid.aos/1176344136.

[15] Carl Rasmussen. The infinite gaussian mixture model. In *Neural Information Processing Systems 13*, Vancouver, 1999. MIT Press.

[16] Matthew J Beal, Zoubin Ghahramani, and Carl Rasmussen. The infinite hidden markov model. In *Neural Information Processing Systems 14*, Vancouver, 2001. MIT Press.

[17] Paul Viola and Michael Jones. Robust real-time object detection. *INTERNATIONAL JOURNAL OF COMPUTER VISION*, 2001. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.23.2751.

[18] David G. Lowe. Distinctive image features from Scale-Invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, November 2004. doi: 10.1023/B:VISI.0000029664.99615.94. URL http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94.

[19] Bodo Rosenhahn, Thomas Brox, and Joachim Weickert. Three-Dimensional shape knowledge for joint image segmentation and pose tracking. *International Journal of Computer Vision*, 73(3):243–262, July 2007. doi: 10.1007/s11263-006-9965-3. URL http://dx.doi.org/10.1007/s11263-006-9965-3.

[20] Han-Pang Chiu, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Automatic Class-Specific 3D reconstruction from a single image. Technical report MIT-CSAIL-TR-2009-008, MIT, CSAIL, Cambridge, Massachusetts, February 2009. URL http://dspace.mit.edu/handle/1721.1/44615.

[21] J. J. Koenderink and A. J. Doorn. The internal representation of solid shape with respect to vision. *Biological Cybernetics*, 32(4):211–216, December 1979. doi: 10.1007/BF00337644. URL http://dx.doi.org/10.1007/BF00337644.

[22] Kevin W. Bowyer and Charles R. Dyer. Aspect graphs: An introduction and survey of recent results. *International Journal of Imaging Systems and Technology*, 2(4):315–328, 1990. doi: 10.1002/ima.1850020407. URL http://dx.doi.org/10.1002/ima.1850020407.

[23] Christopher M. Cyr and Benjamin B. Kimia. A Similarity-Based Aspect-Graph approach to 3D object recognition. *International Journal of Computer Vision*, 57(1):5–22, April 2004. doi: 10.1023/B:VISI.0000013088.59081.4c. URL http://dx.doi.org/10.1023/B:VISI.0000013088.59081.4c.

[24] T. Denton, M.F. Demirci, J. Abrahamson, A. Shokoufandeh, and S. Dickinson. Selecting canonical views for view-based 3-D object recognition. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 2, pages 273–276 Vol.2, 2004. ISBN 1051-4651. doi: 10.1109/ICPR.2004.1334159.

[25] T.F. Cootes, K. Walker, and C.J. Taylor. View-based active appearance models. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, pages 227–232, 2000. doi: 10.1109/AFGR.2000.840639.

[26] A. Thomas, V. Ferrar, B. Leibe, T. Tuytelaars, B. Schiel, and L. Van Gool. Towards Multi-View object class detection. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1589–1596, 2006. ISBN 1063-6919.

[27] M. Ozuysal, V. Lepetit, and P. Fua. Pose estimation for category specific multiview object localization. In *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, volume 0, pages 778–785, Los Alamitos, CA, USA, 2009. IEEE Computer Society. ISBN 978-1-4244-3992-8. doi: http://doi.ieeecomputersociety.org/10.1109/CVPRW.2009.5206633.

[28] Hao Su, Min Sun, Li Fei-Fei, and Silvio Savarese. Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *International Conference in Computer Vision*, 2009. URL http://www.eecs.umich.edu/~silvio/papers/SuSunLiSavarese_ICCV2009.pdf.

[29] Min Sun, Hao Su, Silvio Savarese, and Li Fei-Fei. A Multi-View probabilistic model for 3D object classes. In *Proc. Computer Vision and Pattern Recognition*, 2009.

[30] H.H. Bulthoff, C. Wallraven, and A. Graf. View-based dynamic object recognition based on human perception. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 3, pages 768–776 vol.3, 2002. ISBN 1051-4651. doi: 10.1109/ICPR.2002.1048105.

[31] M. Seibert and A.M. Waxman. Adaptive 3-D object recognition from multiple views. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 14(2):107–124, 1992. ISSN 0162-8828.

[32] Tai Sing Lee and David Mumford. Hierarchical bayesian inference in the visual cortex. *Journal of the Optical Society of America. A, Optics, Image Science, and Vision*, 20(7):1434–1448, July 2003. ISSN 1084-7529. URL http://www.ncbi.nlm.nih.gov/pubmed/12868647. PMID: 12868647.

[33] Boris Epshtein, Ita Lifshitz, and Shimon Ullman. Image interpretation by a single bottom-up top-down cycle. *Proceedings of the National Academy of Sciences*, 105(38):14298–14303, 2008. doi: 10.1073/pnas.0800968105. URL http://www.pnas.org/content/105/38/14298.abstract.

[34] Keiji Tanaka. Inferotemporal cortex and object vision. *Annual Review of Neuroscience*, 19(1):109–139, 1996. ISSN 0147-006X. doi: 10.1146/annurev.ne.19.030196.000545.

URL http://arjournals.annualreviews.org/doi/abs/10.1146/annurev.ne.19.030196.000545?url_ver=Z39.88-2003&rfr_id=ori:rid:crossref.org&rfr_dat=cr_pub%3dncbi.nlm.nih.gov.

[35] Hans Op de Beeck, Johan Wagemans, and Rufin Vogels. Inferotemporal neurons represent low-dimensional configurations of parameterized shapes. *Nat Neurosci*, 4 (12):1244–1252, December 2001. ISSN 1097-6256. doi: 10.1038/nn767. URL http://dx.doi.org/10.1038/nn767.

[36] Yukako Yamane, Eric T. Carlson, Katherine C. Bowman, Zhihong Wang, and Charles E. Connor. A neural code for three-dimensional object shape in macaque inferotemporal cortex. *Nature neuroscience*, 11(11):1352–1360, November 2008. ISSN 1097-6256. doi: 10.1038/nn.2202. PMID: 18836443 PMCID: 2725445.

[37] D I Perrett, E T Rolls, and W Caan. Visual neurones responsive to faces in the monkey temporal cortex. *Experimental Brain Research. Experimentelle Hirnforschung. Expérimentation Cérébrale*, 47(3):329–342, 1982. ISSN 0014-4819. URL http://www.ncbi.nlm.nih.gov/pubmed/7128705. PMID: 7128705.

[38] Doris Y. Tsao, Winrich A. Freiwald, Roger B. H. Tootell, and Margaret S. Livingstone. A cortical region consisting entirely of Face-Selective cells. *Science*, 311(5761):670–674, February 2006. doi: 10.1126/science.1119983. URL http://www.sciencemag.org/cgi/content/abstract/311/5761/670.

[39] Andrew H. Bell, Fadila Hadj-Bouziane, Jennifer B. Frihauf, Roger B. H. Tootell, and Leslie G. Ungerleider. Object representations in the temporal cortex of monkeys and humans as revealed by functional magnetic resonance imaging. *J Neurophysiol*, 101(2):688–700, February 2009. doi: 10.1152/jn.90657.2008. URL http://jn.physiology.org/cgi/content/abstract/101/2/688.

[40] N K Logothetis, J Pauls, H H Bülthoff, and T Poggio. View-dependent object recognition by monkeys. *Current Biology: CB*, 4(5):401–414, May 1994. ISSN 0960-9822. URL http://www.ncbi.nlm.nih.gov/pubmed/7922354. PMID: 7922354.

[41] N K Logothetis and J Pauls. Psychophysical and physiological evidence for viewer-centered object representations in the primate. *Cerebral Cortex (New York, N.Y.: 1991)*, 5(3):270–288, June 1995. ISSN 1047-3211. URL http://www.ncbi.nlm.nih.gov/pubmed/7613082. PMID: 7613082.

[42] MC Booth and ET Rolls. View-invariant representations of familiar objects by neurons in the inferior temporal visual cortex. *Cereb. Cortex*, 8(6):510–523, September 1998. doi: 10.1093/cercor/8.6.510. URL http://cercor.oxfordjournals.org/cgi/content/abstract/8/6/510.

[43] D. I. Perrett, J. K. Hietanen, M. W. Oram, P. J. Benson, and E. T. Rolls. Organization and functions of cells responsive to faces in the temporal cortex [and discussion]. *Philosophical Transactions: Biological Sciences*, 335(1273):23–30, 1992. ISSN 09628436. URL http://www.jstor.org/stable/55471. ArticleType: primary_article / Issue Title: Processing the Facial Image / Full publication date: Jan. 29, 1992 / Copyright © 1992 The Royal Society.

[44] T Vetter, A Hurlbert, and T Poggio. View-based models of 3D object recognition: invariance to imaging transformations. *Cerebral Cortex (New York, N.Y.: 1991)*, 5(3): 261–269, June 1995. ISSN 1047-3211. URL http://www.ncbi.nlm.nih.gov/pubmed/7613081. PMID: 7613081.

[45] Matthew D Hoffman, Perry R Cook, and David M Blei. Data-Driven recomposition using the hierarchical dirichlet process hidden markov model. In *2008 International Computer Music Conference (ICMC)*, Belfast, 2008.

[46] Kuniyoshi Sakai and Yasushi Miyashita. Neural organization for the long-term memory of paired associates. *Nature*, 354(6349):152–155, November 1991. doi: 10.1038/354152a0. URL http://dx.doi.org/10.1038/354152a0.

[47] Volodya Yakovlev, Stefano Fusi, Elisha Berman, and Ehud Zohary. Inter-trial neuronal activity in inferior temporal cortex: a putative vehicle to generate long-term visual associations. *Nat Neurosci*, 1(4):310–317, 1998. ISSN 1097-6256. doi: 10.1038/1131. URL http://dx.doi.org/10.1038/1131.

[48] J. Sivic and A. Zisserman. Video data mining using configurations of viewpoint invariant regions. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–488–I–495 Vol.1, 2004. ISBN 1063-6919. doi: 10.1109/CVPR.2004.1315071.

[49] Josef Sivic, Frederik Schaffalitzky, and Andrew Zisserman. Object level grouping for video shots. *Int. J. Comput. Vision*, 67(2):189–210, 2006. URL http://portal.acm.org/citation.cfm?id=1132117.

[50] Fred Rothganger, Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Segmenting, modeling, and matching video clips containing multiple moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3):477–491, 2007. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.57. URL http://ieeexplore.ieee.org/Xplore/login.jsp?url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel5%2F34%2F4069251%2F04069263.pdf%3Farnumber%3D4069263&authDecision=-203.

[51] Wolfgang Metzger. *Laws of seeing*. MIT Press, Cambridge Mass., 2006. ISBN 9780262134675. German title: Gesetze de Sehens. Lothar Spillmann, Steven Lehar, Mimsey Stromeyer, and Michael Wertheimer, translators.

[52] S. M. Stringer, E. T. Rolls, and J. M. Tromans. Invariant object recognition with trace learning and multiple stimuli present during training. *Network: Computation in Neural Systems*, 18(2):161, 2007. ISSN 0954-898X. doi: 10.1080/09548980701556055. URL http://www.informaworld.com/10.1080/09548980701556055.

[53] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, April 1980. doi: 10.1007/BF00344251. URL http://dx.doi.org/10.1007/BF00344251.

[54] Maximilian Riesenhuber and Tomaso Poggio. Hierarchical models of object recognition in cortex. *NATURE NEUROSCIENCE*, 2:1019—1025, 1999. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.94.3249.

[55] Joshua B. Tenenbaum and William T. Freeman. Separating style and content with bilinear models. *Neural Comput.*, 12(6):1247–1283, 2000. URL http://portal.acm.org/citation.cfm?id=1121518.

[56] David B. Grimes and Rajesh P. N. Rao. Bilinear sparse coding for invariant vision. *Neural Computation*, 17(1):47–73, 2005. doi: 10.1162/0899766052530893. URL http://dx.doi.org/10.1162/0899766052530893.

[57] Xu Miao and Rajesh P. N. Rao. Learning the lie groups of visual invariance. *Neural Computation*, 19(10):2665–2693, October 2007. doi: 10.1162/neco.2007.19.10.2665. URL http://dx.doi.org/10.1162/neco.2007.19.10.2665.

[58] B W Mel. SEEMORE: combining color, shape, and texture histogramming in a neurally inspired approach to visual object recognition. *Neural Computation*, 9(4):

777–804, May 1997. ISSN 0899-7667. URL http://www.ncbi.nlm.nih.gov/pubmed/9161022. PMID: 9161022.

[59] M. Varma and D. Ray. Learning the discriminative Power-Invariance Trade-Off. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, 2007. ISBN 1550-5499.

[60] Peter Földiák. Learning invariance from transformation sequences. *Neural Comput.*, 3(2):194–200, 1991. URL http://portal.acm.org/citation.cfm?id=110089.

[61] Suzanna Becker. Implicit learning in 3D object recognition: The importance of temporal context. *Neural Computation*, 11(2):347–374, February 1999. doi: 10.1162/089976699300016683. URL http://dx.doi.org/10.1162/089976699300016683.

[62] Wolfgang Einhäuser, Jörg Hipp, Julian Eggert, Edgar Körner, and Peter König. Learning viewpoint invariant object representations using a temporal coherence principle. *Biological Cybernetics*, 93(1):79–90, July 2005. doi: 10.1007/s00422-005-0585-8. URL http://dx.doi.org/10.1007/s00422-005-0585-8.

[63] S.M. Stringer and E.T. Rolls. Learning transform invariant object recognition in the visual system with multiple stimuli present during training. *Neural Networks*, 21(7):888–903, September 2008. ISSN 0893-6080. doi: 10.1016/j.neunet.2007.11.004. URL http://www.sciencedirect.com/science/article/B6T08-4S7J5F9-1/2/f372c76dbcc313fc555b99446df8a972.

[64] Hossein Mobahi, Ronan Collobert, and Jason Weston. Deep learning from temporal coherence in video. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 737–744, Montreal, Quebec, Canada, 2009. ACM. ISBN 978-1-60558-516-1. doi: 10.1145/1553374.1553469. URL http://portal.acm.org/citation.cfm?id=1553469.

[65] Laurenz Wiskott and Terrence J. Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, April 2002. doi: 10.1162/089976602317318938. URL http://dx.doi.org/10.1162/089976602317318938.

[66] Charles F. Cadieu and Bruno A. Olshausen. Learning transformational invariants from Time-Varying natural images. In *Computational and Systems Neuroscience (COSYNE) 2008*, Salt Lake City, Utah, 2008.

[67] Thomas Dean. Learning invariant features using inertial priors. *Annals of Mathematics and Artificial Intelligence*, 47(3):223–250, 2006. doi: 10.1007/s10472-006-9039-9. URL http://dx.doi.org/10.1007/s10472-006-9039-9.

[68] Dileep George and Jeff Hawkins. Towards a mathematical theory of cortical microcircuits. *PLoS Comput Biol*, 5(10):e1000532, October 2009. doi: 10.1371/journal.pcbi.1000532. URL http://dx.doi.org/10.1371/journal.pcbi.1000532.

[69] Larry Wasserman. *All of nonparametric statistics*. Springer, New York, 2006. ISBN 9780387251455.

[70] Emanuel Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962. ISSN 0003-4851. doi: 10.1214/aoms/1177704472. URL http://projecteuclid.org/DPubS?service=UI&version=1.0&verb=Display&handle=euclid.aoms/1177704472.

[71] Thomas S. Ferguson. A bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230, 1973. URL http://projecteuclid.org/DPubS?service=UI&version=1.0&verb=Display&handle=euclid.aos/1176342360.

[72] Carl Rasmussen. *Gaussian processes for machine learning*. MIT Press, Cambridge Mass., 2006. ISBN 9780262182539.

[73] Peter Orbanz. Construction of nonparametric bayesian models from parametric bayes equations. In *Neural Information Processing Systems 22*, Vancouver, 2009.

[74] Emily B. Fox, Erik B. Sudderth, Michael I. Jordan, and Alan S. Willsky. An HDP-HMM for systems with state persistence. In *25th International Conference on Machine Learning*, pages 312–319, Helsinki, Finland, 2008. ACM. ISBN 978-1-60558-205-4. doi: 10.1145/1390156.1390196. URL http://portal.acm.org/citation.cfm?id=1390196.

[75] Emily B. Fox, Erik B. Sudderth, Michael I. Jordan, and Alan S. Willsky. Nonparametric bayesian learning of switching linear dynamical systems. In *Neural Information Processing Systems 21*, Vancouver, 2008.

[76] François Caron, Manuel Davy, and Arnaud Doucet. Generalized polya urn for time-varying dirichlet processes. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, volume 23, 2007.

[77] Tianbing Xu, Zhongfei (Mark) Zhang, Philip S. Yu, and Bo Long. Evolutionary clustering by hierarchical dirichlet process with hidden markov state. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pages 658–667. IEEE Computer Society, 2008. ISBN 978-0-7695-3502-9. URL http://portal.acm.org/citation.cfm?id=1511420.

[78] Amr Ahmed and Eric Xing. Dynamic Non-Parametric mixture models and the recurrent chinese restaurant process: with applications to evolutionary clustering. In *Proceedings fo the Eighth SIAM International Conference on Data Mining*, 2008.

[79] Lu Ren, David Dunson, and Lawrence Carin. The dynamic hierarchical dirichlet process. In *Proceedings of the 25th International Conference on Machine Learning*, Helsinki, Finland, 2008.

[80] Iulian Pruteanu-Malinici, Lu Ren, John Paisley, Eric Wang, and Lawrence Carin. Hierarchical bayesian modeling of topics in Time-Stamped documents. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99(1), 2009. ISSN 0162-8828. doi: http://doi.ieeecomputersociety.org/10.1109/TPAMI.2009.125.

[81] Jason A. Duan, Michele Guindani, and Alan E. Gelfand. Generalized spatial dirichlet process models. *Biometrika*, 94(4):809–825, December 2007. doi: 10.1093/biomet/asm071. URL http://biomet.oxfordjournals.org/cgi/content/abstract/94/4/809.

[82] Ya Xue, David Dunson, and Lawrence Carin. The matrix stick-breaking process for flexible multi-task learning. In *Proceedings of the 24th international conference on Machine learning*, pages 1063–1070, Corvalis, Oregon, 2007. ACM. ISBN 978-1-59593-793-3. doi: 10.1145/1273496.1273630. URL http://portal.acm.org/citation.cfm?id=1273630.

[83] David B. Dunson and Ju-Hyun Park. Kernel stick-breaking processes. *Biometrika*, 95(2):307–323, June 2008. doi: 10.1093/biomet/asn012. URL http://biomet.oxfordjournals.org/cgi/content/abstract/95/2/307.

[84] Vinayak Rao and Yee Whye Teh. Spatial normalized gamma processes. In *Neural Information Processing Systems 22*, Vancouver, 2009.

[85] Yeonseung Chung and David Dunson. The local dirichlet process. *Annals of the Institute of Statistical Mathematics*, 2009. doi: 10.1007/s10463-008-0218-9. URL http://dx.doi.org/10.1007/s10463-008-0218-9.

[86] Kai Ni, Lawrence Carin, and David Dunson. Multi-task learning for sequential data via iHMMs and the nested dirichlet process. In *Proceedings of the 24th international conference on Machine learning*, pages 689–696, Corvalis, Oregon, 2007. ACM. ISBN 978-1-59593-793-3. doi: 10.1145/1273496.1273583. URL http://portal.acm.org/citation.cfm?id=1273583.

[87] Lu Ren, David Dunson, Scott Lindroth, and Lawrence Carin. Dynamic nonparametric bayesian models for analysis of music. To appear in Journal of the American Statistical Association, 2009.

[88] David Blackwell and James B. MacQueen. Ferguson distributions via polya urn schemes. *The Annals of Statistics*, 1(2):353–355, 1973. ISSN 0090-5364. doi: 10.1214/aos/1176342372. URL http://projecteuclid.org/euclid.aos/1176342372.

[89] Jayaram Sethuraman. A constructive definition of dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.

[90] David Blei, Thomas Griffiths, Michael Jordan, and Joshua Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. In *Neural Information Processing Systems 15*, Vancouver, 2003. MIT Press.

[91] Matthew J Beal and Praveen Krishnamurthy. Gene expression time course clustering with countably infinite hidden markov models. In *Twenty-Second Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-06)*, Arlington, Virginia, 2006. AUAI Press.

[92] Zoubin Ghahramani. CMU language technologies institute presentation, August 2008.

[93] Radford M. Neal. Markov chain sampling methods for dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, June 2000. ISSN 10618600. URL http://www.jstor.org/stable/1390653. ArticleType: primary_article / Full publication date: Jun., 2000 / Copyright © 2000 American Statistical Association, Institute of Mathematical Statistics and Interface Foundation of America.

[94] Thomas Stepleton. Understanding the ”Antoniak equation”. Unpublished manuscript, 2008. URL http://www.cs.cmu.edu/~tss/antoniak.pdf.

[95] Hemant Ishwaran and Lancelot F James. Gibbs sampling methods for Stick-Breaking priors. *Journal of the American Statistical Association*, 96(453):161–173, 2001. ISSN 0162-1459. doi: 10.1198/016214501750332758. URL http://pubs.amstat.org/doi/abs/10.1198/016214501750332758.

[96] Siddhartha Chib and Edward Greenberg. Understanding the Metropolis-Hastings algorithm. *The American Statistician*, 49(4):327–335, November 1995. ISSN 00031305. URL http://www.jstor.org/stable/2684568. ArticleType: primary_article / Full publication date: Nov., 1995 / Copyright © 1995 American Statistical Association.

[97] Jurgen Van Gael, Yunus Saatci, Yee Teh, and Zoubin Ghahramani. Beam sampling for the infinite hidden markov model. In *25th International Conference on Machine learning*, pages 1095, 1088, Helsinki, Finland, 2008. ACM. ISBN 978-1-60558-205-4. URL http://dx.doi.org/10.1145/1390156.1390293.

[98] Sonia Jain and Radford 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. ISSN 1061-8600. doi: 10.1198/1061860043001. URL http://pubs.amstat.org/doi/abs/10.1198/1061860043001.

[99] Sajid Siddiqi, Geoffrey Gordon, and Andrew Moore. Fast state discovery for HMM model selection and learning. In *AISTATS Eleventh International Conference on Artificial Intelligence and Statistics*, 2007.

[100] Robert G. Edwards and Alan D. Sokal. Generalization of the Fortuin-Kasteleyn-Swendsen-Wang representation and monte carlo algorithm. *Physical Review D*, 38(6): 2009, 1988. doi: 10.1103/PhysRevD.38.2009. URL http://link.aps.org/abstract/PRD/v38/p2009. Copyright (C) 2009 The American Physical Society; Please report any problems to prola@aps.org.

[101] Adrian Barbu and Song-Chun Zhu. Generalizing Swendsen-Wang to sampling arbitrary posterior probabilities. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(8):1239–1253, 2005. URL http://portal.acm.org/citation.cfm?id=1070811.

[102] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, May 1983. ISSN 00368075. URL http://www.jstor.org/stable/1690046. ArticleType: primary_article / Full publication date: May 13, 1983 / Copyright © 1983 American Association for the Advancement of Science.

[103] Jun S. Liu, Faming Liang, and Wing Hung Wong. The Multiple-Try method and local optimization in metropolis sampling. *Journal of the American Statistical Association*, 95(449):121–134, March 2000. ISSN 01621459. URL http://www.jstor.org/stable/2669532. ArticleType: primary_article / Full publication date: Mar., 2000 / Copyright © 2000 American Statistical Association.

[104] Yuan Qi and Thomas P Minka. Hessian-based markov chain Monte-Carlo algorithms. In *First Cape Cod Workshop on Monte Carlo Methods*, Cape Cod, Massachusetts, September 2002. URL http://www.cs.purdue.edu/homes/alanqi/papers/qi-minka-HMH-AMIT-02.ps.

[105] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, December 1985. doi: 10.1007/BF01908075. URL http://dx.doi.org/10.1007/BF01908075.

[106] Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1073–1080, Montreal, Quebec, Canada, 2009. ACM. ISBN 978-1-60558-516-1. doi: 10.1145/1553374.1553511. URL http://portal.acm.org/citation.cfm?doid=1553374.1553511.

[107] N.C. Maddage. Automatic structure detection for popular music. *IEEE Multimedia*, 13(1):65–77, 2006. ISSN 1070-986X. doi: 10.1109/MMUL.2006.3. URL http://ieeexplore.ieee.org/Xplore/login.jsp?url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel5%2F93%2F33377%2F01580435.pdf%3Farnumber%3D1580435&authDecision=-203.

[108] Tuomas Eerola and Petri Toiviainen. *MIDI Toolbox: MATLAB Tools for Music Research*. University of Jyväskylä, Jyväskylä, Finland, 2004. URL http://www.jyu.fi/musica/miditoolbox/.

[109] Thomas Stepleton, Zoubin Ghahramani, Geoffrey Gordon, and Tai Sing Lee. The block diagonal infinite hidden markov model. In *AISTATS Twelfth International Conference on Artificial Intelligence and Statistics*, Clearwater Beach, Florida, 2009.

[110] Brian Potetz and Tai Sing Lee. Efficient belief propagation for higher-order cliques using linear constraint nodes. *Computer Vision and Image Understanding*, 112(1):39–54, October 2008. URL http://dx.doi.org/10.1016/j.cviu.2008.05.007.

[111] Charles Kemp, Joshua B. Tenenbaum, Thomas L. Griffiths, Takeshi Yamada, and Naonori Ueda. Learning systems of concepts with an infinite relational model. In *Proceedings of the 21st national conference on Artificial intelligence - Volume 1*, pages 381–388, Boston, Massachusetts, 2006. AAAI Press. ISBN 978-1-57735-281-5. URL http://portal.acm.org/citation.cfm?id=1597538.1597600.