Scalable inference of discrete data: user behavior, networks and genetic variation

Prem K. Gopalan

A Dissertation
Presented to the Faculty
of Princeton University
in Candidacy for the Degree
of Doctor of Philosophy

Recommended for Acceptance
by the Department of
Computer Science
Adviser: Prof. David M. Blei

January 2015

Abstract

Recent years have seen explosive growth in data, models and computation. Massive data sets and sophisticated probabilistic models are increasingly used in the fields of high-energy physics, biology, genetics and in personalization applications; however, many statistical algorithms remain inefficient, impeding scientific progress.

In this thesis, we present several efficient statistical algorithms for learning from massive discrete data sets. We focus on discrete data because complex and structured activity such as chromosome folding in three dimensions, human genetic variation, social network interactions and product ratings are often encoded as simple matrices of discrete numerical observations. Our algorithms derive from a Bayesian perspective and lie in the framework of directed graphical models and mean-field variational inference. Situated in this framework, we gain computational and statistical efficiency through modeling insights and through subsampling informative data during inference.

We begin with additive Poisson factorization models for recommending items to users based on user consumption or ratings. These models provide sparse latent representations of users and items, and capture the long-tailed distributions of user consumption. We use them as building blocks for article recommendation models by sharing latent spaces across readership and article text. We demonstrate that our algorithms scale to massive data sets, are easy to implement and provide competitive user recommendations. Then, we develop a Bayesian nonparametric model in which the latent representations of users and items grow to accommodate new data.

In the second part of the thesis, we develop novel algorithms for discovering overlapping communities in large networks. These algorithms interleave non-uniform subsampling of the network with model estimation. Our network models capture the basic ways in which nodes connect to each other, through similarity and popularity, using mixed-memberships representations and generalized linear model formulation.

Finally, we present the TeraStructure algorithm to fit Bayesian models of genetic variation in human populations on tera-sample-sized data sets (10^{12} observed genotypes, e.g, 1M individuals at 1M SNPs). On real genomic data collected from thousands of individuals, TeraStructure is faster than existing methods and recovers the latent population structure with equal accuracy. On genomic data simulated at the tera-sample-size scales, TeraStructure is highly accurate and is the only method that can complete its analysis.

Acknowledgements

Foremost, I thank my advisor David M. Blei. Dave has been an excellent teacher and collaborator in the past four years. His enthusiasm and encouragement helped me march forward in a new field of study. Apart from machine learning, Dave taught me what it means to think and write clearly about research.

I would not be here without Michael J. Freedman teaching and guiding me through my early years as a graduate student in the systems/networking group. Mike set a high bar on the quality of research work, and I've tried to carry those lessons with me. He was extremely generous and supportive in my transition to machine learning.

I also thank the other members of my dissertation committee, Robert Schapire, Jake Hofman and John D. Storey for their insightful comments and suggestions.

I am fortunate to have worked with Professors and colleagues at Princeton and elsewhere, who have been a vital part of my education. John Storey, Emmanuel Abbe, Jake Hofman, David Mimno, Jennifer Rexford, Sean Gerrish, Chong Wang, Dae Il Kim, Erik Sudderth, Laurent Charlin, Erik Nordstrom, Rajesh Ranganath, Francisco Ruiz, Matthew Hoffman, Allison Chaney, Wei Hao, Erez Simony, David Shue, Matvey Arye, Steven Ko have all influenced my research through collaboration and discussions. I particularly thank David Mimno and Sean Gerrish for their mentorship during my initial attempts at machine learning research.

I owe many thanks to David M. Blei and Michael J. Freedman for securing grants that have supported my research. These include funding from ONR N00014-12-1-0764, Air Force Office of Scientific Research FA9550-09-1-0668, BBN 1759 (NSF CNS 0940805), ONR N00014-09-1-0910, NSF IIS 1142251 and NSF CNS 0904729. I was supported by a fellowship provided by Princeton to first-year graduate students, and by the Princeton Computer Science department's teaching assignments during my second year. I also thank the NIPS foundation and USENIX for travel grants.

I'm grateful to Massimiliano Poletto and Eddie Kohler for their mentorship over the years. By showing what computer scientists can do, they inspired me to become one.

My wife Claudine and I moved into Princeton graduate housing soon after our marriage. At the time, graduate studies seemed like an enormous challenge. But Claudine soon made Princeton into our new home, a place of happiness. With her constant support and zest for life, she was there at every moment. For that, I'm indebted to her. My sons Milan and Kiran were both born during my Phd years, and provided joyous distractions. Finally I thank my parents, Shanthi and Krishnaswamy Gopalan; they made several sacrifices that helped me get here.

To Claudine, Milan and Kiran.

Contents

Previous publications and manuscripts

This thesis is based on expanded versions of the following publications and manuscripts:

- Prem Gopalan, Laurent Charlin, David M. Blei. *Content-based recommendation with Poisson factorization.* Advances in Neural Information Processing Systems 27 (NIPS), 2014.
- Prem Gopalan, Wei Hao, David M. Blei, John D. Storey. *Fitting probabilistic models of genetic variation on millions of humans.* In submission.
- Prem Gopalan, Francisco J.R. Ruiz, Rajesh Ranganath, David M. Blei. *Bayesian nonparametric Poisson factorization for recommendation systems.* Proceedings of the 17th International Conference on Artificial Intelligence and Statistics (AISTATS), 2014. (Contributed talk at the NIPS Workshop on Probabilistic Models for Big Data, 2013.)
- Prem Gopalan, Jake Hofman, David M. Blei. *Scalable recommendation with Poisson factorization.* arXiv preprint. http://arxiv.org/abs/1311.1704, 2013.
- Prem Gopalan, Chong Wang, David M. Blei. *Modeling overlapping communities with node popularities.* Advances in Neural Information Processing Systems 26 (NIPS), 2013.
- Prem Gopalan, David M. Blei. *Efficient discovery of overlapping communities in massive networks.* Proceedings of the National Academy of Sciences, 110 (36) 14534-14539, 2013.
- Prem Gopalan, David M. Blei. *Sampling strategies for the scalable inference of communities.* In the NIPS Workshop on Social Network and Social Media Analysis: Methods, Models and Applications, 2012.
- Prem Gopalan, David Mimno, Sean Gerrish, Michael J. Freedman, and David M. Blei. *Scalable inference of overlapping communities.* Advances in Neural Information Processing Systems 25 (NIPS), 2012.

Chapter 1

Introduction

Computational advances have made collecting massive numerical data practical. Complex and structured activity such as chromosome folding in three dimensions, interaction of proteins with other proteins, social interactions, human genetic variation and product ratings are often encoded as matrices of numerical data. The eager scientist is then able to download these matrices to her computer, often within minutes. From these matrices, the scientist seeks to reveal latent structure such as functionally similar chromosomes, ancestral populations that mixed to generate observed genotypes, overlapping online communities and user preferences for products. In this thesis, we will take a Bayesian approach to such scientific data analysis using the framework of latent variable models.

Latent variable models encode the hidden structure in the data in a directed graphical framework (Blei, 2014). We can compute and explore the hidden structure through the posterior—the conditional distribution of the latent variables given the observations. Statistical conclusions drawn in this manner are *Bayesian* – they are probability statements that are conditional on the observed data. Thus, a Bayesian statistician frames her problem as one of updating her uncertainty about how the data was generated, given the observed data. This is a natural approach. Since scientific data is often measured in the context of significant prior knowledge, it can be used to develop improved models or treated as priors in Bayesian inference.

Bayesian data analysis is also a natural approach for analyzing massive or streaming data. Writing in 2011, Michael Jordan notes its advantage over traditional statistical analysis. "In analysis of Big Data one is rarely concerned with quantities such as population means; rather one is concerned with small sub-populations, and concerned with the tails of the distributions" (Jordan, 2011).

Bayesian hierarchical modeling, which exploits the sharing of statistical strength, is well-suited to address this problem.

Further, the data may be too large to fit in memory, or the data may be streaming. Selecting the appropriate model from a class of models becomes difficult in these settings. The growing field of Bayesian nonparametrics addresses this very problem. The scientist's uncertainty about the data can be updated in a principled manner as more data arrives, and new phenomena can be accommodated.

In this thesis, we develop probabilistic models and scalable approximate posterior inference algorithms for several models of discrete data: user consumption and preference for products, articles and their readership, network interactions and human genetic variation. We posit and learn latent variable models which discover hidden structure in these data sets of discrete numerical observations. The hidden structure, computed through the posterior distribution, is useful in prediction and exploratory analysis.

The algorithms in this thesis lie in the framework of *variational inference*. Variational inference is an approach to approximate posterior inference that has been adapted to a variety of probabilistic models (Jordan *et al.*, 1999). Recent innovations combine variational inference with stochastic optimization (Robbins and Monro, 1951) to improve the scalability of inference algorithms. These *stochastic variational inference (SVI)* algorithms (Hoffman *et al.*, 2013) repeatedly analyze subsamples of the data and form noisy gradients of the variational objective while updating an estimate of the hidden structure. Prior research work has typically used uniform subsamples of the data.

Despite these advances, for many of the probabilistic models in this thesis a straightforward application of these approximate posterior inference methods fails to scale them to large data sets. There are several challenges that arise. First, the statistical algorithms that subsample data for efficiency, need to exploit the structure of dependencies between the random variables in the model. Two examples we will encounter in this thesis are the mixed-membership stochastic blockmodel (Airoldi *et al.*, 2008) of network communities (Chapter 6) and the PSD model of human genetic variation (Pritchard *et al.*, 2000) (Chapter 8). These models have become important tools for exploring hypotheses in their domains, but each model faces serious scalability challenges. We will develop novel data subsampling methods for these models.

Further challenges arise in models that are not easily amenable to variational inference. We develop scalable algorithms for such *nonconjugate* models in Chapter 5, Chapter 6, and Chapter 8 using model-specific approximations to bring them into the realm of classic methods. Finally, we propose new models in Chapter 3, Chapter 5 and Chapter 6 with characteristics that enable fast, efficient inference algorithms and good predictive performance. In particular, the models of Chapter 3 and Chapter 5 exploit the data sparsity and the presence of long-tailed distributions, such as user activity in preference data sets.

We study a variety of data sets. The network and user behavior data sets are sparse; they often have less than 0.01% of entries as non-zero. In contrast, the human genetic variation data sets in Chapter 8 are dense: depending on the encoding, less than 50% of the data are zeros. Our algorithms often extend to other, similar types of data. For example, the additive Poisson factorization models of Chapter 3 can capture word frequencies in natural languages, user activity in rating products, or a network of linked web-pages. They are all known to demonstrate characteristics of long-tailed distributions (Sato and Nakagawa, 2010; Paquet and Koenigstein, 2013; Clauset *et al.*, 2009).

In summary, we develop scalable inference for a range of sophisticated models, Bayesian nonparametric models (Section 5.1), generalized linear models (Section 6.1.2), Bayesian hierarchical models (Section 3.3.2). On massive data sets, we demonstrate that Bayesian analysis yields high predictive performance and provides an exploratory tool for the hidden structure in the data.

The document is organized as follows. In Chapter 2, we review directed graphical models, exponential family, mixed-membership models, conditional conjugacy, approximate posterior inference, variational inference and stochastic variational inference. We discuss model selection and model checking.

Learning from user behavior and text. In Chapter 3 and Chapter 4, we present additive Poisson factorization (APF) models for recommending items to users based on sparse consumption or preference matrices. We demonstrate that by capturing the long-tailed distribution of user activity APF models make accurate recommendations on massive data sets. We use them as building blocks for article recommendation, and present the CTPF model that shares latent spaces across readership and article text.

In Chapter 5, we develop a Bayesian nonparametric model that draws user weights from the *Gamma process* to accommodate new data. This model adapts the latent dimensionality of user preferences and item attributes to the data. Inference under this *infinite* model is as fast as inference under the finite Poisson factorization model.

Learning from network interactions. In Chapter 6, we present network models that capture basic ways in which nodes form links—nodes connecting to similar nodes, and nodes connecting to popular nodes. We present the assortative MMSB model and the AMP; the AMP extends the assortative MMSB with node popularities. We develop scalable SVI algorithms for both models while overcoming the nonconjugacy of the AMP. In Chapter 7, we study our network algorithms on large synthetic and real networks, and show that capturing network degree distributions is important for link prediction.

Learning from genetic variation. Population genetics is concerned with understanding the variation of genetic polymorphism in populations of organisms. With a million individuals having been densely genotyped to date, the scale of data on which one can fit population genetic models is orders of magnitude beyond current capabilities. In Chapter 8, we develop a SVI algorithm for the PSD model (Pritchard *et al.*, 2000) of population structure. The algorithm can simultaneously analyze one million individuals, who have been densely genotyped at one million loci, on a single computer. We study our algorithms on both real and synthetic data sets, and demonstrate that we recover the ancestral populations with high accuracy.

Chapter 2

Background

Statistical inference methods play a key role in the development of practical learning algorithms. In this background chapter, we review latent variable models and approximate posterior inference methods upon which our later contributions are based. Our contributions lie in the posited models and in the development of scalable algorithms for learning from massive data sets.

We begin in Section 2.1 by describing latent variable models. Section 2.2 addresses the core computational problem in learning with these models, that of approximate posterior inference. Our review emphasizes the role of conjugacy in tractable inference in Section 2.2.3 and subsampling for computational efficiency in Section 2.2.4. We end with a discussion on model checking and model selection in Section 2.3.

2.1 Latent variable models

The statistical applications in this thesis often involve one or more unobserved quantities for each entity in a massive real-world data set, resulting in a large number of variables. These variables are related to each other, constrained by the assumed structure of the problem. As an example, consider the problem from Chapter 7 of discovering overlapping communities in real-world networks. We can model each user in a social network with a parameter governing their distribution over memberships in multiple communities. Similarly, the presence or absence of a link between two users may be governed by the community memberships of users involved. For example, two graduate students who both attend Princeton University and have children attending the same schools, may be more likely to be linked to each other on Facebook. Such assumptions about the user memberships in hidden communities and the observed connections between them can be captured by a probabilistic model

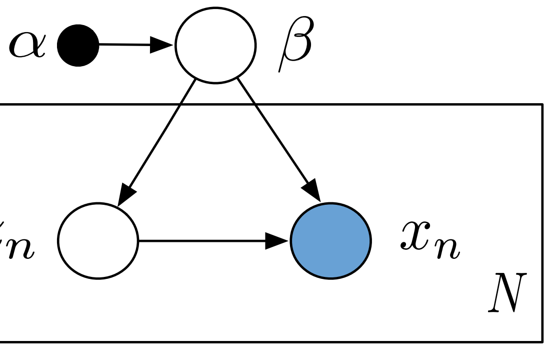


Figure 2.1: A general graphical model with observations $x_{1:N}$, local hidden variables $z_{1:N}$, global hidden variables β and hyperparameters α. Nodes are random variables, arrows denote dependency and plates denote replication. The unshaded nodes β and z_n are hidden, while the shaded x_n are observed. Using the N data points observed, learning the latent structure involves computing the posterior distribution over hidden variables $p(\beta, z|x)$.

of the social network. In particular, a *latent variable model* posits a joint probability distribution of the hidden and observed variables.

In this thesis, we will develop and study latent variables models of networks, human genetic variation and user behavior. In each application, we will posit probabilistic models that are roughly similar to the one shown in Figure 2.1. The figure shows a *graphical model* representation of a latent variable model with N observations $x = x_{1:N}$, N hidden variables $z = z_{1:N}$, a vector of hidden variables β and a vector of fixed *hyperparameters* α. A graphical model is a specification for a family of distributions that conform to the independencies in the graph. We study models whose network structure is a directed acyclic graph.

In Figure 2.1, the β are global variables and the $z_{1:N}$ are local variables. This distinction arises from conditional dependencies in the model. Given β, the nth observation x_n and the nth local hidden variables z_n are conditionally independent of all other observations and other local hidden variables.

The graphical model in Figure 2.1 is a joint distribution of the observed data and the hidden variables that formally describes their interactions. The joint probability distribution is

$$p(x_{1:N}, z_{1:N}, \beta|\alpha) = p(\beta|\alpha) \prod_{i=1}^{N} p(z_i|\beta) p(x_i|\beta, z_i). \tag{2.1}$$

The joint distribution in Equation 2.1 factorizes into a global term and a product of local terms. The global variables β have a prior $p(\beta|\alpha)$, and the local variables z_n encode the hidden structure governing the nth observation.

For descriptive tasks, we seek the conditional distribution of the hidden variables given the observed data $x_{1:N}$ and the fixed hyperparameter α,

$$p(\beta, z_{1:N}|x_{1:N}, \alpha) = \frac{p(x_{1:N}, z_{1:N}, \beta|\alpha)}{p(x_{1:N}|\alpha)}. \tag{2.2}$$

The posterior distribution $p(\beta, z_{1:N}|x_{1:N}, \alpha)$ provides a way to explore the data and interpret the hidden structure. For example, in Chapter 7 we use an approximation to the posterior to discover the overlapping communities in large real-world networks and identify nodes that bridge these communities; in Chapter 8, we use it to study the ancestral populations underlying an individual's genotype variations.

For predictive tasks we seek the predictive distribution which marginalizes out the hidden structure via the posterior distribution,

$$p(x_{N+1}|x_{1:N}, \alpha) = \int p(x_{N+1}|z_{N+1}, \beta)\ p(z_{N+1}|\beta)\ p(\beta|x_{1:N}, \alpha)\ dz_{N+1} d\beta. \tag{2.3}$$

The distribution over future data x_{N+1} in Equation 2.3 can be used to make predictions. For example, we use the predictive distribution in recommendation tasks in Chapter 4 and in link prediction tasks in Chapter 7. In Equation 2.3, the $p(\beta|x_{1:N}, \alpha)$ is the marginalized posterior distribution where the $z_{1:N}$ are integrated out.

In this thesis we will focus on latent variable models where the hyperparameters are assumed to be fixed. The standard approach to estimating the hyperparameters from data is to use *empirical Bayes* (Efron, 2013). This is an important area of future work for the methods presented here.

2.1.1 Latent Dirichlet allocation

We now give an example using a latent variable model of document collections, the Latent Dirichlet allocation (Blei *et al.*, 2003). Latent Dirichlet allocation (LDA) models an observed collection of documents $w = w_{1:D}$ where each document d of length N is a *bag of words* $w_{d,1:N}$. LDA assumes that the documents are generated by a set of K latent topics $\beta_1, \cdots, \beta_k$, where each β_i parameterizes a distribution over a vocabulary of size V. In particular, each document is a mixture θ_d of K topics. The generative process for the LDA is

1. Draw topics $\beta_k \sim \text{Dirichlet}(\eta)$ for $k \in \{1 \cdots K\}$.

2. For each document $d \in \{1 \cdots D\}$:

(a) Draw topic proportions $\theta_d \sim \text{Dirichlet}(\alpha)$.

(b) For each word $w \in \{1 \cdots N\}$:

i. Draw topic indicator $z_{d,n} \sim \text{Mult}(\theta_d)$.

ii. Draw word $w_{d,n} \sim \text{Mult}(\beta_{z_{d,n}})$.

The joint distribution that corresponds to the above generative process and the LDA graphical model in Figure 2.2 is

$$p(\beta, \theta, w|\alpha, \eta) = \prod_{k=1}^{K} p(\beta_k|\eta_k) \prod_{d=1}^{D} p(\theta_d|\alpha) \prod_{d=1}^{D} \prod_{n=1}^{N} p(z_{d,n}|\theta_d) p(w_{d,n}|z_{d,n}, \beta_{z_{d,n}}). \tag{2.4}$$

We note that the global variables are the topics $\beta_{1:K}$, while the local variables are the per-document topic proportions θ_d and the per-word topic indicators $z = z_{d,n}$. We analyze documents by computing the posterior distribution $p(\beta, \theta, z|w)$. Given the documents, the posterior distribution reveals the latent topics $\beta_{1:K}$ that describe the documents, in what proportions each document exhibits those topics $\theta_{1:D}$, and which topic best explains each word $z_{d,n}$. We can use the posterior to study a massive corpus.

A note on global variables

The LDA model of Section 2.1.1 is more computationally tractable than some of the models we will study in this thesis. For example, in the mixed-membership stochastic blockmodel (MMSB) (Airoldi *et al.*, 2008) of networks in Chapter 7, the Markov blanket of each node's community membership is much larger than that of a document, and includes the memberships of all other nodes. The Markov blanket for a node z in a directed graphical model is the set of nodes consisting of z's parents, its children, and the other parents of its children. In other words, while the per-entity mixed-membership variables θ_d are local in the LDA, they are global in the MMSB. These model-wide dependencies introduce new challenges in scalable approximate posterior inference.

2.2 Approximate posterior inference

We have described the motivation behind latent variable models, their representations and provided an example through Latent Dirichlet allocation. We now address the core computing problem in learning with latent variable models. As for most interesting Bayesian models the marginal probability of the data, and consequently the posterior distribution, are intractable to compute

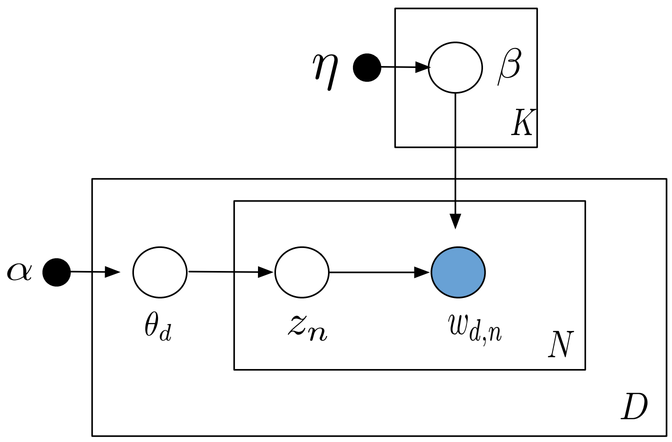


Figure 2.2: The graphical model for Latent Dirichlet allocation (Blei *et al.*, 2003).

(see Equation 2.2). For example, to compute the posterior in the LDA model of Section 2.1.1 we must marginalize over all possible assignments of words to topics, and there are exponential such combinations. For this reason, the posterior distribution is often approximated.

The foremost methods for approximate posterior inference include Markov Chain Monte Carlo (MCMC) (Robert and Casella, 2004) and variational inference (Wainwright and Jordan, 2008). MCMC sampling methods such as Metropolis-Hastings (Metropolis *et al.*, 1953; Hastings, 1970) and Gibbs sampling (Geman and Geman, 1984) form a Markov chain over the latent variables. The stationary distribution of this chain is the posterior distribution we seek. After a burn-in phase, samples from the simulated chain are used to compose an empirical distribution that approximates the posterior. MCMC can be slow on massive data sets due to its dependence on sampling. MCMC is still a powerful tool for Bayesian inference (Neal, 1993; Robert and Casella, 2004).

The algorithms in this thesis lie in the framework of variational inference. Variational inference provides a deterministic alternative to approximate posterior inference in complex probabilistic models (Wainwright and Jordan, 2008; Jordan *et al.*, 1999). It has been adapted to a variety of probabilistic models, though its roots are in the statistical physics literature (Feynman, 1972).

Variational inference algorithms approximate the posterior by defining a parameterized family of distributions over the hidden variables and then fitting the parameters to find a distribution that is close to the posterior. Thus the problem of posterior inference becomes an optimization problem. Using stochastic optimization (Robbins and Monro, 1951), variational inference scales to massive data sets (Hoffman *et al.*, 2013).

In this section, we will focus on two strategies for approximate posterior inference: mean field variational inference and stochastic variational inference. Our algorithms use one of these methods under latent variable models that can broadly be divided into *conditionally conjugate models* and *nonconjugate models*. Conditional conjugacy is an important property of latent variable models that makes mean-field variational inference algorithms simple to derive.

2.2.1 Conditionally conjugate models

We first describe conditionally conjugate models, a class of models for which mean-field variational inference is simple to derive, and has closed-form updates. Conditionally conjugate models make assumptions about the complete conditionals in a model. A *complete conditional* is the conditional distribution of a latent variable given all other latent variables and observations (Ghahramani and Beal, 2001). We assume that each complete conditional is in the exponential family, a family that includes common distributions such as the Gaussian, Poisson, multinomial and the Dirichlet. Our presentation and notation in this section follows Hoffman *et al.* (2013).

The distribution of a variable β is in the exponential family if its density has the form,

$$p(\beta|\eta) = h(\beta)\exp\{\eta^T t(\beta) - a(\eta)\}. \tag{2.5}$$

In Equation 2.5, the vector function $t(\beta)$ is the sufficient statistic, the scalar function $h(\beta)$ is the base measure, and η is the natural parameter vector and $a(\eta)$ is the scalar log normalizer. If the complete conditional of each latent variable is in the same exponential family as its prior distribution, then the model is *conditionally conjugate*.

We now write down the complete conditionals for our general model of Figure 2.1. The complete conditional for the global variable is

$$p(\beta|x, z) = h(\beta)\exp\{\eta_g(x, z)^T t(\beta) - a(\eta_g(x, z))\}. \tag{2.6}$$

The complete conditional for the local variable is

$$p(z_{n,j}|x_n, z_{n,-j}, \beta) = h(z_{n,j})\exp\{\eta_{l,j}(x_n, z_{n,-j}, \beta)^T t(z_n) - a(\eta_{l,j}(x_n, \beta))\} \tag{2.7}$$

The natural parameters in Equation 2.6 and Equation 2.7 are functions of the variables that are conditioned on. The subscripts on the natural parameter η indicate conditionals for the local or global variables. Following the dependencies induced by the graphical model in Figure 2.1, the local conditional for $z_{n,j}$ in Equation 2.7 conditions only on the observation x_n, other local variables that govern x_n, the $z_{n,-j}$ and the global variables β; it is independent of all other observations and local variables. Following Blei (2014), we use the same notation for the base measure, sufficient statistic and log normalizer even though, for example, one set of complete conditionals may be Gaussian, and the other Dirichlet. Finally, there may be multiple vectors of global variables.

The conditionally conjugate models that we study include matrix factorization models of user behavior in Chapter 3 and Chapter 4, and mixed-membership models of networks in Chapter 6 and Chapter 7. We will also develop variational inference algorithms for nonconjugate user behavior, networks and genotype variation models in Chapter 5, Chapter 6 and Chapter 8.

2.2.2 Mean-field variational inference

In mean-field variational inference, we specify a variational family over the latent variables where we independently consider each hidden variable with a different parameterized distribution. The mean-field variational family for the model of Figure 2.1 can be written as

$$q(z, \beta) = q(\beta|\lambda) \prod_{n=1}^{N} q(z_n|\phi_n). \tag{2.8}$$

The global parameters λ govern the global variables; the local parameters ϕ_n govern the variables local to the nth observation. We then maximize the *evidence lower bound* (ELBO), a lower bound on the logarithm of the marginal probability of the observations, $\log p(x)$. We then optimize the variational parameters to find the member of the family that is closest to the posterior. Closeness is measured with Kullback-Leibler divergence (Kullback and Leibler, 1951). The ELBO, denoted as $\mathcal{L}(q)$, is equal to the negative KL divergence up to an additive constant.

$$\begin{aligned} \mathrm{KL}(q(z,\beta)||p(z,\beta|x)) &= \mathbb{E}_q[\log q(z,\beta)] - \mathbb{E}_q[p(z,\beta|x)] \\ &= \mathbb{E}_q[\log q(z,\beta)] - \mathbb{E}_q[p(x,z,\beta)] + \log p(x) \\ &= -\mathcal{L}(q) + \text{const.} \end{aligned} \tag{2.9}$$

The optimized *variational distribution* $q^*(z, \beta)$, which maximizes the ELBO while minimizing the KL-divergence, is a proxy for the true posterior.

To fully specify the variational family in Equation 2.8, we set each variational factor to be in the same family as the complete conditional in the model. For example, if $p(\beta|x, z)$ is a Dirichlet, then λ are Dirichlet parameters. Specifying the variational family in this manner for a conditionally conjugate model results in the simple coordinate ascent variational inference algorithm.

A note on specifying the variational family

In Chapter 5, we present a variant to the classic method, where we posit a mean-field variational family conditional on observed network data sets. This allows for flexible parameterization, accommodating different considerations for links and non-links. In particular, we specify a variational parameter governing each node's hidden interaction variable underlying a link, resulting in two parameters per link. For the non-links, we specify a variational parameter governing the joint distribution of the pair of hidden interaction variables underlying the non-link between two nodes.

2.2.3 Coordinate ascent variational inference

Given a conditionally conjugate model and the mean field variational family, coordinate ascent inference guarantees a local optimum through iterative closed-form updates. In each iteration, a variational parameter is updated while holding all other parameters fixed. Further, the closed-form update is the expected natural parameter of the complete conditional. The global variational parameter of Equation 2.8 is updated using

$$\lambda = \mathbb{E}_{\phi}[\eta_g(z, x)]. \tag{2.10}$$

Due to the mean field assumptions, the $\mathbb{E}_{\phi}[\eta_g(z, x)]$ is an expectation over the local variational parameters ϕ and does not depend on λ. Similarly, the local parameter update is,

$$\phi_{n,j} = \mathbb{E}_{\lambda,\phi_{n,-j}}[\eta_l(x_n, z_{n,-j}, \beta)]. \tag{2.11}$$

In summary, the parameter to each complete conditional is a function of the other latent variables and the mean-field family sets all the variables to be independent. This ensures that the parameter we are optimizing will not appear in the expected parameter.

It can be shown that the global natural parameter vector has the following form,

$$\eta_g(x, z, \alpha) = (\alpha_1 + \sum_{n=1}^{N} t(z_n, x_n), \alpha_2 + N), \tag{2.12}$$

where $t(.)$ are the sufficient statistics, and (α_1, α_2) are the components of the hyperparameter α. The first component α_1 is a vector of the same dimension as β; the second component α_2 is a scalar. We refer the reader to Hoffman *et al.* (2013) for the details.

With a similar closed-form local update in hand, the coordinate ascent algorithm is simply to iterate between updating each local parameter and the global parameters. This algorithm finds a stationary point of the ELBO.

2.2.4 Stochastic variational inference

The coordinate ascent algorithm of Section 2.2.3 often fails to scale to the massive data set sizes we study in this thesis. This is because coordinate ascent is a *batch algorithm*; the variational parameters corresponding to all latent variables are updated in each iteration. The algorithm must analyze every observation (using the initialized values of variational parameters) before any progress in inference can be made.

As we discuss in Chapter 6 on scalable inference of network data, the simple coordinate ascent algorithm requires time quadratic in the number of nodes in the network. This makes inference computationally intractable on even moderately-sized networks. With N nodes in a network, the computational problem is that there are $O(N^2)$ terms in the ELBO and $O(N^2)$ variational parameters. Coordinate ascent inference must consider each pair of nodes at each iteration (Airoldi *et al.*, 2008), but even a single pass through a large network can be prohibitive. (Note that the sparsity of the network does not help; for the models we study in Chapter 6, we need to optimize parameters for all pairs of nodes regardless of whether they are connected in the observed network.)

Stochastic variational inference (SVI) (Hoffman *et al.*, 2013) overcomes this problem, and it allows handling of massive and streaming data sets. Using SVI, in Chapter 6, we analyze networks with millions of nodes; and in Chapter 8 we analyze a "tera-scale" data set with 10^{12} observations corresponding to genotype variations.

Stochastic variational inference uses stochastic optimization to fit the global variational parameters, and repeatedly subsamples the data and fits the local parameters only to that subsample, in each iteration. Stochastic optimization algorithms follow noisy estimates of the gradient of an objective with a decreasing step-size. In their classic paper, Robbins and Monro showed that with certain step-size schedules, such algorithms provably lead to the optimum of a convex function (Robbins and Monro, 1951). In our case, they provably lead to a local optimum. Since the fifties, stochastic optimization has blossomed into a field of its own (Spall, 2003; Kushner and Yin, 1997). It plays an important role in scaling machine learning algorithms up to very large data sets (Bottou and LeCun, 2004).

Stochastic optimization is particularly efficient when the objective (and gradient) are a sum of terms, as is the case for the variational objective of the models we study in this thesis. In these settings, we cheaply compute a stochastic gradient by first subsampling a subset of terms and then forming an appropriately scaled gradient. The scaled gradient is a random variable whose expectation is the true gradient.

We can derive a general form of this scaled gradient in the case when a single observation x_n is subsampled uniformly at random, and the scaling factor is simply the number of data points N. Using the complete conditional from Equation 2.10, we can compute the conditional natural parameter for the global parameter β given N replicates of x_n,

$$\eta_g(x, z, \alpha) = (\alpha_1 + N(t(z_n, x_n)), \alpha_2 + N), \tag{2.13}$$

where $t(.)$ are the sufficient statistics, and (α_1, α_2) are the components of the hyperparameter α (see Equation 2.12). We again refer the reader to Hoffman *et al.* (2013) for the details.

An important assumption underlying Equation 2.13 is that the N data points can be independently sampled. This assumption is natural for the LDA model of a collection of documents, but does not extend to network models such as those studied in Chapter 6. We develop sampling methods appropriate for models of massive networks, where the Markov blanket of each node includes the variables associated with all other nodes. These methods work by non-uniformly subsampling the more informative observations in the network.

Stochastic variational inference uses *natural gradients* instead of the standard “Euclidean” gradients. The classical gradient method for ELBO maximization tries to find a maximum of the ELBO $\mathcal{L}(\lambda)$ by taking a series of steps of size ρ in the direction of the steepest ascent

$$\lambda^{(t+1)} = \lambda^{(t)} + \rho \bigtriangledown_\lambda \mathcal{L}(\lambda). \tag{2.14}$$

The gradient direction implicitly depends on the Euclidean distance metric associated with the vector λ, a parameter to a probability distribution. A better measure of dissimilarity between probability distributions is the symmetrized KL divergence (Hoffman *et al.*, 2013). The natural gradient points in the direction of steepest ascent in the Riemannian space, where the local distance is defined by KL divergence. We can find the natural gradient by premultiplying the gradient by the Fisher information matrix G (Amari, 2001),

$$G(\lambda) = \mathbb{E}_\lambda[(\bigtriangledown_\lambda \log q(\beta|\lambda))(\bigtriangledown_\lambda \log q(\beta|\lambda))^T]. \tag{2.15}$$

When $q(\beta|\lambda)$ is in the exponential family, the metric is the second derivative of the log normalizer (Hoffman *et al.*, 2013),

$$G(\lambda) = \bigtriangledown^2_\lambda a(\lambda). \tag{2.16}$$

This results in a simple form for the natural gradient because the premultiplying the gradient by the inverse Fisher metric cancels out the covariance matrix of the sufficient statistic of $q(\beta|\lambda)$,

$$\hat{\bigtriangledown}_\lambda \mathcal{L} = \mathbb{E}_\phi[\eta_g(x, z)] - \lambda. \tag{2.17}$$

A similar update can be derived for the local variational parameters.

In the Euclidean gradients, the Fisher information matrix is expensive to compute for variational parameters with many components. The simple form in Equation 2.17 allows the natural gradients to be computed easily.

2.2.5 Nonconjugate variational inference

Our review has so far concerned itself with conditionally conjugate latent variable models. In Section 2.2.2, we described how mean-field variational inference can be used as an off-the-shelf method for conditionally conjugate models. In Section 2.2.4 we reviewed stochastic variational inference (SVI) algorithms that combine mean-field variational inference for conditionally conjugate models and stochastic optimization to scale inference to massive data sets. They gain efficiency by repeatedly subsampling the data.

Real-world applications often require working with models that are not conditionally conjugate. In these models, the coordinate updates are not available in closed form because a subset of the nodes in the graphical model do not satisfy conditional conjugacy. For example, in the AMP model of node popularities and communities that we present in Chapter 6, the priors on the node popularity and the community strengths are not conjugate to the conditional likelihood of the data.

Wang and Blei (2013) present variational inference methods for nonconjugate models including the Laplace variational inference and the Delta method variational inference. Their work unifies existing algorithms derived for specific models such as Bayesian logistic regression (Jaakkola and Jordan, 1996), discrete choice models (Braun and McAuliffe, 2010) and the correlated topic model (Blei and Lafferty, 2007). Laplace VI uses Laplace approximations (MacKay, 1992; Tierney *et al.*, 1989) within the coordinate ascent updates of Equation 2.10 and Equation 2.11; Delta method VI apply Taylor approximations to approximate the variational objective $\mathcal{L}(q)$ of Equation 2.9 and then derive the corresponding updates.

In this thesis, to construct the approximation $\hat{\mathcal{L}}(q)$ to $\mathcal{L}(q)$, we generally use the zeroth-order Delta method for moments (Braun and McAuliffe, 2010), and then derive either coordinate ascent or SVI algorithms.

2.3 Model checking and model selection

Once a model is fit we often cannot immediately apply it to the predictive or explorative task at hand. There are often multiple competing models or a set of models indexed by hyperparameter settings. For example, varying the number of communities K in a network model results in multiple models, each with its own setting of K. Therefore, the posterior distribution approximated in the previous section, for any particular model, underestimates uncertainty: the assumed model is wrong in any real-world setting and there are likely to be other reasonable models. This necessitates the complementary tasks of model checking and model selection.

In model checking we compare the model against the observed data, to assess ways in which the given model falls short and how it can be improved. One approach is to use *posterior predictive checks* (Rubin, 1984; Gelman *et al.*, 1996). In a posterior predictive check (PPC), we define a discrepancy $T(X)$ to be a function of the data. Let x^{REP} be a random data set drawn from the posterior predictive distribution. Then, the PPC check is $p(T(X^{REP}) > T(x)|x)$. This is the probability that the replicated data differ greatly in terms of the function T, from the observations.

In model selection, we select a model from multiple candidate models using a measure of its performance on its assigned task. For example, the network models require setting the number of communities K; in typical applications we will want to set this number based on the data. In our empirical study in Chapter 6, we evaluate the predictive performance of the model for varying numbers of communities. We held out a portion of the network $\mathbf{y}_{\text{held-out}}$ and calculated $p(\mathbf{y}_{\text{held-out}} \,|\, \mathbf{y}_{\text{observed}})$; a better model will assign higher probability to the held out set. This reflects a *predictive approach* to model selection, and has good statistical properties (Geisser and Eddy, 1979).

As a concrete example consider the LDA model of Section 2.1.1, where a portion of a document is held-out. Let the held-out portion be $w_{d,\text{test}}$ and the observed portion be $w_{d,\text{observed}}$. The probability of the held-out portion is given by integrating over the posterior Dirichlet θ_d,

$$p(w_{d,\text{held-out}}|w_{d,\text{observed}}) = \int \sum_z p(w_{d,\text{observed}}|z)p(z|\theta_d)p(\theta_d|w_{d,\text{observed}})d\theta_d \tag{2.18}$$

where the posterior distributions are approximated using variational inference.

We also use the predictive approach to assess convergence of the inference algorithm. A simple approach to assess convergence in the coordinate ascent algorithm of Section 2.2.3 is to compute the training log-likelihood $\log p(y_{\text{observed}})$. A serious limitation of this metric is it does not measure whether a model is overfit, and is more appropriate when an MAP or MLE estimate is fit. Instead the predictive approach determines convergence using the metric from model selection, $p(\mathbf{y}_{\text{held-out}} \,|\, \mathbf{y}_{\text{observed}})$. It is important to note that the held-out set used for these experiments, called the *validation* set, is different from the validation set used in model selection.

2.4 Discussion

In this chapter, we described latent variable models and their representation through directed graphical models. For the class of conditionally conjugate models, we summarized mean-field variational methods of approximate posterior inference. We outlined the ideas underlying SVI, an approach to efficient inference on massive data sets that leverages subsampling and natural gradients.

The applications in this thesis introduce new challenges, making a straightforward development of the variational inference algorithms difficult. A key assumption underlying SVI's immediate application is the uniform subsampling of data. In Chapter 6, we develop sampling methods appropriate for models of massive networks, where the Markov blanket of each node includes the variables associated with all other nodes. These methods work by non-uniformly subsampling more informative observations in the network. While our contributions lie in the scalable algorithms developed for the particular models, the subsampling methods we develop in Chapter 6 are more widely applicable. Similarly, in Chapter 8, we develop a particular subsampling strategy for high accuracy on both simulated and real-world data sets of genetic variation.

Another challenge is that many practical models lack the convenience of conditional conjugacy. We develop both coordinate ascent and SVI algorithms for a variety of nonconjugate models: a nonconjugate mixed-membership models of networks in Chapter 6, a Bayesian nonparametric nonconjugate model of user behavior in Chapter 5 and a nonconjugate model of genotype variation in Chapter 8. However, we leverage conditional conjugacy whenever it is satisfied. We develop coordinate ascent algorithms for conditionally conjugate user behavior models in Chapter 3 and stochastic variational inference algorithms for conditionally conjugate network models in Chapter 6.

Part I

User behavior

Chapter 3

Additive Poisson factorization

In this chapter, we present additive Poisson factorization (APF), a class of latent variable models for modeling massive discrete data that commonly arise in the context of recommendation tasks. We model user behavior such as users watching movies, scientists reading articles and readers clicking on newspaper articles. Our applications are prediction tasks on massive data sets. Our goal is to recommend items to users based on consumption or ratings provided by millions of users. In the case of articles, we also use their context in suggesting articles to readers. We develop efficient algorithms for learning under these models.

Poisson factorization models are widely studied in the statistical computing (Dunson and Herring, 2005; Cemgil, 2009) and machine learning literature (Lee and Seung, 1999; Canny, 2004). Our contributions include variations on these models aimed at scalable and accurate *top-n* recommendations. The goal of the recommender system is to find a few items which are most relevant to the user; the top-n performance can be measured by metrics such as mean rank (Hu *et al.*, 2008) and precision/recall. Further, we present Bayesian models that extend PF using the additive property of the Poisson distribution.

We review recommendation using matrix factorization in Section 3.2, before presenting three finite probabilistic models from the class of additive Poisson factorization in Section 3.3. We first study a simple instance of this class: the Bayesian Poisson factorization model (BPF) of recommendation. We extend BPF to a hierarchical model in Section 3.3.2 that captures the heterogenous interests of users and the wide range of popularity of items. We extend BPF to article recommendation in Section 3.3.3 by capturing both readership and article text. We discuss their statistical properties in Section 3.4, and review related work in Section 3.5.

Practical applications of the APF models require computationally efficient inference algorithms. We derive algorithms that only iterate over the non-zero observations in the data set in Section 3.6.

Finally, in Section 3.7, we present a unified approach to modeling discrete data for recommendation tasks, emphasizing their shared statistical properties, and wider applicability to other sources of data such as those arising from network interactions.

We evaluate the PF models on prediction tasks in Chapter 4, studying their model fitness to user behavior and text in the process. In Chapter 5, we use Bayesian nonparametric assumptions to infer the latent dimensionality of user preferences and item attributes. In general, with the ideas presented here, we can develop sophisticated statistical models of discrete data.

3.1 Introduction

In an additive Poisson factorization model each observation is drawn from a Poisson distribution whose rate parameter is an inner product of vectors of non-negative latent variables. The basic model we present is Bayesian Poisson factorization (BPF), a form of probabilistic matrix factorization (Salakhutdinov and Mnih, 2008a) that replaces the standard Gaussian likelihood and real-valued representations with a Poisson likelihood and non-negative representations. It associates each user with a latent vector of preferences, each item with a latent vector of attributes, and constrains both sets of vectors to be sparse and non-negative. In Section 3.3.2, we extend the BPF to a hierarchical model that explicitly captures the diversity of users, some tending to consume much more than others, and the diversity of items, some being much more popular than others.

In Section 3.3.3, we consider the setting in which we recommend both old and new scientific articles to readers. A natural approach to recommending new articles is to extend the PF model to explain the text of articles in addition to the user behavior, i.e., users listing articles in their library. We call this model the collaborative topic Poisson factorization model (CTPF). In addition to effectively solving the cold-start recommendation problem, the CTPF provides a new exploratory window into the structure of the document collection. It organizes the articles according to their topics and identifies important articles both in terms of those important to their topic and those that have transcended disciplinary boundaries.

Using the additive properties of independent Poisson random variables, the PF models capture dependencies between discrete data, for example, the dependence of user ratings of an article on its content. We demonstrate how the sharing of latent spaces can capture such dependencies, providing modeling flexibility and good predictive performance.

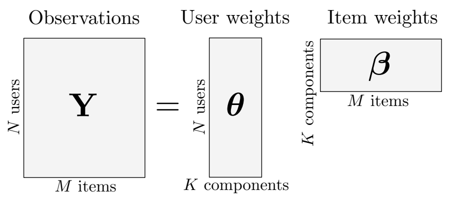


Figure 3.1: Matrix factorization represents users and items with low dimensional vectors.

The models we present are tailored to the real-world properties of discrete user behavior data. They capture the heterogeneous interests of users, the finite resources that users have to consume items, the data sparsity and the long-tailed distributions of user consumption. They provide sparse latent representations and enable efficient inference algorithms. We discuss these statistical properties in Section 3.4.

We will show that the BPF, and its hierarchical variant, the HPF, provide good predictive performance on large data sets of Netflix users watching movies, Last.FM users listening to music, scientists reading papers, and *New York Times* readers clicking on articles. Further, we will show that CTPF scales easily and provides better recommendations than the leading method—the collaborative topic regression (Wang and Blei, 2011)—and alternatives.

3.2 Recommendation using matrix factorization

Recommendation systems are a vital component of the modern Web. They help readers effectively navigate otherwise unwieldy archives of information and help websites direct users to items—movies, articles, songs, products—that they will like. A basic "in-matrix" recommendation system is built from user behavior data, historical data about which items each user has consumed, be it clicked, viewed, rated, or purchased. We define in-matrix items as those that have been rated by at least one user in the recommendation system. We refer to the outcome from a user-item interaction as a "rating". The observation y_{ui} is the rating that user u gave to item i, or zero if no rating was given. In so-called "implicit" consumer data, y_{ui} equals one if user u consumed item i and zero otherwise. User behavior data, such as purchases, ratings, clicks, or views, are typically sparse. Most of the values of the matrix y are zero.

Given the user-item matrix of ratings, we uncover the behavioral patterns that characterize various types of users and the kinds of items they tend to like. Then, we exploit these discovered patterns to recommend future items to its users.

Currently, the workhorse method for recommendation systems is matrix factorization (MF) (Ko-

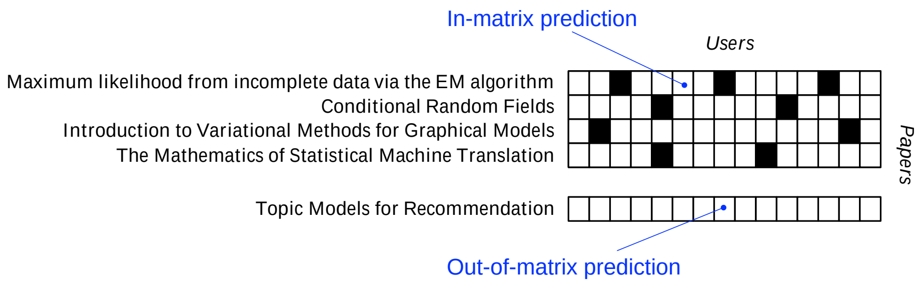

In-matrix prediction
Users
Maximum likelihood from incomplete data via the EM algorithm
Conditional Random Fields
Introduction to Variational Methods for Graphical Models
The Mathematics of Statistical Machine Translation
Papers
Topic Models for Recommendation
Out-of-matrix prediction

Figure 3.2: An illustration of in-matrix and out-of-matrix (cold-start) recommendations of scientific articles.

ren *et al.*, 2009; Hu *et al.*, 2008). MF represents users and items with low dimensional vectors and computes the affinity between a user and item with the inner product of their respective representations. MF is typically fit with squared loss, where the algorithm finds representations that minimize the squared difference between the predicted value and the observed rating. This corresponds to a Gaussian model of the data (Salakhutdinov and Mnih, 2008b). MF has been extended in many ways to implement modern recommendation systems (Dror *et al.*, 2012b,a; Rendle *et al.*, 2009a).

Matrix factorization has also been studied for the "out-of-matrix" or "cold-start" recommendation problem. In this setting, the algorithm suggests previously unrated items to users, along with in-matrix items. Figure 3.2 illustrates the two types of recommendations. Prior work on cold-start recommendation of articles to users has captured both article text and user ratings of these articles (Wang and Blei, 2011).

3.3 Additive Poisson factorization

In an additive Poisson factorization model each observation is drawn from a Poisson distribution, and the rate parameter of each Poisson distribution is an inner product of two vectors of non-negative latent variables. We now consider a simple instance of this class: the Bayesian Poisson factorization model (BPF) of recommendation. We extend the model in Section 3.3.2 and Section 3.3.3, and then discuss their statistical properties in Section 3.4.

3.3.1 The basic model

In this section, we present a basic model of both explicit and implicit user-item ratings—the Bayesian Poisson factorization model (BPF).

We model the rating y_{ui} that a user u gave to item i with factorized Poisson distributions (Canny,

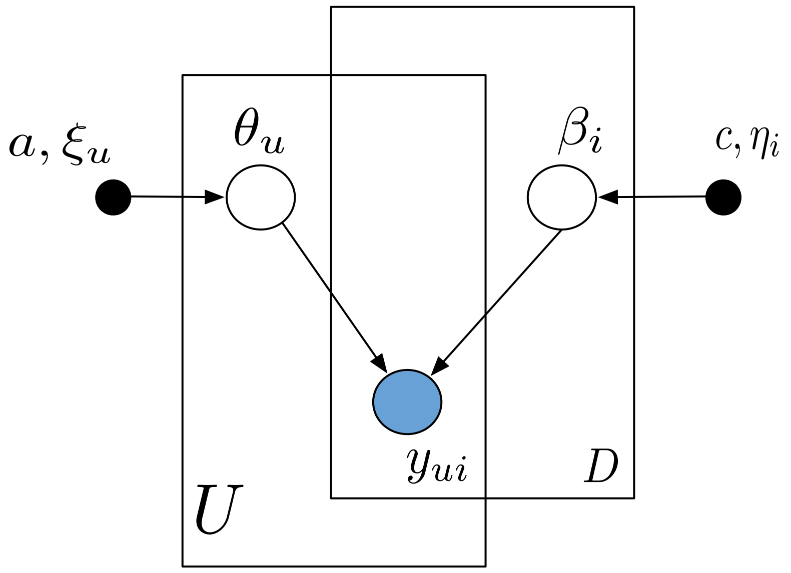


Figure 3.3: The Bayesian Poisson factorization model (BPF) with U users and D items.

2004), where each item i is represented by a vector of K latent attributes β_i and each user u by a vector of K latent preferences θ_u. The observations y_{ui} are modeled with a Poisson, parameterized by the inner product of the user preferences and item attributes. This is a variant of probabilistic matrix factorization (Salakhutdinov and Mnih, 2008c) but where each user and item's weights are positive (Lee and Seung, 1999) and where the Poisson replaces the Gaussian. Beyond the basic data generating distribution, we place Gamma priors on the latent attributes and latent preferences, which encourage the model towards sparse representations of the users and items.

1. For each user u, sample preferences $\theta_{uk} \sim \text{Gamma}(a, \xi_u)$.

2. For each item i, sample attributes $\beta_{ik} \sim \text{Gamma}(c, \eta_i)$.

3. For each user u and item i, sample rating $y_{ui} \sim \text{Poisson}(\theta_u^\top \beta_i)$.

The main computational problem is posterior inference: given an observed matrix of user behavior, we discover the latent attributes that describe the items and the latent preferences of the users.

In the BPF, we fix the hyperparameters η_i to a single constant η for all items, and the hyperparameters ξ_u to a single value ξ for all users. The subscripts are used here to maintain consistency across the models. In the next section, we will place priors on η_i and ξ_u.

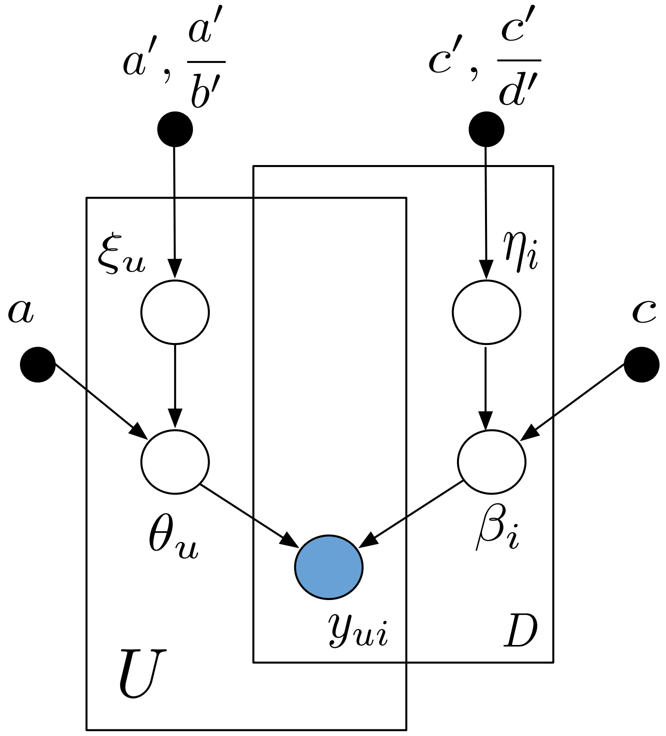


Figure 3.4: The hierarchical Poisson factorization model (HPF).

3.3.2 The hierarchical model

The literature on recommendation systems suggests that a good model of user behavior must capture the heterogenous interests of users and the popularity of items (Koren *et al.*, 2009). This is typically addressed in classical matrix factorization by adding explicit bias terms to the model. In the Bayesian Poisson factorization, there's a natural way to do this.

The hierarchical model extends BPF by placing additional Gamma priors on the user and item-specific rate parameter of those Gammas, which controls the average size of the representation. This hierarchical structure explicitly captures the diverse consumption activity of users and the popularity of items.

Putting this together, the generative process of the hierarchical Poisson factorization model (HPF) is as follows:

1. For each user u:

   (a) Sample activity $\xi_u \sim \text{Gamma}(a', a'/b')$.

   (b) For each component k, sample preference $\theta_{uk} \sim \text{Gamma}(a, \xi_u)$.

2. For each item i:

    (a) Sample popularity $\eta_i \sim \text{Gamma}(c', c'/d')$.

    (b) For each component k, sample attribute $\beta_{ik} \sim \text{Gamma}(c, \eta_i)$.

3. For each user u and item i, sample rating $y_{ui} \sim \text{Poisson}(\theta_u^\top \beta_i)$.

This process describes the statistical assumptions behind the model. The BPF is therefore a subclass of the HPF where we fix the rate parameters for all users and items to the same pair of hyperparameters.

Recommending in-matrix items to users

The central computational problem is posterior inference, which is akin to "reversing" the generative process. In the BPF and the HPF, given a user behavior matrix, we want to estimate the conditional distribution of the latent user preferences and item attributes, $p(\theta_{1:N}\beta_{1:M} \,|\, \boldsymbol{y})$.

The posterior is the key to recommendation. Once the posterior is fit, we estimate the posterior expectation of each user's preferences, each items attributes and, subsequently, form predictions. We recommend items to users by ranking each user's unconsumed items by their posterior expected Poisson parameters,

$$\text{score}_{ui} = \mathrm{E}\left[\theta_u^\top \beta_i \,|\, \boldsymbol{y}\right]. \tag{3.1}$$

This amounts to asking the model to rank by probability which of the presently unconsumed items each user will likely consume in the future.

Figure 3.5 illustrates the HPF on data from Netflix. The Netflix data contains the ratings of 480,000 users on 17,000 movies, organized in a matrix of 8.16 billion cells (and containing 250 million ratings). Each observed rating is an integer ranging from 1 to 5 that the user provided for a movie. From these data, we extract the user preferences and the movies associated with those preferences. These movies help us interpret the components in the model. The left panel illustrates some of those components—the algorithm has uncovered action movies, independent comedies, and 1980s science fiction.

The top panel illustrates how we can use these patterns to form recommendations for an (imaginary) user. This user enjoys various types of movies, including fantasy ("Lord of the Rings"), classic science fiction ("Star Wars: Episode V"), and independent comedies ("Clerks", "High Fidelity"). Of course, she has only seen a handful of the available movies. The HPF first uses the movies she has seen to infer what kinds of movies she is interested in, and then uses these inferred interests

"Action"	**"Indie Comedy, Romance"**	**"80's Science Fiction"**
The Matrix	Grosse Pointe Blank	Star Wars: Episode IV: A New Hope
The Matrix: Reloaded	Four Weddings and a Funeral	Star Wars: Episode VI: Return of the Jedi
Spider-Man	High Fidelity	Star Wars: Episode V: The Empire Strikes Back
X2: X-Men United	Much Ado About Nothing	Back to the Future Part II

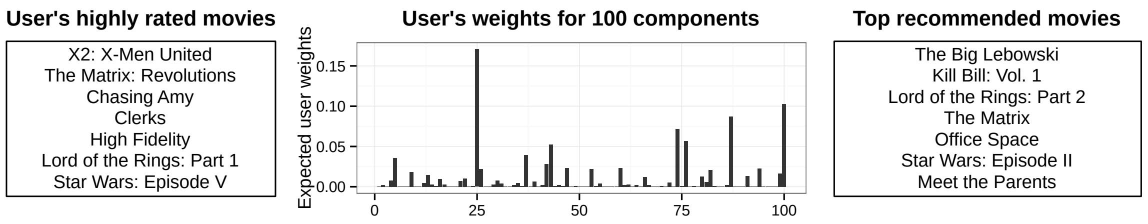


Figure 3.5: The top panel shows the top movies in 3 components for a user from the Netflix data set. The bottom panel is an illustration showing a subset of the highly rated movies by this user, and the right panel shows movies recommended to the user by our algorithm. The expected user's K-vector of weights θ_u, inferred by our algorithm is shown in the middle panel.

to suggest new movies. The list of movies at the bottom of the figure was suggested by our algorithm. It includes other comedies (such as "The Big Lebowski") and other science fiction (such as "Star Wars: Episode II"). The components in Figure 3.5 (left) illustrate the top items for specific attribute dimensions and the plot in Figure 3.5 (middle) illustrates the estimated preference vector for the given user. A spike in the preference vector implies that the user tends to like items with the corresponding latent attribute. We note that this general procedure is common to many variants of matrix factorization discussed in Section 3.2 and Section 3.5.

3.3.3 A combined model of readership and article text

In this section, we develop a model that demonstrates how additive PF models can capture multiple types of discrete data with shared latent spaces.

Consider the setting in which we recommend scientific articles to readers. The BPF and the HPF are basic recommendation models; they cannot handle the cold-start problem or easily give topic-based representations of readers and articles. The collaborative topic PF model we present in this section extends BPF and is a model of the article content (the document-word matrix) and the readers of an article (the reader-document matrix). As we will see, CTPF effectively solves the cold-start problem and organizes readers and articles by their topic structure.

The BPF is a building block in the CTPF, modeling both reader behavior and article content. Rather than modeling them as independent factorization problems, we connect the two latent spaces using a correction term (Wang and Blei, 2011) which we'll describe below.

The CTPF captures a user u's *topic preferences* with a vector θ_u of size K and assumes that a

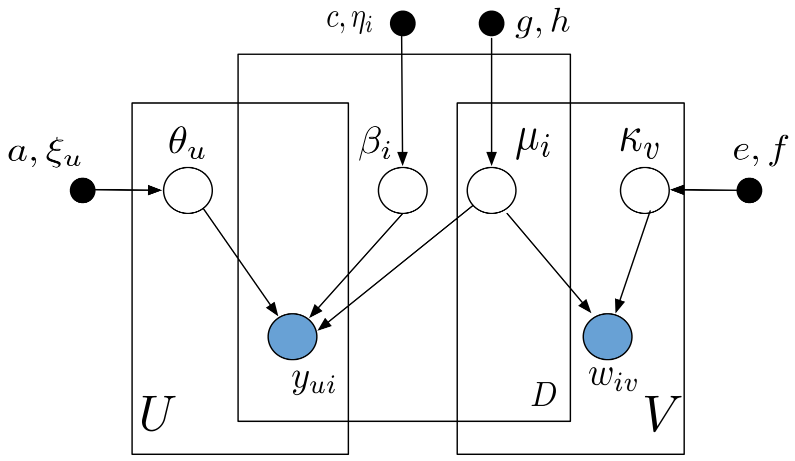


Figure 3.6: The collaborative topic Poisson factorization model (CTPF) with U users, D documents and vocabulary size V.

document i is generated by a topic model with *topic intensities* μ_i and unnormalized *topics* κ.

Suppose we have data containing D documents and U users. CTPF assumes a collection of K unnormalized *topics* $\kappa_{1:K}$. Each topic κ_k is a collection of word intensities on a vocabulary of size V. Each component κ_{vk} of the unnormalized topics is drawn from a Gamma distribution. Given the topics, CTPF assumes that a document i is generated with a vector of K latent *topic intensities* μ_i, and represents users with a vector of K latent topic preferences θ_u. Additionally, the model associates each document with K latent *topic offsets* β_i that capture the documentss deviation from the topic intensities. These deviations occur when the content of a document is insufficient to explain its ratings. For example, these variables can capture that a machine learning article is interesting to a biologist, because other biologists read it.

We now define a hierarchical generative process for the observed word counts in documents and observed user ratings of documents under the CTPF:

1. **Document model:**

   (a) Draw topics $\kappa_{vk} \sim \text{Gamma}(e, f)$

   (b) Draw document topic intensities $\mu_{ik} \sim \text{Gamma}(g, h)$

   (c) Draw word count $w_{iv} \sim \text{Poisson}(\mu_i^T \kappa_v)$.

2. **Recommendation model:**

   (a) Draw user preferences $\theta_{uk} \sim \text{Gamma}(a, \xi_u)$

   (b) Draw document topic offsets $\beta_{ik} \sim \text{Gamma}(c, \eta_i)$

(c) Draw rating $r_{ui} \sim \text{Poisson}(\theta_u^T(\mu_i + \beta_i))$.

CTPF is an instance of additive Poisson factorization. As in the BPF, we fix the hyperparameters η_i to the same value η across items,and ξ_u to the same value ξ across users.

CTPF specifies that the conditional probability that a user u rated document i with rating r_{ui} is drawn from a Poisson distribution with rate parameter $\theta_u(\mu_i + \beta_i)$. The form of the factorization couples the user preference for the document topic intensities μ_i and the document topic offsets β_i. This allows the user preferences to be interpreted as affinity to latent topics. Using this interpretable latent space, we explore and organize real-world articles in Chapter 4.

The above intuitions for capturing readers preferences derive from collaborative topic regression (Wang and Blei, 2011). However, the CTPF has two main advantages over competing methods, both of which contribute to its empirical performance (see Section 4.1). First, the CTPF model augmented with auxiliary variables is conditionally conjugate (see Section 3.6 and Chapter 2). This allows CTPF to conveniently use standard variational inference with closed-form updates (see Section 3.6). Second, since CTPF is built on Poisson factorization, which provides sparse representations of the users and the items, allowing for greater interpretability. Further, it can take advantage of the natural sparsity of user consumption of documents, and efficiently analyze massive real-world data. We discuss these properties in more detail in Section 3.4.

Recommending in-matrix and out-of-matrix articles

We analyze data with D documents (articles) and U users (readers) with the CTPF via the posterior distribution over latent variables $p(\kappa_{1:K}, \mu_{1:D}, \beta_{1:D}, \theta_{1:U}|\boldsymbol{w}, \boldsymbol{r})$. By estimating this latent structure, we can characterize user preferences and the readership of documents in many useful ways.

Once the posterior is fit, we use the CTPF model to recommend in-matrix documents and out-of-matrix or cold-start articles to readers. Out-of-matrix articles are new and therefore have no ratings from readers. Therefore, a cold-start recommendation of a new article is based entirely on its content. For predicting these articles, we rank each reader's unread documents by their posterior expected Poisson parameters,

$$\text{score}_{ui} = \text{E}\left[\theta_u^T(\mu_i + \beta_i)|\boldsymbol{w}, \boldsymbol{r}\right]. \tag{3.2}$$

The intuition behind the CTPF posterior is that when there is no reader data, we depend on the topics to make recommendations. When there is both reader data and article content, this gives information about the topic offsets. Notice that under the CTPF the in-matrix and cold-start

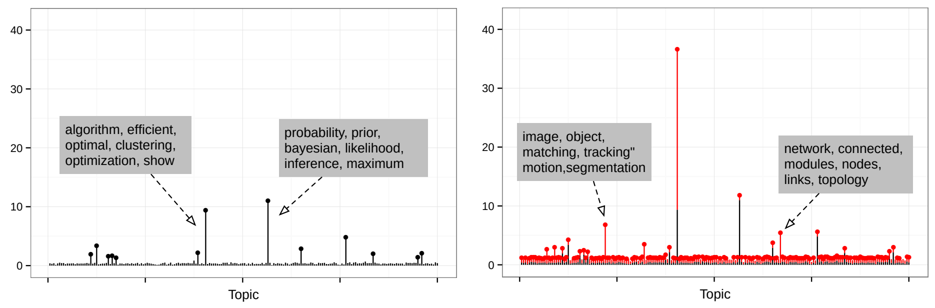


Figure 3.7: We visualized the inferred topic intensities μ (the black bars) and the topic offsets β (the red bars) of an article in the Mendeley (Jack *et al.*, 2010) dataset. The plots are for the statistics article titled "Maximum likelihood from incomplete data via the EM algorithm". The black bars represent the topics that the EM paper is about. These include probabilistic modeling and statistical algorithms. The red bars represent the preferences of the readers who have the EM paper in their libraries. It is popular with readers interested in many fields outside of those the paper discusses, including computer vision and statistical network analysis.

recommendations are not disjoint tasks. There is a continuum between these recommendations tasks. For example, the model can provide better predictions for articles with few ratings by leveraging the article's latent topic intensities μ_i.

Exploring impact of scientific articles

We illustrate the model with an example. Consider the classic paper "Maximum likelihood from incomplete data via the EM algorithm" (Dempster *et al.*, 1977). This paper, published in the Journal of the Royal Statistical Society (B) in 1977, introduced the expectation-maximization (EM) algorithm. The EM algorithm is a general method for finding maximum likelihood estimates in models with hidden random variables. As many readers will know, EM has had an enormous impact on many fields, including computer vision, natural language processing, and machine learning. This original paper has been cited over 37,000 times. (Wang and Blei also use the EM paper as an example in Wang and Blei (2011).)

Figure 1 illustrates the CTPF representation of the EM paper. (This model was fit to the shared libraries of scientists on the Mendeley website; the number of readers is 80,000 and the number of articles is 261,000.) In the figure, the horizontal axes contains topics, latent themes that pervade the collection (Blei *et al.*, 2003). Consider the black bars in the left figure. These represent the topics that the EM paper is about. (These were inferred from the abstract of the paper.) Specifically, it is about probabilistic modeling and statistical algorithms. Now consider the red bars on the right, which are summed with the black bars. These represent the preferences of the readers who have the EM paper in their libraries. CTPF has uncovered the interdisciplinary impact of the EM paper.

It is popular with readers interested in many fields outside of those the paper discusses, including computer vision and statistical network analysis.

The CTPF representation has advantages. For forming recommendations, it naturally interpolates between using the text of the article (the black bars) and the inferred representation from user behavior data (the red bars). On one extreme, it recommends rarely or never read articles based mainly on their text; this is the cold start problem. On the other extreme, it recommends widely-read articles based mainly on their readership. (In this setting, it can make good inferences about the red bars.) Further, in contrast to traditional matrix factorization algorithms, the space of preferences and articles is defined via interpretable topics. CTPF thus offers reasons for making recommendations, readable descriptions of reader preferences, and an interpretable organization of the collection. For example, CTPF can recognize the EM paper is among the most important statistics papers that has had an interdisciplinary impact.

3.4 Statistical properties

With the modeling details in place, we highlight several statistical properties of the additive Poisson factorization (APF) models. We use the HPF in our study, but these properties generalize to other models as they depend on the shared aspects such as the Poisson likelihood, the factorized, sparse representations and the conditional conjugacy. These properties provide advantages over some variants of the Gaussian matrix factorization, and contribute to the models' good empirical performance as demonstrated in Chapter 4.

When we refer to Gaussian matrix factorization in this section, we mean L2 regularized matrix factorization with bias terms for users and items, fit using stochastic gradient descent (Koren *et al.*, 2009). Without the bias terms, this corresponds to maximum a-posteriori inference under Probabilistic Matrix Factorization (Salakhutdinov and Mnih, 2008c). We used a popularity-based sampling scheme to generate negative examples: we sample users by activity—the number of items rated in the training set—and items by popularity—the number of training ratings an item received.

3.4.1 Capturing user consumption

One statistical characteristic of real-world user behavior data is the distribution of user activity (i.e., how many items a user consumed) and item popularity (i.e., how many users consumed an item). As shown in Figure 3.8, these distributions tend to be long-tailed: while most users consume a handful few items, a few "tail users" consume thousands of items. A question we can ask of a statistical

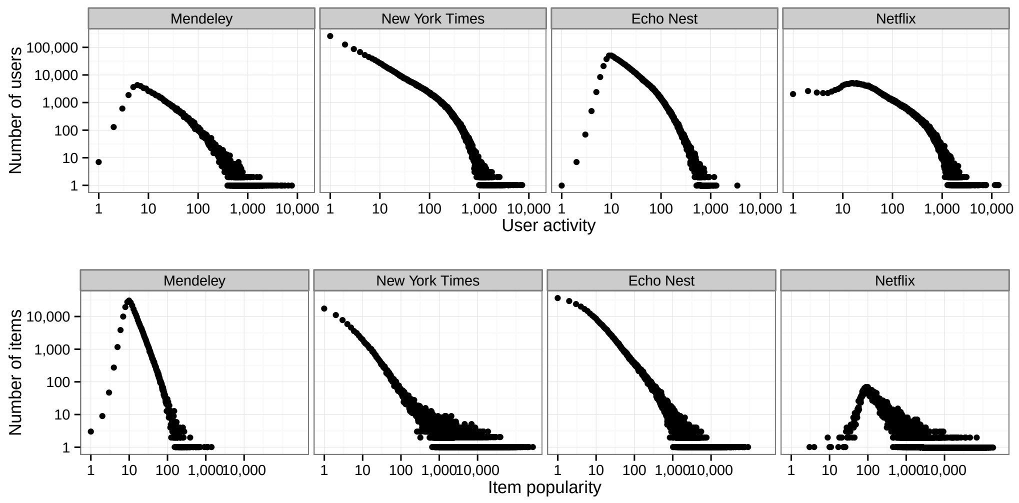


Figure 3.8: Long-tailed distribution of user activity and item popularity in four large user consumption data sets. The data include users listing articles in their libraries (Mendeley), users rating songs (Echo Nest) and users rating movies (Netflix). The data sets are described in Section 4.1.

model of user behavior data is how well it captures these distributions. On both explicit and implicit data, we found that the HPF captures them very well.

To check this, we implemented a *posterior predictive check* (PPC) (Rubin, 1984; Gelman *et al.*, 1996), a technique for model assessment from the Bayesian statistics literature. The idea behind a PPC is to simulate a complete data set from the posterior predictive distribution—the distribution over data that the posterior induces—and then compare the generated data set to the true observations. A good model will produce data that captures the important characteristics of the observed data.

We developed a PPC for matrix factorization algorithms on user behavior data. First, we formed posterior estimates of user preferences and item attributes for both classical MF and the HPF. Then, from these estimates, we simulated user behavior by drawing values for each user and item. (For classical matrix factorization, we truncated these values at zero and rounded to one in order to generate a plausible matrix.) Finally, we compared the matrix generated by the posterior predictive distribution to the true observations.

Figure 3.9 illustrates our PPC for the Netflix data. In this figure, we illustrate three distributions over user activity: the observed distribution (squares), the distribution from a data set replicated by the HPF (red line), and a distribution from a data set replicated by Gaussian MF with generated negatives using popularity-based sampling (blue line). The HPF captures the truth much more closely than Gaussian MF (with subsampled zeros), which badly overestimates the distribution

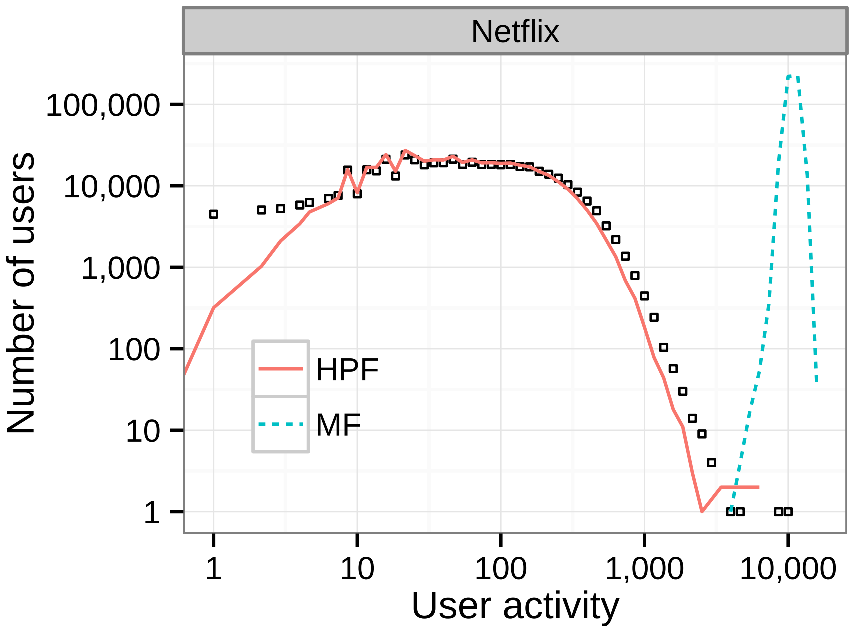


Figure 3.9: A posterior predictive check of the distribution of total ratings for the Netflix data set. The pink curve shows the empirical count of the number of users who have rated a given number of items, while the green and blue curves show the simulated totals from fitted Poisson and the Gaussian MF (with subsampled zeros) models, respectively. The Poisson marginal closely matches the empirical, whereas this variant of Gaussian MF fits a large mean to account for skew in the distribution and the missing ratings.

of user activity. This is likely an effect of the subsampled negatives. This misfit leads to an over-weighting of the zeros, which explains why practitioners require complex methods for downweighting them (Hu *et al.*, 2008; Gantner *et al.*, 2012; Dror *et al.*, 2012b; Paquet and Koenigstein, 2013).

The Gaussian MF based algorithm proposed by Hu *et al.* (2008) treats all missing ratings as observed zeroes. It downweights zeros in the weighted MF objective, capturing greater uncertainty over the zeros, to control their effect on predictions. A comparison to Hu *et al.* (2008), in capturing user activity and in predictive performance, is ongoing work at the time of writing of this thesis.

The PPC indicates that the HPF better represents real data, at least in comparison to Gaussian MF with subsampled negatives, and when measured by its ability to capture distributions of user activity.

Rewriting additive PF models

Although we've studied the PPC for a specific model—the HPF—additive PF models, in general, capture the marginal user and item counts well. To see this, we rewrite the Poisson observation model as a two stage process where a user u first decides on a budget b_u she has to spend on items, and then spends this budget rating items that she is interested in:

$$b_u \sim \text{Poisson}(\theta_u^T \sum_i \beta_i)$$
$$[y_{u1}, \cdots, y_{uM}] \sim \text{Mult}(b_u, \frac{\theta_u^T \beta_i}{\theta_u^T \sum_i \beta_i}).$$

This shows that learning a PF model for user-item ratings is effectively the same as learning user budgets and how they choose to spend their budgets.

With an appropriate fit to user activity, the model has two ways of explaining an unconsumed item: either the user is not interested in it or she would be interested in it but is likely to not be further active or has run out of resources. In contrast, a user that consumes an item must be interested in it. Thus, the model benefits more from making a consumed user/item pair more similar than making an unconsumed user/item pair less similar. This leads to an implicit downweighting of the unwatched or unconsumed items.

As an example, consider two similar science fiction movies, “Star Wars” and “The Empire Strikes Back”, and consider a user who has seen one of them. The Gaussian model pays an equal penalty for making the user similar to these items as it does for making the user different from them—with quadratic loss, seeing “Star Wars” is evidence for liking science fiction, but not seeing “The Empire Strikes Back” is evidence for disliking it. The Poisson model, however, will prefer to bring the user’s latent weights closer to the movies’ weights because it favors the information from the user watching “Star Wars”. Further, because the movies are similar, this increases the Poisson model’s predictive score that a user who watches “Star Wars” will also watch “The Empire Strikes Back”.

3.4.2 Shared, sparse latent factors

PF models can capture multiple types of discrete data, for example, the article text and user ratings of the articles. Each observation is drawn from a Poisson with the rate specified as an inner product of two vectors. The models can easily accommodate the sharing of latent spaces. The CTPF model of Section 3.3.3 is an example. While sparse representations are fundamental to Poisson factorization—as they are to NMF with KL-divergence based loss function—the Gamma priors on user preferences and item attributes in the models of Section 3.3 provide further control over the sparse representations of users and items.

3.4.3 Fast inference with sparse matrices

We will see that the inference algorithms for the models of Section 3.3 need only iterate over the viewed items in the observed matrix of user behavior, i.e., the non-zero elements, and this is true even for implicit or "positive only" data sets. This follows from two useful properties that hold for the APF models.

1. **Dependence of likelihood on consumed items only.** The likelihood of the observed data under the models of Section 3.3 depend only on the consumed items, that is, the non-zero elements of the user/item matrix $\boldsymbol{y}$. This facilitates computation for the kind of sparse matrices we observe in real-world data.

   We can see this property from the form of the Poisson distribution. Given the latent preferences θ_u and latent attributes β_i, the Poisson distribution of the rating y_{ui} is

$$p(y_{ui} \,|\, \theta_u, \beta_i) = \left(\theta_u^\top \beta_i\right)^y \exp\left\{-\theta_u^\top \beta_i\right\} / y_{ui}! \tag{3.3}$$

   Recall the elementary fact that $0! = 1$. The log probability of the complete matrix y is

$$\begin{aligned} \log p(y \,|\, \theta, \beta) = &\left(\textstyle\sum_{\{y_{ui}>0\}} y_{ui} \log(\theta_u^\top \beta_i) - \log y_{ui}!\right) \\ &- \left(\textstyle\sum_u \theta_u\right)^\top \left(\textstyle\sum_i \beta_i\right). \end{aligned} \tag{3.4}$$

2. **Conjugacy through auxiliary variables.** Another useful property of the Poisson distribution is that the sum of independent Poisson random variables is itself a Poisson with rate equal to the sum of the rates. This allows for *auxiliary* latent variables that modify a nonconjugate model to being conditionally conjugate.

   Conditionally conjugate models enjoy simple, standard variational inference algorithms (see Chapter 2). For example, for each user and item in the PF models of Section 3.3, we add K latent variables $z_{uik} \sim \text{Poisson}(\theta_{uk}\beta_{ik})$, which are integers that sum to the user/item value y_{ui}. These new latent variables preserve the marginal distribution of the observation, $y_{ui} \sim \text{Poisson}(\theta_u^\top \beta_i)$. These variables can be thought of as the contribution from component k to the total observation y_{ui}. Note that when $y_{ui} = 0$, these auxiliary variables are not random—the posterior distribution of z_{ui} will place all its mass on the zero vector. Consequently, any inference procedure need only consider z_{ui} for those user/item pairs where $y_{ui} > 0$. We include more details in Section 3.6.

As a consequence of these properties, APF models take advantage of the natural sparsity of user behavior data and can easily analyze massive real-world data.

3.5 Related work

In this section, we discuss the roots of Poisson factorization, and compare our models to alternative models of discrete data in the literature.

Non-negative matrix factorization. The roots of Poisson factorization come from nonnegative matrix factorization (Lee and Seung, 1999), where the objective function is equivalent to a factorized Poisson likelihood. More recently, the original NMF update equations have been shown to be an expectation-maximization (EM) algorithm for maximum likelihood estimation of a Poisson model via data augmentation (Cemgil, 2009). That NMF is a "Poisson version" of Latent Dirichlet Allocation (Blei *et al.*, 2003) was first pointed out by Buntine (2002), and proven in Gaussier and Goutte (2005).

Similarly to Cemgil (2009), our latent variable modeling perspective of NMF lends itself to powerful Bayesian extensions, such as the combined model of text and ratings in Section 3.3.3, and a Bayesian nonparametric model in Chapter 5. We demonstrate better predictive performance than NMF in Chapter 4.

Log-linear models. Dunson and Herring (2005) present a Poisson variable framework for discrete outcomes with an underlying simple Poisson log-linear model. These models are motivated by applications to tumor studies where covariates are typically available. The observed outcome y_{ui} is linked to an underlying Poisson variable z_{ui}, which results in a regression model for y_{ui}. The z_{ui} are modeled using the Poisson log-linear model. This generalized linear model is useful because it simultaneously ensures that the expected count is non-negative, while capturing a multiplicative effect of predictors on the mean. These models can link a vector of mixed discrete outcomes to a vector of underlying variables. The dependency between these mixed outcomes is captured through a standard Poisson-gamma shared frailty model, which scales the predictor with a shared Gamma variable (Dunson and Herring, 2005). The CTPF captures multiple discrete data —the article text and user ratings of the articles—using a linear model, rather than a log-linear model. Also, the dependency between these outcomes is modeled differently. We borrow the data augmentation strategy from Dunson and Herring (2005).

Gamma-Poisson models. Placing a Gamma prior on the user preferences along with normalized item attributes results in the GaP model (Canny, 2004), which was developed as an alternative text model to LDA (Blei *et al.*, 2003; Inouye *et al.*, 2014). The GaP model is fit using the expectation-maximization algorithm to obtain point estimates for user preferences and item attributes.

The Probabilistic Factor Model (PFM) (Ma *et al.*, 2011) improves upon GaP by placing a Gamma prior on the item weights as well, and using multiplicative update rules to infer an approximate maximum a posteriori estimate of the latent factors. In contrast, our models use a hierarchical prior structure of Gamma priors on user and item weights, and Gamma priors over the rate parameters from which these weights are drawn. This enables us to accurately model the skew in user activity and item popularity, which contributes to good predictive performance. Furthermore, we approximate the full posterior over all latent factors using a scalable variational inference algorithm.

Other applications. Independently of GaP and user behavior models, Poisson factorization has been studied in the context of signal processing for source separation (Cemgil, 2009; Hoffman, 2012) and for the purpose of detecting community structure in network data (Ball *et al.*, 2011). This research includes variational approximations to the posterior, though the issues and details around these data differ significantly from user data we consider and our derivation below (based on auxiliary variables) is more direct.

Recommendation algorithms. When modeling implicit feedback data sets, researchers have proposed merging factorization techniques with neighborhood models (Koren, 2008), weighting techniques to adjust the relative importance of positive examples (Hu *et al.*, 2008), and sampling-based approaches to create informative negative examples (Gantner *et al.*, 2012; Dror *et al.*, 2012b; Paquet and Koenigstein, 2013). In addition to the difficulty in appropriately weighting or sampling negative examples, there is a known selection bias in provided ratings that causes further complications (Marlin and Zemel, 2009; Marlin *et al.*, 2012).

Although Poisson factorization does not require such special adjustments, the downweighting it naturally induces on zero observations may not work well for all data sets. In preliminary results comparing to the Gaussian MF model (with downweighted zeros) of Hu *et al.* (2008), we obtained mixed results. In Gaussian MF, the downweighting of zeros translates to lower confidence and increased variance on the zero observations; the ability to capture variance over classes of ratings is an area of future work for the APF models.

Models of text and user ratings. Several research efforts propose joint models of item covariates and user activity. Singh and Gordon (2008) present a framework for simultaneously factorizing related matrices, using generalized link functions and coupled latent spaces. Hong *et al.* (2013) propose Co-factorization machines for modeling user activity on twitter with tweet features, including content. They study several design choices for sharing latent spaces. While CTPF is roughly an instance of these frameworks, we focus on the task of recommending articles to readers.

Agarwal and Chen (2010) propose fLDA, a latent factor model which combines document features through their empirical LDA (Blei *et al.*, 2003) topic intensities and other covariates, to predict user preferences. The coupling of matrix decomposition and topic modeling through shared latent variables is also considered in Shan and Banerjee (2010). Like fLDA, both papers tie latent spaces without corrective terms. Wang and Blei (2011) have shown the importance of using corrective terms through the collaborative topic regression (CTR) model which uses a latent topic offset to adjust a document's topic proportions. CTR has been shown to outperform a variant of fLDA (Wang and Blei, 2011). Our proposed model CTPF uses the CTR approach to sharing latent spaces.

CTR (Wang and Blei, 2011) combines topic modeling using LDA (Blei *et al.*, 2003) with Gaussian matrix factorization for one-class collaborative filtering (Hu *et al.*, 2008). Like CTPF, the underlying MF algorithm has a per-iteration complexity that is linear in the number of non-zero observations. Unlike CTPF, CTR is not conditionally conjugate, and the inference algorithm depends on numerical optimization of topic intensities. Further, CTR requires setting confidence parameters that govern uncertainty around a class of observed ratings. As we show in Chapter 4, CTPF scales more easily and provides better recommendations than CTR.

We discuss additional related recommendation methods in Section 4.1, where we evaluate the APF models.

3.6 Inference using Variational Bayes

In this section, we develop variational inference algorithms for the HPF and the CTPF models. The algorithm for the BPF model is an instance of the HPF algorithm, with a subset of the parameters held fixed.

3.6.1 The hierarchical model

Using the HPF for recommendation hinges on solving the posterior inference problem. Given a set of observed ratings, we would like to infer the user preferences and item attributes that explain these ratings, and then use these inferences to recommend new content to the users. In this section we discuss the details and practical challenges of posterior inference for the HPF, and present a mean-field variational inference algorithm as a practical and scalable approach. Our algorithm easily accommodates data sets with millions of users and hundreds of thousands of items on a single CPU.

Given a matrix of user behavior, we would like to compute the posterior distribution of user preferences θ_{uk}, item attributes β_{ik}, user activity ξ_u and item popularity η_i. As discussed in Chapter 2, the exact posterior is computationally intractable and we use mean-field variational inference.

Recall from Chapter 2 that variational inference is an optimization-based strategy for approximating posterior distributions in complex probabilistic models (Jordan *et al.*, 1999; Wainwright and Jordan, 2008). Variational algorithms posit a family of distributions over the hidden variables, indexed by free "variational" parameters, and then find the member of that family that is closest in Kullback-Liebler (KL) divergence to the true posterior.

The algorithm

We will describe a simple variational inference algorithm for the HPF. To do so, however, we first give an alternative formulation of the model in which we add an additional layer of latent variables. These auxiliary variables facilitate derivation and description of the algorithm (Ghahramani and Beal, 2001; Hoffman *et al.*, 2013).

As discussed in Section 3.4, we add K latent variables $z_{uik} \sim \text{Poisson}(\theta_{uk}\beta_{ik})$, which are integers that sum to the user/item value y_{ui}. With these latent variables in place, we have a conditionally conjugate model; we can now derive a standard variational inference algorithm. First, we posit the variational family over the hidden variables. Then we show how to optimize its parameters to find the member close to the posterior of interest.

The latent variables in the model are user weights θ_{uk}, item weights β_{ik}, and user-item contributions z_{uik}, which we represent as a K-vector of counts z_{ui}. The *mean-field family* considers these variables to be independent and each governed by its own distribution (see Chapter 2),

$$q(\beta, \theta, \xi, \eta, z) = \quad \prod_{i,k} q(\beta_{ik} \mid \lambda_{ik}) \prod_{u,k} q(\theta_{uk} \mid \gamma_{uk}) \prod_{u} q(\xi_u \mid \omega_u) \prod_{i} q(\eta_i \mid \tau_i) \prod_{u,i} q(z_{ui} \mid \phi_{ui}).$$

The variational factors for preferences θ_{uk}, attributes β_{ik}, activity ξ_u, and popularity η_i are all Gamma distributions, with freely set scale and rate variational parameters. The variational factor for z_{ui} is a free multinomial, i.e., ϕ_{ui} is a K-vector that sums to one. This form stems from z_{ui} being a bank of Poisson variables conditional on a fixed sum y_{ui}, and the property that such conditional

For all users and items, initialize the user parameters γ_u, ω_u^{rte} and item parameters λ_i, τ_i^{rte} to the prior with a small random offset. Set the user activity and item popularity shape parameters:

$$\omega_u^{\text{shp}} = a' + Ka; \quad \tau_i^{\text{shp}} = c' + Kc$$

Repeat until convergence:

1. For each user/item such that $y_{ui} > 0$, update the multinomial:

$$\phi_{ui} \propto \exp\{\Psi(\gamma_{uk}^{\text{shp}}) - \log \gamma_{uk}^{\text{rte}} + \Psi(\lambda_{ik}^{\text{shp}}) - \log \lambda_{ik}^{\text{rte}}\}.$$

2. For each user, update the user weight and activity parameters:

$$\begin{aligned} \gamma_{uk}^{\text{shp}} &= a + \textstyle\sum_i y_{ui}\phi_{uik} \\ \gamma_{uk}^{\text{rte}} &= \frac{\omega_u^{\text{shp}}}{\omega_u^{\text{rte}}} + \textstyle\sum_i \lambda_{ik}^{\text{shp}}/\lambda_{ik}^{\text{rte}} \\ \omega_u^{\text{rte}} &= \frac{a'}{b'} + \sum_k \frac{\gamma_{uk}^{\text{shp}}}{\gamma_{uk}^{\text{rte}}} \end{aligned}$$

3. For each item, update the item weight and popularity parameters:

$$\begin{aligned} \lambda_{ik}^{\text{shp}} &= c + \textstyle\sum_u y_{ui}\phi_{uik} \\ \lambda_{ik}^{\text{rte}} &= \frac{\tau_i^{\text{shp}}}{\tau_i^{\text{rte}}} + \textstyle\sum_u \gamma_{uk}^{\text{shp}}/\gamma_{uk}^{\text{rte}} \\ \tau_i^{\text{rte}} &= \frac{c'}{d'} + \sum_k \frac{\lambda_{ik}^{\text{shp}}}{\lambda_{ik}^{\text{rte}}} \end{aligned}$$

Figure 3.10: Variational inference for the hierarchical Poisson factorization (HPF) model. Each iteration only needs to consider the non-zero elements of the user/item matrix.

Poissons are distributed as a multinomial (Johnson *et al.*, 2005; Cemgil, 2009).

After specifying the family, we fit the variational parameters $\nu = \{\lambda, \gamma, \omega, \tau, \phi\}$ to minimize the KL divergence to the posterior, and then use the corresponding variational distribution $q(\cdot \mid \nu^*)$ as its proxy. The mean-field factorization facilitates both optimizing the variational objective and downstream computations with the approximate posterior, such as the recommendation score of Equation 5.18.

We optimize the variational parameters with the coordinate ascent algorithm described in Section 2.2.3, iteratively optimizing each parameter while holding the others fixed. The algorithm is illustrated in Figure 3.10. We denote shape with superscript "shp" and rate with superscript "rte".

Note that our algorithm is efficient on sparse matrices. In step 1, we need only update variational multinomials for the non-zero user/item observations y_{ui}. In steps 2 and 3, the sums over users and items need only to consider non-zero observations. This efficiency is thanks the likelihood of the full matrix only depending on the non-zero observations, as discussed in Section 3.4.

We terminate the algorithm when the variational distribution converges. Convergence is measured by computing the prediction accuracy on a validation set. Specifically, we approximate the probability that a user consumed an item using the variational approximations to posterior expectations of θ_u and β_i, and compute the average predictive log likelihood of the validation ratings. The HPF algorithm stops when the change in log likelihood is less than 0.0001%. For the HPF and the BPF we find that the algorithm is largely insensitive to small changes in the hyper-parameters. To enforce sparsity, we set the shape hyperparameters a', a, c and c' to provide exponentially shaped prior Gamma distributions. We fixed each hyperparameter at 0.3. We set the hyperparameters b' and d' to 1, fixing the prior mean at 1.

3.6.2 The combined model

Given a set of observed document ratings and their word counts, the goal is to infer the topics $\kappa_{1:K}$, the user preferences $\theta_{1:U}$, the document topic intensities $\mu_{1:D}$, the offsets $\beta_{1:D}$, and then use these inferences to recommend in-matrix documents and out-of-matrix documents to users. To this end, our goal is to compute the posterior distribution $p(\kappa_{1:V}, \mu_{1:D}, \beta_{1:D}, \theta_{1:U} | \boldsymbol{w}, \boldsymbol{r})$. As with the HPF, we use mean-field variational inference (Jordan *et al.*, 1999). We first develop a simple coordinate ascent algorithm—a batch algorithm that iterates over only the non-zero document-word counts and the non-zero user-document ratings.

Proceeding as in Section 3.6.1, we first augment the model with auxiliary variables to obtain a conditionally conjugate model. We then define the mean-field variational family and derive a coordinate ascent algorithm.

Auxiliary variables

To facilitate inference, we augment the CTPF model with auxiliary variables. Following (Dunson and Herring, 2005), we add K latent variables $z_{iv,k} \sim \text{Poisson}(\kappa_{ik}\mu_{vk})$, which are integers such that $w_{iv} = \sum_k z_{iv,k}$. Further, for each observed rating r_{ui}, we add K latent variables $y^a_{ui,k} \sim \text{Poisson}(\theta_{uk}\mu_{ik})$ and K latent variables $y^b_{ui,k} \sim \text{Poisson}(\theta_{uk}\beta_{ik})$ such that $r_{ui} = \sum_k y^a_{ui,k} + y^b_{ui,k}$. As in Section 3.6.1, these new latent variables preserve the marginal distribution of the observations, w_{iv} and r_{ui}, and the CTPF model with the auxiliary variables is conditionally conjugate.

The complete conditionals for the CTPF model from Section 3.3.3 augmented with auxiliary variables is shown in Table 3.1. As with the HPF, notice that z_{iv} is a set of Poisson variables, which

Latent Variable	Type	Complete conditional	Variational parameters
μ_{ik}	Gamma	$g + \sum_v z_{iv,k} + \sum_u y^a_{ui,k}, h + \sum_v \kappa_{vk} + \sum_u \theta_{uk}$	$\tilde{\mu}^{\text{shp}}_{ik}, \tilde{\mu}^{\text{rte}}_{ik}$
κ_{vk}	Gamma	$g + \sum_d z_{iv,k}, h + \sum_i \mu_{ik}$	$\tilde{\kappa}^{\text{shp}}_{vk}, \tilde{\kappa}^{\text{rte}}_{vk}$
θ_{uk}	Gamma	$a + \sum_i y^a_{ui,k} + \sum_i y^b_{ui,k}, \xi_u + \sum_i(\mu_{ik} + \beta_{ik})$	$\tilde{\theta}^{\text{shp}}_{uk}, \tilde{\theta}^{\text{rte}}_{uk}$
β_{ik}	Gamma	$c + \sum_u y^b_{ui,k}, \eta_i + \sum_u \theta_{uk}$	$\tilde{\beta}^{\text{shp}}_{ik}, \tilde{\beta}^{\text{rte}}_{ik}$
z_{iv}	Mult	$\log \mu_{ik} + \log \kappa_{vk}$	ϕ_{iv}
y_{ui}	Mult	$\begin{cases} \log \theta_{uk} + \log \mu_{ik} & \text{if } k < K, \\ \log \theta_{uk} + \log \beta_{ik} & \text{if } K \leq k < 2K \end{cases}$	τ_{ui}

Table 3.1: The complete conditionals of the latent variables in CTPF.

Initialize the topics $\kappa_{1:K}$ and topic intensities $\mu_{1:D}$ using LDA Blei *et al.* (2003).
Repeat until convergence:

1. For each word count $w_{iv} > 0$, set ϕ_{iv} to the expected conditional parameter of z_{iv}.
2. For each rating $r_{ui} > 0$, set τ_{ui} to the expected conditional parameter of y_{ui}.
3. For each document i and each k, update the block of variational topic intensities $\tilde{\mu}_{ik}$ to their expected conditional parameters. Perform similar block updates for $\tilde{\kappa}_{vk}$, $\tilde{\theta}_{uk}$ and $\tilde{\beta}_{ik}$, in sequence.

Figure 3.11: The CTPF coordinate ascent algorithm. The expected conditional parameters of the latent variables are computed from Table 3.1.

when conditional on a fixed sum w_{iv}, is distributed as a multinomial (Johnson *et al.*, 2005; Cemgil, 2009). A similar reasoning underlies the conditional for y_{iv}. With our complete conditionals in place, we now derive the coordinate ascent algorithm for the expanded set of latent variables.

Coordinate ascent algorithm

We first define the mean-field variational over the latent variables in CTPF. The complete conditionals in Table 3.1 show that the variational distributions are in the same exponential family as the conditional; we can therefore optimize each coordinate in closed form. The coordinate ascent algorithm is illustrated in Figure 3.11. In step 1, we need only update variational multinomials for the non-zero word counts w_{iv} and the non-zero ratings r_{ui}. In steps 2 and 3, the sums over the expected $z_{iv,k}$ and the expected $y_{ui,k}$ need only to consider non-zero observations.

Stochastic algorithm

The CTPF coordinate ascent algorithm is efficient. With linear-time per-iteration complexity, the algorithm can compute approximate posteriors for datasets with ten million observations within hours (see Chapter 4). To fit to larger datasets, within hours, we develop an algorithm that subsamples a document and estimates variational parameters using stochastic variational inference (SVI) Hoffman *et al.* (2013). We refer the reader to Chapter 2 for background on SVI. The stochastic algorithm is also useful in settings where new items continually arrive in a stream.

To obtain noisy gradients for SVI, assume that we operate under the setting where we subsample a single document d uniformly at random from the D documents. This sampling strategy is similar to online LDA (Hoffman *et al.*, 2010a). However, our approach differs in the use of separate learning rates for each user, allowing the inference to update only the relevant users in each iteration.

We are given observations about a single document in each iteration. Following Hoffman *et al.* (2013), we use the conditional dependencies in our graphical model to divide our variational parameters into *local* and *global*. The multinomial parameters (ϕ_{iv}, ξ_{ui}) for the sampled document i and for all $u \in U$, and the Gamma parameters for (μ_{ik}, β_{ik}) are local. All other variational parameters are global.

In each iteration of our algorithm, we first subsample a document. We then update the local multinomial parameters and the local topic intensities and offset parameters for this document using the coordinate updates from Figure 3.11. This optimizes local parameters with respect to the subsample. We then compute scaled natural gradients Amari (1982) for the global user preference parameters $(\tilde{\theta}^{\text{shp}}_{uk}, \tilde{\theta}^{\text{rte}}_{uk})$ for the users u that have rated document i and for all topic parameters $(\tilde{\kappa}^{\text{shp}}_{vk}, \tilde{\kappa}^{\text{rte}}_{vk})$. The global step for the global parameters follows the noisy gradient with an appropriate step-size.

We maintain separate learning rates ρ_u for each user, and only update the users who have rated the document i. We proceed similarly for words. We maintain a global learning rate ρ' for the topic parameters, which are updated in each iteration. For each of these learning rates ρ, we require that $\sum_t \rho(t)^2 < \infty$ and $\sum_t \rho(t) = \infty$ for convergence to a local optimum (Robbins and Monro, 1951). We set $\rho(t) = (\tau_0 + t)^{-\kappa}$, where $\kappa \in (0.5, 1]$ is the learning rate and $\tau_0 \geq 0$ downweights early iterations (Hoffman *et al.*, 2013).

Computational efficiency

The stochastic algorithm is more efficient than the batch algorithm. The batch algorithm has a per-iteration computational complexity of $O((W + R)K)$ where R and W are the total number of non-zero observations in the document-user and document-word matrices, respectively. For the SVI algorithm, this is $O((w_i + r_i)K)$ where r_i is the number of users rating the sampled document i and w_i is the number of unique words in it. (We assume that a single document is sampled in each iteration.) In Figure 3.11, the sums involving the multinomial parameters can be tracked for efficient memory usage. The bound on memory usage is $O((D + V + U)K)$.

Hyperparameters, initialization and stopping criteria

We fix each Gamma shape and rate hyperparameter at 0.3. We initialize the variational parameters for θ_{uk} and β_{ik} to the prior on the corresponding latent variables and add small uniform noise. We initialize $\tilde{\kappa}_{vk}$ and $\tilde{\mu}_{ik}$ using estimates of their normalized counterparts from LDA (Blei *et al.*, 2003) fitted to the document-word matrix $\boldsymbol{w}$. For the SVI algorithm described in the Appendix, we set learning rate parameters $\tau_0 = 1024, \kappa = 0.5$ and use a mini-batch size of 1024. In both algorithms, we declare convergence when the change in expected predictive likelihood is less than 0.001%.

3.7 A unified approach to modeling discrete outcomes

Many modern prediction tasks involve analyzing massive data sets of discrete outcomes. In topic modeling, it is standard to represent a corpus as a document-word matrix where each cell is the word frequency in a document (Blei *et al.*, 2003). In recommendation tasks, user behavior is represented as a user-item matrix of counts where each cell represents the users' rating or consumption of an item (Koren *et al.*, 2009). Community detection algorithms commonly represent a network using an adjacency matrix where each cell is a binary or weighted link between nodes (Newman, 2003).

Machine learning algorithms depend on capturing the statistical properties of discrete outcomes to obtain a good fit to the data and to predict well. The desired representations and properties are shared across different types of discrete data domains:

- **Data sparsity:** Most documents contain a small fraction of words in a vocabulary, most users rate a small fraction of movies, and most web pages link to a small number of other web pages. This leads to a sparse matrix of observed outcomes, with rows of non-negative integers dominated by zeros. For instance, the large data sets we analyze in Chapter 4 and Chapter 7 have less than 0.01% non-zero entries.

- **Long-tailed distributions:** Word frequencies in natural languages, user activity in rating products, and degree distributions of web pages, are all known to follow some form of power-law or characteristics of long-tailed distributions (Sato and Nakagawa, 2010; Paquet and Koenigstein, 2013; Clauset *et al.*, 2009).

- **Sparse factors:** For a given observation, only a small number of latent factors may be relevant. For example, although a large number of communities may be needed to explain a network, each node is likely to participate in a few communities. Similarly, a small number of preferences may underlie a user's movie choices and a small number of topics may explain a document (Blei *et al.*, 2003; Bengio *et al.*, 2013).

These shared properties motivate us to take a unified approach to modeling discrete outcomes of user behavior, text and networks. Although the APF models we develop in this chapter focus on recommendation using user behavior and text data, they can be applied to other types of discrete data, such as those arising from network interactions.

3.8 Conclusion

We presented additive Poisson factorization (APF) models of discrete data with applications in recommendation and exploratory tasks. In an APF model each observation is drawn from a Poisson distribution whose rate parameter is an inner product of vectors of non-negative latent variables. We presented BPF and HPF, both models of user behavior, and CTPF, a combined model of readership and article text. CTPF couples the user preference for the article content and the article's readership. This allows the user preferences to be interpreted as affinity to latent topics.

The APF models capture real-world user activity, provide sparse latent representations and enable efficient inference algorithms. We derived efficient variational inference for these models, and developed a SVI algorithm for CTPF.

In this chapter, we have modeled both count and binary data using Poisson distributions. For binary data, a natural approach is to use censored Poisson distributions (Greene, 2005)—we observe whether $y_{ui} > 0$. Another line of work is to combine other data sources relevant to recommendation such as user's social networks and item covariates. Finally, in our study of the models' ability to capture user activity, we compared to Gaussian MF with subsampled negatives. A natural alternative is the algorithm of Hu *et al.* (2008) – it observes the full matrix and downweights zero observations in the MF objective. This helps capture greater uncertainty around those observations.

In Chapter 4, we evaluate these models on a variety of real-world recommendation problems–users rating movies, users listening to songs, users reading scientific papers, and users reading news articles. We find that the APF models are robust to hyperparameter settings, making them a good off-the-shelf tool for recommendation; they are efficient building blocks for more sophisticated models such as those with Bayesian nonparametric assumptions. We develop Bayesian nonparametric Poisson factorization in Chapter 5.

Chapter 4

Scalable recommendation

In this chapter we study the additive PF models of Chapter 3 on large-scale user behavior data sets: users listening to music, users watching movies, users reading scientific articles, and users reading the newspaper. We study the CTPF on two data sets of scientific article content and readers listing these articles. For each data set, we estimated and examined the posterior distributions of the Poisson parameters and used them to recommend items to users.

The study in Section 4.1 demonstrates that the HPF outperforms competing methods such as nonnegative matrix factorization (Lee and Seung, 1999), topic models (Blei *et al.*, 2003), and one variant of Gaussian matrix factorization (Koren *et al.*, 2009); and the study in Section 4.2 demonstrates that CTPF outperforms collaborative topic regression (Wang and Blei, 2011). Further, we provide validation of the PF models as an exploratory tool in Section 4.3. In particular, we show that the CTPF organizes articles according to their topics and identifies important articles both in terms of those important to their topic and those that have transcended disciplinary boundaries.

Our experiments reveal that the additive PF models of Section 3.3 are robust to hyperparameter settings and can be used as off-the-shelf tools; they are efficient building blocks for more sophisticated models such as those with Bayesian nonparametric assumptions. We will develop Bayesian nonparametric Poisson factorization in Chapter 5.

A main limitation of our study is we do not include a comparison to Hu *et al.* (2008), as this effort is ongoing. Preliminary results suggest that the variant of Gaussian MF proposed in Hu *et al.* (2008) outperforms APF models on some data sets. Their method works by downweighting the contribution of zeros to the MF objective, capturing greater uncertainty around them. Extending the APF models to capture greater uncertainty around a class of ratings is future work.

4.1 In-matrix recommendations

In this section, we evaluate the performance of the hierarchical Poisson factorization (HPF) algorithm from Section 3.3.2 and its non-hierarchical variant (BPF) from Section 3.3.1 on a variety of large-scale user behavior data sets. We first discuss the details of each data set and of the competing recommendation methods. We then describe our study, noting the good predictive performance and computational efficiency of HPF. We conclude with an exploratory analysis of preferences and attributes on two of the data sets.

Data Sets

We study the HPF algorithm in Figure 5.1 on both implicit and explicit feedback:

- The **Mendeley** data set (Jack *et al.*, 2010) of scientific articles is a binary matrix of 80,000 users and 260,000 articles, with 5 million observations. Each cell corresponds to the presence or absence of an article in a scientist's online library.

- The **Echo Nest** music data set (Bertin-Mahieux *et al.*, 2011) is a matrix of 1 million users and 385,000 songs, with 48 million observations. Each observation is the number of times a user played a song.

- The **New York Times** data set is a matrix of 1,615,675 users and 103,390 articles, with 80 million observations. Each observation is the number of times a user viewed an article.

- The **Netflix** data set (Koren *et al.*, 2009) contains 480,000 users and 17,770 movies, with 100 million observations. Each observation is the rating (from 1 to 5 stars) that a user provided for a movie.

The scale and diversity of these data sets enables a robust evaluation of our algorithm. The Mendeley, Echo Nest, and New York Times data are sparse compared to Netflix. For example, we observe only 0.001% of all possible user-item ratings in Mendeley, while 1% of the ratings are non-zero in the Netflix data. This is partially a reflection of large number of items relative to number users in these data sets.

Furthermore, the intent signaled by an observed rating varies significantly across these data sets. For instance, the Netflix data set gives the most direct measure of stated preferences for items, as users provide an explicit star rating for movies they have watched. In contrast, article click counts in the New York Times data are a less clear measure of how much a user likes a given article—most articles are read only once, and a click through is only a weak indicator of whether the article was fully read, let alone liked. Ratings in the Echo Nest data presumably fall somewhere in between, as the number of times a user listens to a song likely reveals some indirect information about their preferences.

As such, we treat each data set as a source of implicit feedback, where an observed positive rating indicates that a user likes a particular item, but the rating value itself is ignored. The Mendeley data are already of this simple binary form. For the Echo Nest and New York Times data, we consider any song play or article click as a positive rating, regardless of the play or click count. We also consider two versions of the Netflix data—the original, explicit ratings, and an implicit version in which only 4 and 5 star ratings are retained as observations (Paquet and Koenigstein, 2013).

Competing methods

We compare the HPF algorithm of Figure 3.10 against an array of competing methods:

- **NMF**: Non-negative Matrix Factorization (Lee and Seung, 1999). In NMF, user preferences and item attributes are modeled as non-negative vectors in a low-dimensional space. These latent vectors are randomly initialized and modified via an alternating multiplicative update rule to minimize the Kullback-Leibler divergence between the actual and modeled rating matrices.

- **LDA**: Latent Dirichlet Allocation (Blei *et al.*, 2003). LDA is a Bayesian probabilistic generative model where user preferences are represented by a distribution over different topics, and each topic is a distribution over items. Interest and topic distributions are randomly initialized and updated using stochastic variational inference (Hoffman *et al.*, 2013) to approximate these intractable posteriors. The model is typically used for text, but also for more general discrete data.

- **MF**: Probabilistic Matrix Factorization with user and item biases. We use a variant of matrix factorization popularized through the Netflix Prize (Koren *et al.*, 2009), where a linear predictor—comprised of a constant term, user activity and item popularity biases, and a low-rank interaction term—is fit to minimize the mean squared error between the predicted and observed rating values, subject to L2 regularization to avoid overfitting. Weights are randomly initialized and updated via stochastic gradient descent using the Vowpal Wabbit package (Weinberger *et al.*, 2009). This corresponds to maximum a-posteriori inference under Probabilistic Matrix Factorization (Salakhutdinov and Mnih, 2008c).

We note that while HPF, BPF, and LDA take only the non-zero observed ratings as input, while the Gaussian MF that we study requires that we provide explicit zeros in the ratings matrix as negative examples for the implicit feedback setting. In practice, this amounts to either treating all missing ratings as zeros (as in NMF) and down-weighting to balance the relative importance of observed and missing ratings (Hu *et al.*, 2008), or generating negatives by randomly sampling from missing ratings in the training set (Gantner *et al.*, 2012; Dror *et al.*, 2012b; Paquet and Koenigstein, 2013). We take the latter approach for computational convenience, employing a popularity-based sampling scheme: we sample users by activity—the number of items rated in the training set—and items by popularity—the number of training ratings an item received to generate negative examples.[1]

A main limitation of our study is we do not include a comparison to Hu *et al.* (2008), as this effort is ongoing work. Preliminary results suggest that the variant of Gaussian MF proposed in Hu *et al.* (2008) may outperform APF models on some implicit data sets. Their method works by downweighting the contribution of zeros to the MF objective, rather than subsampling negative examples. This captures greater uncertainty around the negative/zero observations. Extending the APF models to capture greater uncertainty around a class of ratings is future work.

Finally, we note a couple of candidate algorithms that failed to scale to our data sets. The fully Bayesian treatment of the Probabilistic Matrix Factorization (Salakhutdinov and Mnih, 2008a), uses a MCMC algorithm for inference. Salakhutdinov and Mnih (2008a) report that a single Gibbs iteration on the Netflix data set with 60 latent factors, requires 30 minutes, and that they throw away the first 800 samples. This implies at least 16 days of training, while the HPF variational inference algorithm converges within 13 hours on the Netflix data. Another alternative, Bayesian Personalized Ranking (BPR) (Rendle *et al.*, 2009b; Gantner *et al.*, 2012), optimizes a ranking-based criteria using stochastic gradient descent. The algorithm performs an expensive bootstrap sampling step at each iteration to generate negative examples from the vast set of unobserved. We found time and space constraints to be prohibitive when attempting to use BPR with the data sets considered here.

Evaluation

Prior to training any models, we randomly select 20% of ratings in each data set to be used as a held-out test set comprised of items that the user has consumed. Additionally, we set aside 1% of the training ratings as a validation set and use it to determine algorithm convergence and to tune

[1]We also compared this to a uniform random sampling of negative examples, but found that the popularity-based sampling performed better.

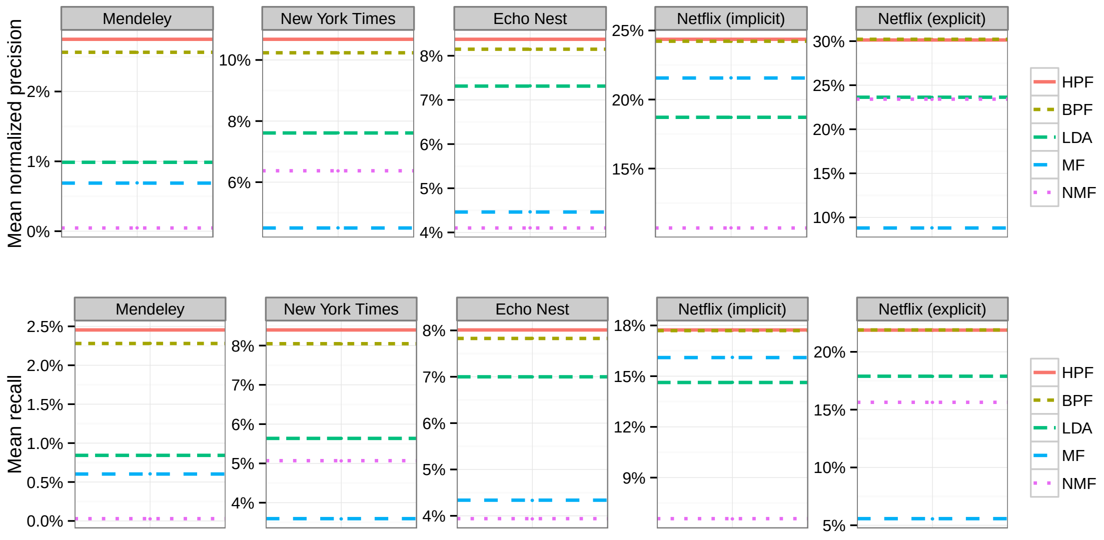


Figure 4.1: Predictive performance on data sets. The top and bottom plots show normalized mean precision and mean recall at 20 recommendations, respectively. While competing method performance varies across data sets, HPF and BPF has consistently good predictive performance.

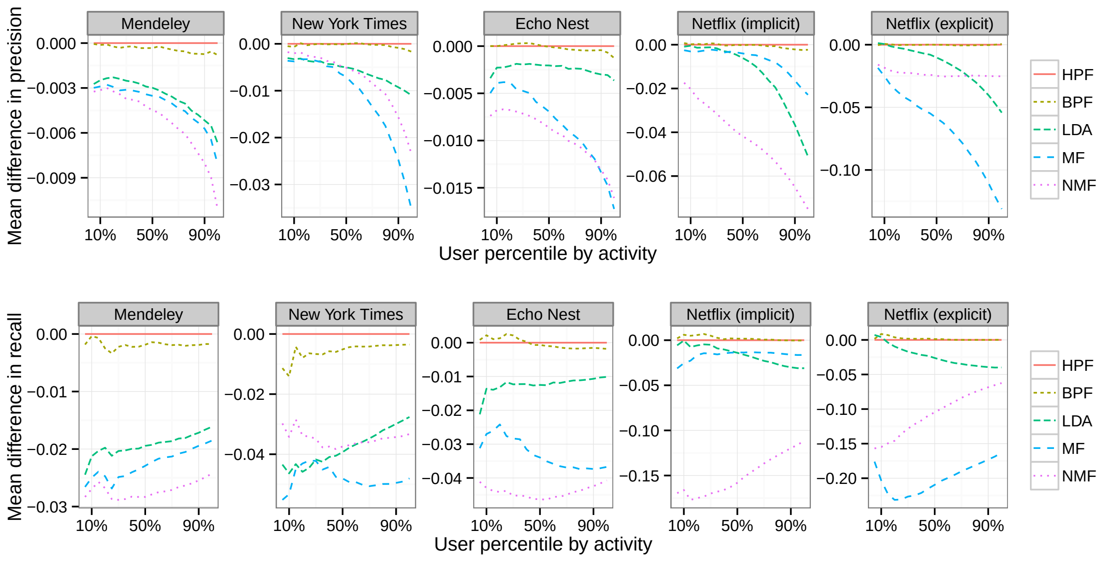


Figure 4.2: Predictive performance across users. The top and bottom plots show the mean difference in precision and recall to HPF at 20 recommendations, respectively, by user activity.

free parameters. We used the BPF and HPF settings described in Section 3.6 across all data sets, and set the number of latent components K to 100.

During testing, we generate the top M recommendations for each user as those items with the highest predictive score under each method. For each user, we compute a variant of precision-at-M that measures the fraction of relevant items in the user's top-M recommendations. So as not to artificially deflate this measurement for lightly active users who have consumed fewer than M items, we compute *normalized* precision-at-M, which adjusts the denominator to be at most the number of items the user has in the test set. Likewise, we compute recall-at-M, which captures the fraction of items in the test set present in the top M recommendations.

Figure 4.1 shows the normalized mean precision at 20 recommendations for each method and data sets. We see that HPF and BPF provide good predictive performance—a relatively high fraction of items recommended by HPF are found to be relevant, and many relevant items are recommended. While not shown in these plots, the relative performance of methods within a data set is consistent as we vary the number of recommendations shown to users.

We also study precision and recall as a function of user activity to investigate how performance varies across users of different types. In particular, Figure 4.2 shows the mean difference in precision and recall to HPF, at 20 recommendations, as we look at performance for users of varying activity, measured by percentile. For example, the 10% mark shows mean performance across the bottom 10% of users, who are least active; the 90% mark shows the mean performance for all but the top 10% of most active users. Here we see that Poisson factorization outperforms other methods for users of all activity levels—both the "light" users who constitute the majority, and the relatively few "heavy" users who consume more—for all data sets.

4.2 Out-of-matrix recommendations of scientific articles

We compared the predictive accuracy of the CTPF coordinate ascent algorithm in Figure 3.11 to collaborative topic regression (CTR) (Wang and Blei, 2011), and to the variants of CTPF. We demonstrate that CTPF outperforms its variants, and CTR. The comparison to CTPF variants is an attempt to validate it's modeling assumptions. Finally, we explore large real-world data sets revealing the interaction patterns between readers and articles.

Data sets

We study the CTPF algorithm on two data sets:

- The **Mendeley** data set (Jack *et al.*, 2010) of scientific articles is a binary matrix of 80,000 users and 260,000 articles with 5 million observations. Each cell corresponds to the presence or absence of an article in a scientist's online library.

- The **arXiv** data set is a matrix of 120,297 users and 825,707 articles, with 43 million observa-

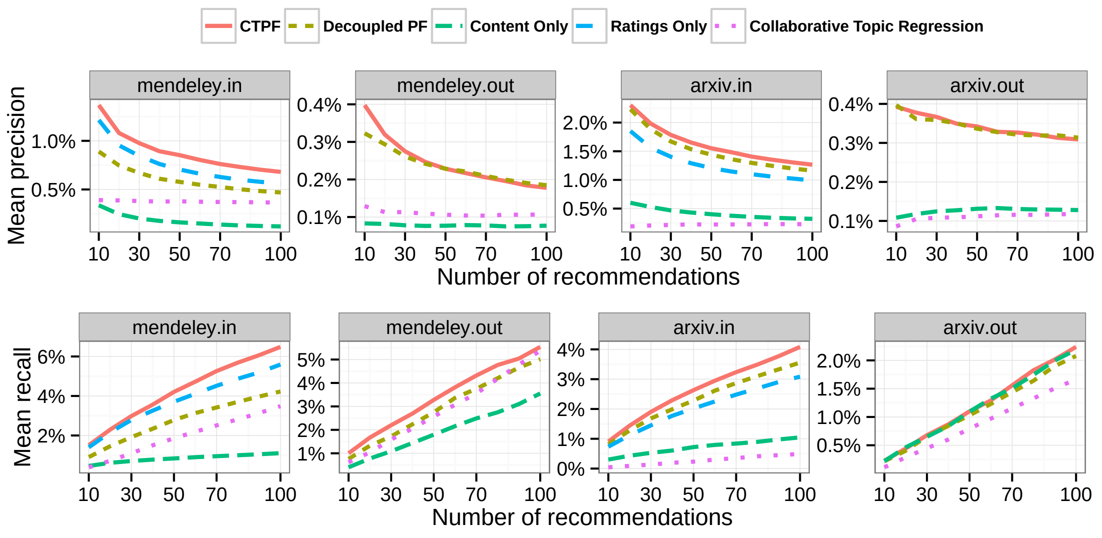


Figure 4.3: The CTPF model outperforms CTR and the CTPF variants on both in-matrix and out-matrix predictions. Each panel either the in-matrix or out-matrix recommendation task on a data set. Notice that the Ratings model cannot make out-matrix predictions. The mean precision and mean recall are computed from a random sample of 10,000 users. The CTPF models were trained on the Mendeley data set and the 1-year arXiv data set using the coordinate ascent variational inference algorithm from Figure 3.11.

tion. Each observation indicates whether or not a user has consulted an article (or its abstract). This data was collected from the access logs of registered users on the http://arXiv.org paper repository. The articles and the usage data spans a timeline of 10 years (2003-2012). In addition, in some of our experiments we use a subset of the dataset, with 64,978 users 636,622 papers and 7.6 million clicks, which spans one year of usage data (2012). We treat the user clicks as implicit feedback and specifically as binary data. For each article in the above data sets, we remove stop words and use tf-idf to choose the top 10,000 distinct words (14,000 for arXiv) as the vocabulary. We implemented the CTPF algorithm in 4500 lines of C++ code. [2]

Competing methods

We study the predictive performance of the following models. With the exception of BPF, which does not model content, the topics and topic intensities (or proportions) in all CTPF models are initialized using LDA (Blei *et al.*, 2003), and fit using batch variational inference. We set K = 100 in all of our experiments.

- **CTPF**: CTPF is our proposed model with latent user preferences tied to a single vector θ_u, and interpreted as affinity to latent topics κ.

[2]Our source code is available from: github.premgopalan.com/collabtm.

- **Decoupled Poisson Factorization**: This model is similar to CTPF but decouples the user latent preferences into distinct components p_u and q_u, each of dimension K. We have,

$$w_{dv} \sim \text{Poisson}(\mu_i^T \kappa_v); \quad r_{ud} \sim \text{Poisson}(p_u^T \mu_i + q_u^T \beta_i). \tag{4.1}$$

  The user preference parameters for content and ratings can vary freely. The q_u are independent of topics and offer greater modeling flexibility, but they are less interpretable than the η_u in CTPF. Decoupling the factorizations has been proposed by Porteous *et al.* (2010).

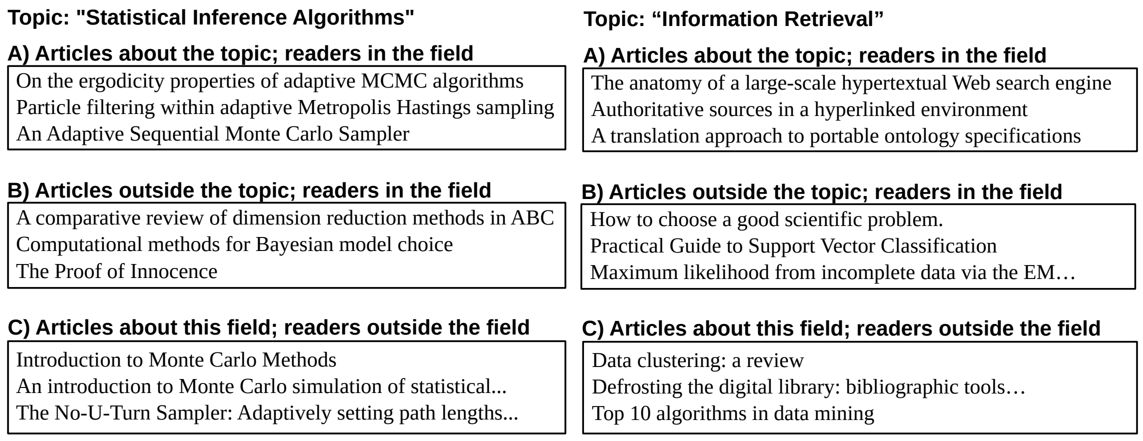
- **Content Only**: We use the CTPF model without the document topic offsets β_i. This resembles the idea developed in Agarwal and Chen (2010) but using Poisson generating distributions.

- **Ratings Only**: We use BPF from Section 3.3.1 to the observed ratings. This model can only make in-matrix predictions.

- **CTR (Wang and Blei, 2011)**: A full optimization of this model does not scale to the size of our data sets despite running for several days. Accordingly, we fix the topics and document topic proportions to their LDA values. This procedure is shown to perform almost as well as jointly optimizing the full model in Wang and Blei (2011). We follow the authors' experimental settings. Specifically, for hyperparameter selection we started with the values of hyperparameters suggested by the authors and explored various values of the learning rate as well as the variance of the prior over the correction factor (λ_v in Wang and Blei (2011)). Training convergence was assessed using the model's complete log-likelihood on the training observations. (CTR does not use a validation set.)

Evaluation

Prior to training models, we randomly select 20% of ratings and 1% of documents in each data set to be used as a held-out test set. Additionally, we set aside 1% of the training ratings as a validation set (20% for arXiv) and use it to determine algorithm convergence. We used the CTPF settings described in Section 3.6.2 across both data sets. During testing, we generate the top M recommendations for each user as those items with the highest predictive score under each method. For each user, we compute the precision-at-M, which measures the fraction of relevant items in the user's top-M recommendations. Similarly, we compute recall-at-M, which captures the fraction of items in the test set present in the top M recommendations.


Topic: "Statistical Inference Algorithms"

A) Articles about the topic; readers in the field

On the ergodicity properties of adaptive MCMC algorithms
Particle filtering within adaptive Metropolis Hastings sampling
An Adaptive Sequential Monte Carlo Sampler

B) Articles outside the topic; readers in the field

A comparative review of dimension reduction methods in ABC
Computational methods for Bayesian model choice
The Proof of Innocence

C) Articles about this field; readers outside the field

Introduction to Monte Carlo Methods
An introduction to Monte Carlo simulation of statistical...
The No-U-Turn Sampler: Adaptively setting path lengths...

Topic: “Information Retrieval”

A) Articles about the topic; readers in the field

The anatomy of a large-scale hypertextual Web search engine
Authoritative sources in a hyperlinked environment
A translation approach to portable ontology specifications

B) Articles outside the topic; readers in the field

How to choose a good scientific problem.
Practical Guide to Support Vector Classification
Maximum likelihood from incomplete data via the EM…

C) Articles about this field; readers outside the field

Data clustering: a review
Defrosting the digital library: bibliographic tools…
Top 10 algorithms in data mining

Figure 4.4: The top articles by the expected weight μ_{ik} from a component discovered by our stochastic variational inference in the arXiv data set (Left) and Mendeley (Right). Using the expected topic proportions μ_{ik} and the expected topic offsets β_{ik}, we identified subclasses of articles. A) corresponds to the top articles by topic proportions in the field of “Statistical inference algorithms” for arXiv and “Ontologies and applications” for Mendeley; B) corresponds to the top articles with low topic proportions in this field, but a large $\mu_{ik} + \beta_{ik}$, demonstrating the outside interests of readers of that field (e.g., very popular papers often appear such as “The Proof of Innocence” which describes a rigorous way to “fight your traffic tickets”). C) corresponds to the top articles with high topic proportions in this field but that also draw significant interest from outside readers.

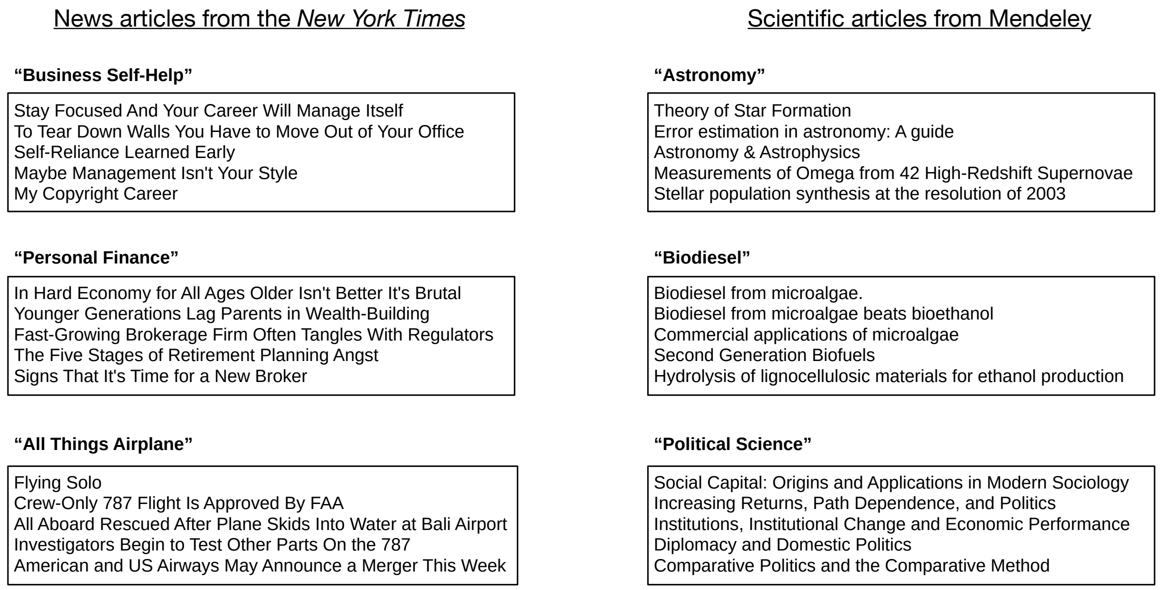
Results

Figure 4.3 shows the mean precision and mean recall at 20 recommendations for each method and data set. We see that CTPF outperforms CTR and the Ratings model on all data sets. CTPF outperforms the Decoupled PF model and the Content model on all data sets except on cold-start predictions on the arXiv data set. The Decoupled PF model lacks the CTPF’s interpretable latent space. The Content model performs poorly on most tasks due to the lack of a corrective term on topics to account for user ratings.

4.3 Exploratory analysis

In this section, we fit the HPF and the CPF to real-world user behavior data sets. Using the HPF, we explore the discovered components in the data, and interaction patterns between users and items. Using the CTPF, we explore scientific articles according to their topics, and identify important and inter-disciplinary articles.

In Figure 4.5, we explore the scientific articles in the Mendeley data set and the new articles in New York Times using the HPF model. The goal is to discover the latent structure among items and users and to confirm that the model is capturing the components in the data in a reasonable



News articles from the *New York Times*

"Business Self-Help"

Stay Focused And Your Career Will Manage Itself
To Tear Down Walls You Have to Move Out of Your Office
Self-Reliance Learned Early
Maybe Management Isn't Your Style
My Copyright Career

"Personal Finance"

In Hard Economy for All Ages Older Isn't Better It's Brutal
Younger Generations Lag Parents in Wealth-Building
Fast-Growing Brokerage Firm Often Tangles With Regulators
The Five Stages of Retirement Planning Angst
Signs That It's Time for a New Broker

"All Things Airplane"

Flying Solo
Crew-Only 787 Flight Is Approved By FAA
All Aboard Rescued After Plane Skids Into Water at Bali Airport
Investigators Begin to Test Other Parts On the 787
American and US Airways May Announce a Merger This Week

Scientific articles from Mendeley

"Astronomy"

Theory of Star Formation
Error estimation in astronomy: A guide
Astronomy & Astrophysics
Measurements of Omega from 42 High-Redshift Supernovae
Stellar population synthesis at the resolution of 2003

"Biodiesel"

Biodiesel from microalgae.
Biodiesel from microalgae beats bioethanol
Commercial applications of microalgae
Second Generation Biofuels
Hydrolysis of lignocellulosic materials for ethanol production

"Political Science"

Social Capital: Origins and Applications in Modern Sociology
Increasing Returns, Path Dependence, and Politics
Institutions, Institutional Change and Economic Performance
Diplomacy and Domestic Politics
Comparative Politics and the Comparative Method

Figure 4.5: The top 10 items by the expected weight β_i from three of the 100 components discovered by our algorithm for the New York Times and Mendeley data sets.

way. For example, in Figure 4.5 we illustrate the components discovered by our algorithm. For each data set, the illustration shows the top items—items sorted in decreasing order of their expected weight β_i—from three of the 100 components discovered by our algorithm. From these, we see that learned components both cut across and differentiate between conventional topics and categories. For instance, in the New York Times data, we find that multiple business-related topics (e.g., self help and personal finance) comprise separate components, whereas other articles that appear across different sections of the newspaper (e.g., business and regional news) are unified by their content (e.g., airplanes).

In Figure 4.4, we explored the scientific articles in Mendeley and the arXiv data set using CTPF. We fit the Mendeley data set using the coordinate ascent algorithm, and the full arXiv data set using the stochastic algorithm from Section 3.6.2. We discovered interpretable latent topics corresponding to all components. Using the expected topic proportions μ_{ik} and the expected topic offsets β_{ik}, we identified subclasses of articles that reveal the interaction patterns between readers and articles.

4.4 Conclusion

In this chapter, we evaluated the additive Poisson factorization models of Chapter 3. We demonstrated that the APF models have good predictive performance on a range of data sets, with implicit and explicit feedback. As noted, a comparison to Gaussian MF with downweighted zeros (Hu *et al.*,

2008) is important ongoing work. This includes extending the APF models to capture greater uncertainty around missing/zero ratings.

In the next chapter, we develop an additive PF model that infers the number of latent components using Bayesian nonparametric assumptions (Zhou *et al.*, 2012).

Chapter 5

Bayesian nonparametric Poisson factorization

In Chapter 3, we developed additive Poisson factorization models for recommendation systems. We fitted these models to large-scale user behavior data sets and predicted which items a user will like based on his or her history of purchases or ratings.

One of the main limitations of Poisson factorization (and in general, matrix factorization) is model selection, i.e., choosing the number of components with which to model the data. The typical approach in matrix factorization, Poisson or Gaussian, is to determine the number of components by predictive performance on a held-out set of ratings (Salakhutdinov and Mnih, 2008d). But this can be prohibitively expensive with large data sets because it requires fitting many models.

In this chapter, we develop a Bayesian nonparametric (BNP) Poisson factorization model. Our model, described in Section 5.1, is based on the weights from a collection of Gamma processes (Ferguson, 1973), one for each user, which share an infinite collection of atoms. Each atom represents a preference pattern of items, such as action movies or comedies. Through its posterior distribution, our model adapts the dimensionality of the latent representations, learning the preference patterns (and their number) that best describe the users. We discuss related models in Section 5.2.

As for with the BPF and the HPF models from Chapter 3, the main computational challenge is posterior inference, where we approximate the posterior distribution of the latent user/item structure given a data set of user/item ratings. We again develop an efficient algorithm based on variational inference in Section 5.3. Our method simultaneously finds both the latent components and the latent dimensionality, easily handles large data sets, and takes time roughly equal to one fit of the model with fixed dimension.

The contributions in this chapter are: (a) a new Bayesian nonparametric model for Poisson factorization (BNPPF), (b) a scalable variational inference algorithm with nested variational families (Kurihara *et al.*, 2006), and (c) a thorough study of BNPPF on large scale movie recommendation problems. On two large real-world data sets—10M movie ratings from MovieLens and 100M movie ratings from Netflix—we found that Bayesian nonparametric PF outperformed (or at least performed as well as) its parametric counterpart, which requires searching over K. Unlike previous algorithms in Section 5.2, our approach takes advantage of the sparsity and non-negativity to scale to very large data sets. We analyze data that cannot be handled by previous research. Finally, we note that our BNP model can be used in a a range of applications requiring matrix factorization, such as topic modeling (Canny, 2004) or community detection (Ball *et al.*, 2011).

5.1 An infinite model based on the Gamma process

We develop a statistical model of user/item rating matrices. As in Chapter 3, our data are observations y_{ui}, which contains the rating that user u gave to item i, or zero if no rating was given. The data can be based on "implicit feedback", where y_{ui} is one if the user consumed it and zero otherwise. User behavior data is typically sparse, i.e., most of the ratings are zero.

In the BPF model of recommendation from Chapter 3, each user and each item is associated with a K-dimensional latent vector of non-negative preferences and attributes, respectively. Let $\boldsymbol{\theta}_u = [\theta_{u1}, \ldots, \theta_{uK}]^\top$ be the preferences of user u, and $\boldsymbol{\beta}_i = [\beta_{i1}, \ldots, \beta_{iK}]^\top$ be the attributes of item i. In this chapter, we will jointly refer to these as "weights". These weights are given Gamma priors and each observation y_{ui} is modeled by a Poisson distribution parameterized by the inner product of the user's and item's weights,

$$\beta_{ik} \sim \text{Gamma}(a, b), \tag{5.1}$$

$$\theta_{uk} \sim \text{Gamma}(c, d), \tag{5.2}$$

$$y_{ui} \sim \text{Poisson}(\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i). \tag{5.3}$$

With the number of components K fixed, the standard approach to model selection is to fit the model with different values of K and select the one that gives best performance on a held-out set of ratings.

Choosing the value of K is a nuisance because it is expensive to fit many models on large data sets. We want a model with support over arbitrary K such that the posterior distribution captures the effective latent dimensionality of our data. The desirable properties for such a model are: (i) The user weights $\boldsymbol{\theta}_u$ and item weights $\boldsymbol{\beta}_i$ must be infinite-dimensional non-negative vectors, (ii) the expected dot product $\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i$ of any pair must be finite, and (iii) the item weights $\boldsymbol{\beta}_i$ have to be shared among all users.

We place independent and identical Gamma priors over each element of the infinite-dimensional vector $\boldsymbol{\beta}_i$, following Equation 5.1. Since all elements β_{ik} are iid, to satisfy condition (ii), we require the convergence of $\sum_{k=1}^{\infty} \mathrm{E}\left[\theta_{uk}\right]$. A natural way to construct a summable infinite set of positive weights is to use a Gamma process (Ferguson, 1973) for each user weight vector $\boldsymbol{\theta}_u$. A Gamma process, $\mathrm{GP}(c, H)$, has two parameters: c the rate parameter and H the base measure. A draw from a Gamma process is an atomic random measure with finite total mass when H is finite. This means that a draw from a Gamma process gives an infinite collection of positive-valued random weights (the atom weights) whose summation is almost surely finite. Thus, we make θ_{uk} the weights of a draw from a Gamma process.

To ensure the items are shared among all users and satisfy condition (iii), we need to match the item weights to the user weights. Notice that the item weights are indexed by the natural numbers. Thus, to match the user weights to the item weights, we need an ordering of the user weights. We use a size-biased ordering. The size-biased ordering promotes sharing by penalizing higher index components. We obtain per-user Gamma process weights, in size-biased order, by scaling the stick-breaking construction of the Dirichlet process with a Gamma random variable. Miller (2011) states this construction for Gamma processes with unit scale. In general, scaling the sticks from a Dirichlet process by a draw from $\mathrm{Gamma}(\alpha, c)$ yields a draw from a Gamma process with parameters c and H, where $H(\Omega) = \alpha$ (Zhou and Carin, 2013). While α governs the sparsity of the user weights, both α and c influence the rating budget available to the user.

Using the scaled sticks for $\boldsymbol{\theta}_u$, the generative process for the Bayesian nonparametric Poisson factorization model with N users and M items is as follows:

1. For each user $u = 1, \dots, N$:

   (a) Draw $s_u \sim \mathrm{Gamma}(\alpha, c)$.

   (b) Draw $v_{uk} \sim \mathrm{Beta}(1, \alpha)$,
   for $k = 1, \dots, \infty$.

   (c) Set $\theta_{uk} = s_u \cdot v_{uk} \prod_{i=1}^{k-1}(1 - v_{ui})$,
   for $k = 1, \dots, \infty$.

2. Draw $\beta_{ik} \sim \text{Gamma}(a, b)$,

   for $k = 1, \ldots, \infty$, $i = 1, \ldots, M$.

3. Draw $y_{ui} \sim \text{Poisson}(\sum_{k=1}^{\infty} \theta_{uk}\beta_{ik})$,

   for $u = 1, \ldots, N$, $i = 1, \ldots, M$.

Unlike draws from a standard Gamma process, these atoms are shared across users. In this way, our model is similar to a hierarchical Gamma process (Çinlar, 1975). However, in a hierarchical Gamma process, the atoms are shared across users through the common base measure. In contrast, the atoms in our model are shared due to the size-biased ordering. Intuitively, components with different indices are unlikely to be similar due to the penalty levied by the size-biased ordering. This method of sharing atoms can be leveraged in other hierarchical BNP models like the hierarchical Dirichlet process (Teh and Jordan, 2010).

Note that, unlike BNP Gaussian matrix factorization (Knowles and Ghahramani, 2011), the generative process provides a sparse observation matrix. The probability of y_{ui} being zero can be lower bounded as

$$p(y_{ui} = 0) \geq \exp\left(-\frac{a\alpha}{bc}\right) \tag{5.4}$$

and, therefore, the expected number of zeros in the observation matrix is lower bounded by $NM \exp(-\frac{a\alpha}{bc})$.

5.2 Related work

There has been significant research on Bayesian nonparametric factor models. Griffiths and Ghahramani (2011) introduced the IBP, and developed a Gaussian BNP factor model with binary weights. Knowles and Ghahramani (2011) later extended this work to non-binary weights. Other extensions include Hoffman *et al.* (2010b) and Porteous *et al.* (2008). Situated within this literature, our model is a BNP factor model that assumes non-negative weights and sparse observations.

Closer to our model is the work of Titsias (2007), Zhou *et al.* (2012), and Broderick *et al.* (2013a). Titsias (2007) derives a BNP factor model, the infinite Gamma-Poisson feature model. The Gamma process can be shown to be the De Finetti mixing distribution for this model (Thibaux, 2008), with the latent counts being drawn from a Poisson process. Our model does not have an underlying latent discrete stochastic process. Zhou *et al.* (2012) generalize Titsias (2007) and extend the infinite prior to a Beta-Gamma-Gamma-Poisson hierarchical structure. Our model is simpler than Zhou *et al.*

(2012) because it is not hierarchical, and our model choices afford a scalable variational inference algorithm to tackle the kinds of problems that we are trying to solve. Titsias (2007) uses an MCMC algorithm that does not scale and Zhou *et al.* (2012) further do inference with a truncated model (which also does not scale). Finally, Broderick *et al.* (2013a) give a more general class of hierarchical models than Zhou *et al.* (2012) but, again, only develop MCMC algorithms. Our model is simpler than and complements Broderick *et al.* (2013a)—Poisson factorization does not immediately fall out of their framework—and, again, affords scalable inference algorithms.

5.3 Inference using variational expectation-maximization

In this section, we derive a scalable mean-field inference algorithm for approximate posterior inference under the BNPPF. Recall from Chapter 2 that variational inference algorithms approximate the posterior by defining a parametrized family of distributions over the hidden variables and then fitting the parameters to find a distribution that is close to the posterior

Following our approach in Dunson and Herring (2005); Zhou *et al.* (2012) and Chapter 3, we introduce for each user-item pair the auxiliary latent variables $z_{ui,k}$, such that $z_{ui,k} \sim \text{Poisson}(\theta_{uk}\beta_{ik})$. Recall that due to the additive property of Poisson random variables, each observation y_{ui} can be expressed as

$$y_{ui} = \sum_{k=1}^{\infty} z_{ui,k}. \tag{5.5}$$

Thus, the variables $z_{ui,k}$ preserve the marginal Poisson distribution of the observation y_{ui}. Note that when $y_{ui} = 0$, the posterior distribution of $z_{ui,k}$ will place all its mass on $z_{ui,k} = 0$. Consequently, our inference procedure needs to only consider $z_{ui,k}$ for those user-item pairs such that $y_{ui} > 0$. This is not the case for BNP Gaussian MF (Knowles and Ghahramani, 2011) and makes our inference procedure extremely efficient for sparse user/item data.

Using the auxiliary variables, and introducing the notation $\boldsymbol{\beta} = \{\boldsymbol{\beta}_i\}$, $\boldsymbol{s} = \{s_u\}$, $\boldsymbol{v} = \{v_{uk}\}$ and $\boldsymbol{z} = \{z_{ui,k}\}$, the joint distribution over the hidden variables can be written as

$$p(\boldsymbol{z}, \boldsymbol{\beta}, \boldsymbol{s}, \boldsymbol{v}|\alpha, c, a, b) = \prod_{u=1}^{N} p(s_u|\alpha, c) \prod_{k=1}^{\infty}\prod_{u=1}^{N} p(v_{uk}|\alpha) \prod_{k=1}^{\infty}\prod_{i=1}^{M} p(\beta_{ik}|a, b) \prod_{k=1}^{\infty}\prod_{u=1}^{N}\prod_{i=1}^{M} p(z_{ui,k}|\theta_{uk}, \beta_{ik}), \tag{5.6}$$

and the observations are generated following Equation 5.5.

The mean-field family considers the latent variables to be independent of each other, yielding the completely factorized variational distribution:

$$q(\boldsymbol{z}, \boldsymbol{\beta}, \boldsymbol{s}, \boldsymbol{v}) = \prod_{u=1}^{N} q(s_u) \prod_{k=1}^{\infty} \prod_{u=1}^{N} q(v_{uk}) \prod_{k=1}^{\infty} \prod_{i=1}^{M} q(\beta_{ik}) \prod_{u=1}^{N} \prod_{i=1}^{M} q(\boldsymbol{z}_{ui}), \tag{5.7}$$

in which we denote by $\boldsymbol{z}_{ui}$ the vector with the infinite collection of variables $z_{ui,k}$ for user u and item i.

Typical mean field methods optimize the KL divergence by coordinate ascent, iteratively optimizing each parameter while holding the others fixed. Recall from Chapter 2 that these update are easy to derive for conditionally conjugate variables, i.e., variables whose complete conditional is in the exponential family (Ghahramani and Beal, 2001). This is the case for most of the variables in our model and for these variables we set the form of their variational distributions to be the same as their complete conditionals.

The exceptions are the stick proportions. These variables are not conditionally conjugate because the Beta prior over the stick proportions v_{uk} is not conjugate to the Poisson likelihood. Rather, the complete conditional for the stick proportions v_{uk} is a truncated Gamma distribution. Letting the variational distribution be in the prior or the complete conditional family results in coordinate updates that do not have a closed form. Therefore, we resort to a degenerate delta distribution for $q(v_{uk})$, i.e., $q(v_{uk}) = \delta_{\tau_{uk}}(v_{uk})$, an alternative that is widely used in the BNP literature (Liang *et al.*, 2007; Bryant and Sudderth, 2012).[1]

Finally, we handle the infinite collection of variational factors in Eq. 7 by adapting the technique of (Kurihara *et al.*, 2006), in which the variational families are nested over a truncation level T. We allow for an infinite number of components for the variational distribution, but we tie the variational distribution after level T to the prior. Specifically, $q(v_{uk})$ and $q(\beta_{ik})$ are set to the prior for $k \geq T+1$.

[1]In variational inference, minimizing the KL divergence is equivalent to maximizing an objective function called ELBO (evidence lower bound). When the support of the variational distribution and the true posterior do not coincide, maximizing the ELBO is not equivalent to minimizing the KL divergence. In our case, $q(v_{uk})$ is a degenerate delta function and, therefore, its support is not the whole interval $[0, 1]$. The resulting algorithm that maximizes the ELBO can be understood instead as a variational expectation maximization algorithm (Beal and Ghahramani, 2003).

Given a truncation level T, for all users and items, initialize the user scaling parameters $\{\gamma_{u,0}, \gamma_{u,1}\}$ and the item parameters $\{\lambda_{ik,0}, \lambda_{ik,1}\}$ to the prior with a small random offset. Initialize the stick proportions τ_{uk}, where $k \leq T$, randomly.

Repeat until convergence:

1. For each user/item pair such that $y_{ui} > 0$,
   - Update the multinomial parameter $\boldsymbol{\phi}_{ui}$ using Equation 5.16.
2. For each user,
   - Update the user scaling parameters $\gamma_{u,0}$ and $\gamma_{u,1}$ using Equation 5.9 and Equation 5.10.
   - Update the user stick proportions τ_{uk} for all $k \leq T$ using Equation 5.12.
3. For each item,
   - Update the item weight parameters $\lambda_{ik,0}$ and $\lambda_{ik,1}$ using Equation 5.13 and Equation 5.14 for all $k \leq T$.

Figure 5.1: Batch variational inference for the Bayesian nonparametric Poisson factorization model. Each iteration only needs to consider the non-zero elements of the user/item matrix.

With these considerations in mind, the forms of the variational distributions in Equation 5.7 are:

$$
\begin{aligned}
q(s_u) &= \mathrm{Gamma}(s_u | \gamma_{u,0}, \gamma_{u,1}), \\
q(v_{uk}) &= \begin{cases} \delta_{\tau_{uk}}(v_{uk}), & \text{for } k \leq T, \\ p(v_{uk}), & \text{for } k \geq T+1, \end{cases} \\
q(\beta_{ik}) &= \begin{cases} \mathrm{Gamma}(\beta_{ik} | \lambda_{ik,0}, \lambda_{ik,1}), & \text{for } k \leq T, \\ p(\beta_{ik}), & \text{for } k \geq T+1, \end{cases} \\
q(\boldsymbol{z}_{ui}) &= \mathrm{Mult}(\boldsymbol{z}_{ui} | y_{ui}, \boldsymbol{\phi}_{ui}). \qquad (5.8)
\end{aligned}
$$

We now describe the specific updates for each variational parameter. Figure 5.1 gives the algorithm.

1. The update equations for the *user scaling parameters* $\gamma_{u,0}$ and $\gamma_{u,1}$ are given by

$$\gamma_{u,0} = \alpha + \sum_{i=1}^{M} y_{ui}, \qquad (5.9)$$

$$\gamma_{u,1} = c + \mathrm{E}\left[\sum_{k=1}^{\infty} v_{uk} \left(\prod_{j=1}^{k-1} (1 - v_{uj}) \right) \sum_{i=1}^{M} \beta_{ik} \right], \qquad (5.10)$$

where $\mathrm{E}[\cdot]$ denotes expectation with respect to the distribution q. The infinite sum in the update equation for $\gamma_{u,1}$ can be split into the sum of the terms for $k \le T$ and the sum of the terms for $k \ge T+1$. The first sum is straightforward to compute, but the second one involves infinitely many terms. However, it results in a convergent geometric series whose value is given by

$$\begin{aligned} &\mathrm{E}\left[\sum_{k=T+1}^{\infty} v_{uk} Y_{uT} \left(\prod_{j=T+1}^{k-1} (1 - v_{uj})\right) \sum_{i=1}^{M} \beta_{ik}\right] \\ &= Y_{uT} D, \end{aligned} \tag{5.11}$$

where $Y_{uT} = \prod_{k=1}^{T}(1 - \tau_{uk})$ and $D = \sum_{i=1}^{M} \mathrm{E}\left[\beta_{i(T+1)}\right] = Ma/b$.

2. The update equations for the *stick proportions* τ_{uk} can be obtained by taking the derivative of the objective function with respect to τ_{uk} and setting it equal to zero. This yields the quadratic equation $A_{uk}\tau_{uk}^2 + B_{uk}\tau_{uk} + C_{uk} = 0$, where the coefficients A_{uk}, B_{uk} and C_{uk} are provided in the Supplementary Material. Solving for τ_{uk}, we get

$$\tau_{uk} = \frac{-B_{uk} \pm \sqrt{B_{uk}^2 - 4A_{uk}C_{uk}}}{2A_{uk}}, \tag{5.12}$$

and we discard the solution that is not in $[0, 1]$. Note that we require $\alpha > 1$ for the variational objective to be a concave function of τ_{uk}.

3. For the *item weights*, the equations for the variational parameters $\lambda_{ik,0}$ and $\lambda_{ik,1}$ are straightforward due to the conditional conjugacy of the distributions involved:

$$\lambda_{ik,0} = a + \sum_{u=1}^{N} y_{ui}\phi_{ui,k}, \tag{5.13}$$

$$\lambda_{ik,1} = b + \sum_{u=1}^{N} \mathrm{E}\left[s_u v_{uk} \prod_{j=1}^{k-1}(1 - v_{uj})\right]. \tag{5.14}$$

4. Exploiting the Gamma-Poisson conjugacy, we know that the optimal $q(\boldsymbol{z}_{ui})$ can be parameterized by an infinite-dimensional vector $\boldsymbol{\phi}_{ui}$ whose components take the form

$$\phi_{ui,k} \propto \exp\{\mathrm{E}[\log \theta_{uk}] + \mathrm{E}[\log \beta_{ik}]\}. \tag{5.15}$$

Let $R_{ui,k} = \mathrm{E}\left[\log \theta_{uk}\right] + \mathrm{E}\left[\log \beta_{ik}\right]$. Then,

$$\phi_{ui,k} = \frac{\exp\{R_{ui,k}\}}{\sum_{k=1}^{\infty} \exp\{R_{ui,k}\}}. \tag{5.16}$$

Similarly to the derivation by (Kurihara *et al.*, 2006), we are left with computing a normalizer that is an infinite sum. The summation up to the truncation level T is straightforward, and thus we focus on computing $\sum_{k=T+1}^{\infty} \exp\{R_{ui,k}\}$. It is also a convergent geometric series, which can be computed as

$$\sum_{k=T+1}^{\infty} \exp\{R_{ui,k}\} = \frac{\exp\{R_{ui,T+1}\}}{1 - \exp\left\{\mathrm{E}\left[\log(1 - v_{u(T+1)})\right]\right\}}. \tag{5.17}$$

We have described our batch variational inference algorithm for the BNPPF. We emphasize that the algorithm needs to only iterate over the nonzero elements. For a dataset with N users, M items and r ratings, the algorithm in Figure 5.1 has a computational complexity of $O(T^2N + TM + Tr))$. The dominant cost is the iteration over ratings captured by the $O(Tr)$ term, which equals the cost for the finite Bayesian Poisson factorization (BPF) model from Chapter 3 with a fixed number of components $K = T$. This allows us, in the next section, to analyze very large user/item data sets.

5.4 An empirical study of model selection

In this section, we compare the Bayesian nonparametric Poisson factorization model (BNPPF) and the Bayesian Poisson factorization (BPF) model from Section 3.3.1 on three different data sets. Our goal is to demonstrate that the BNPPF variational inference algorithm can avoid model selection, while yielding better or similar performance than the variational inference algorithm for its finite counterpart. On large data sets, model selection involves fitting the finite model on a wide range of the latent dimensionality K, making it computationally intractable. We also show that our inference algorithm scales, and that it runs in roughly the same time needed for a single run of the finite model. We do not compare to Gaussian Bayesian nonparametric matrix factorization algorithms which require iteration over all elements of the user/item matrix. We demonstrated in Chapter 4 that the finite algorithm outperforms Gaussian MF.

We implemented the algorithm of Figure 5.1 in $4,000$ lines of C++ code. The input to this algorithm is the user/item matrix. The output are the parameters for the approximate posterior

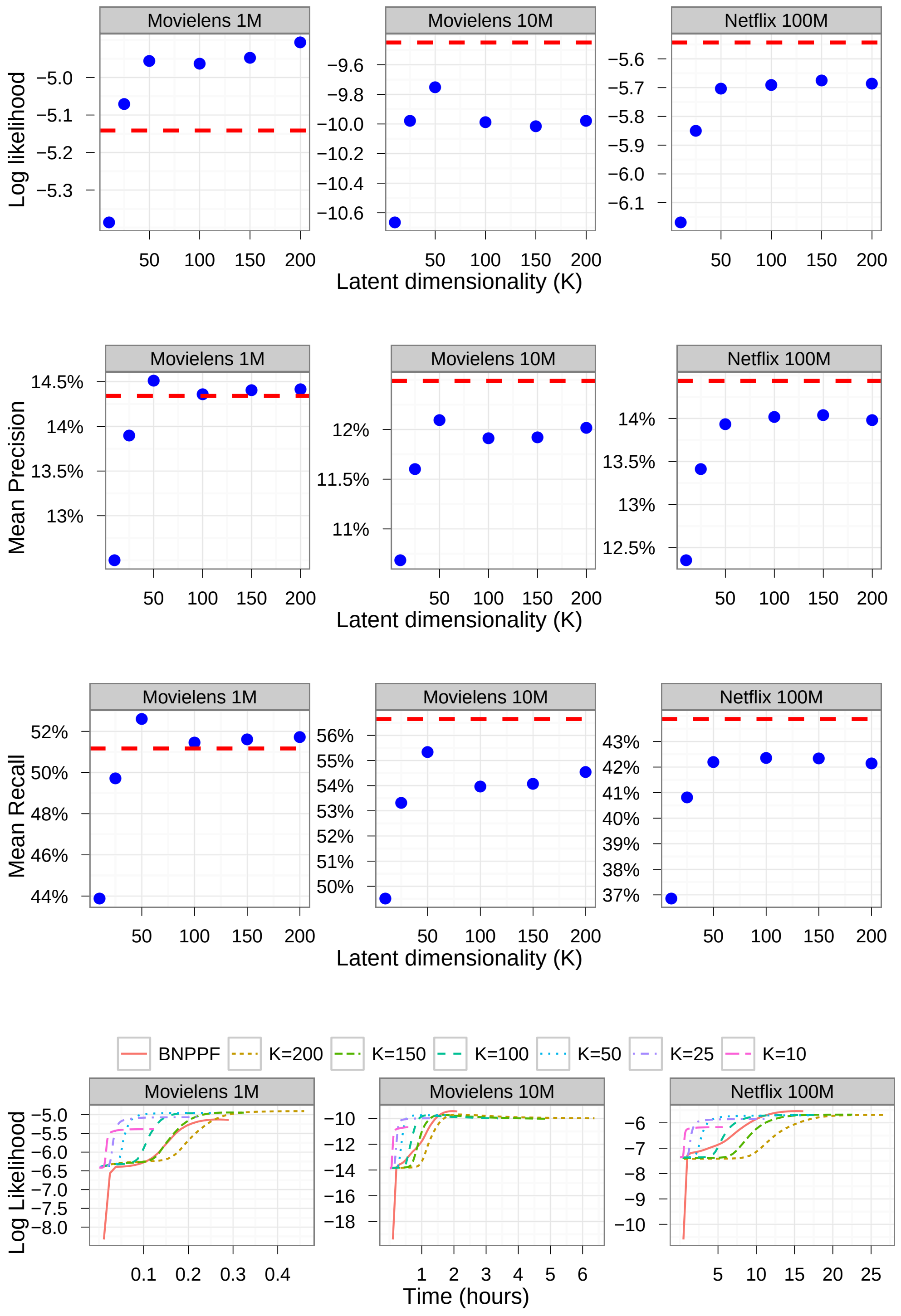


Figure 5.2: Rows 1-3: Generalization performance on the MovieLens and the Netflix data sets. The data sets vary in size from 1 million ratings to 100 million ratings. Points indicate the finite model predictive performance with varying dimensionality K, while the horizontal line indicates the BNPPF model performance. The BNPPF model demonstrates better predictive log likelihood for the two largest data sets and is at least as good as the finite models in terms of mean precision and recall. Row 4: Predictive log likelihood as a function of run-time for all the considered models. The BNPPF model is as fast as a single run of the finite model with $K = T$.

distribution over the user and item weights.[2]

Data sets

We study the predictive performance and runtime of the BNPPF model on these data sets:

- The **MovieLens** data set (Herlocker *et al.*, 1999) contains 1 million (MovieLens1M) ratings of movies provided by users, with $6,040$ users and $3,980$ movies. The ratings range from 0 (no rating) to 5 stars.

- The **MovieLens** dataset with 10 million (MovieLens10M) ratings of movies, $71,567$ users and $10,681$ movies. Ratings are made on a 5-star scale, with half-star increments. We multiply times 2 to get a scale from 0 to 10 "half stars".

- The **Netflix** dataset (Koren *et al.*, 2009), which is similar to MovieLens1M dataset but significantly larger. It contains 100 million ratings, with $480,000$ users and $17,770$ movies. Unlike MovieLens, the Netflix data is highly skewed: some users rate more than $10,000$ movies, while others rate less than 5.

Metrics

As figures of merit, we use the quantities below, measured over a held-out test set which is not observed during training. For each dataset, the test set consists of randomly selected ratings which make up 20% of the total number of ratings. This test set consists of items that the users have consumed. During training, these test set observations are treated as zeros.

- Predictive log likelihood (or mean held-out log likelihood). We approximate the probability that a user consumed an item using the variational approximations to the posterior expectations of the hidden variables. For the BNPPF model, we compute the expectation $\mathrm{E}\left[\sum_{k=1}^{\infty} \theta_{uk}\beta_{ik}\right]$ exactly using a convergent geometric series as in the updates in Equation 5.11. We use these expectations to compute the average predictive log likelihood of the held-out ratings.

- Mean precision and recall. Once the posterior is fit, we use the BNPPF to recommend items to users by predicting which of the unconsumed items each user will like. We rank each user's

[2]Our software is available at https://github.com/premgopalan/bnprec.

unconsumed items by their posterior expected Poisson parameters,

$$\text{score}_{ui} = \mathrm{E}\left[\sum_{k=1}^{\infty} \theta_{uk}\beta_{ik}\right], \tag{5.18}$$

where $\mathrm{E}[\cdot]$ denotes expectation with respect to the distribution q. During testing, we generate the top M recommendations for each user as those items with the highest predictive score under each method. For each user, we compute the precision-at-M, which measures the fraction of relevant items in the user's top-M recommendations. Likewise, we compute recall-at-M, which is the fraction of items in the test set present in the top M recommendations. We set M to 100 in our experiments. We computed the mean precision and mean recall over $10,000$ randomly chosen users (for MovieLens1M, we compute the mean over all users).

Hyperparameter selection

In order to set the hyperparameters, we first notice that both the finite and infinite models share the same prior on the item weights and, therefore, we use the same hyperparameter values for a fair comparison (specifically, we set $a = b = 0.3$ for both models). Due to the stick breaking construction, the prior on the user weights differs between the finite and the infinite models. For the BNPPF model, we set the user scaling hyperparameter $c = 1$ and $\alpha = 1.1$ (recall from Section 5.3 that we require $\alpha > 1$). For the finite model, we explored the values in the set $\{0.1, 1, 10\}$ for both the shape and scale in Equation 5.2, and choose unit shape and unit scale because these values provided the best performance on the test set in terms of predictive log likelihood (see Supplementary Material for a comparison). We use these values in all our experiments.

Assessing convergence

We terminate the training process when the algorithm converges. We measure convergence by computing the prediction accuracy on a validation set, composed of 1% randomly selected ratings, which are treated as zeros during training and are not considered to measure performance. The algorithm stops either when the change in log likelihood on this validation set is less than 0.0001%, or if the log likelihood decreases for consecutive iterations.

Results

In Figure 5.2, we show the results for the two MovieLens data sets with 1 million and 10 million ratings, and the Netflix dataset with 100 million ratings. The top row corresponds to mean held-out log likelihood, while the second and third rows correspond, respectively, to mean precision and recall. We set the truncation level for the BNPPF to $T = 200$ across all data sets. As seen in Figure 5.2, the BNPPF model has better held-out log likelihood with a fixed truncation level than the corresponding finite models, as we vary K from 10 to 200 (with the exception of the small MovieLens1M dataset). Further, the BNPPF model performs at least as well as the finite models in the mean precision and mean recall metrics.

For the BNPPF model, we computed the effective dimensionality K^* of the latent weights. In order to identify K^*, for each user, we identify the top latent components $k \leq T$, that contribute to at least 95% of the user's expected budget under the approximate posterior, i.e., $\mathrm{E}\left[\sum_{k=1}^{\infty} \theta_{uk} \sum_{i=1}^{M} \beta_{uk}\right]$. We rank the latent components by their contribution to the expected budget. Across all users, it gives us the effective dimensionality of the latent weights. For the MovieLens1M dataset, we found $K^* = 92$, and for MovieLens10M and Netflix data sets, we found that all latent components were used with $T = 200$.

The last row of Figure 5.2 shows that the BNPPF model runs as fast as the inference algorithm for the finite model with K set to the truncation level of 200. As discussed in Section 5.3, the dominant computational cost is the same for these algorithms.

5.5 Limitations and conclusions

In this chapter, we developed a Bayesian nonparametric Poisson factorization model for recommendation systems. In our BNP variant, the number of latent components is theoretically unbounded and effectively estimated when computing a posterior with observed user behavior data. To approximate the posterior, we developed an efficient variational inference algorithm. It adapts the dimensionality of the latent components to the data, only requires iteration over the user/item pairs that have been rated, and has computational complexity on the same order as for the parametric model (the BPF) with fixed dimensionality. We studied our model and algorithm on large real-world data sets of user-movie preferences. Our model eases the computational burden of searching for the number of latent components and gives better predictive performance than its parametric counterpart: the BPF from Chapter 3.

There are two limitations. First, as discussed in Section 5.3, the choice of degenerate variational distributions constrains α to be greater than 1. We focussed on movie data sets in this work. With other types of data, there may be increased sparsity in user preferences or item attributes. One example is that of users reading scientific articles, where each user is likely to be interested in articles belonging to a very small number of research areas. This may require a prior setting of $\alpha \leq 1$. Second, the use of point estimates for the stick proportions may result in overfitting. Both limitations can be overcome by placing a non-degenerate variational distribution over the stick proportions, which may come at an increased computational cost due to numerical optimization.

Part II

Networks

Chapter 6

Assortative mixed-membership stochastic blockmodels

Network analysis is vital to understanding and predicting interactions between network entities (Fortunato, 2010; Liben-Nowell and Kleinberg, 2003; Newman, 2002). Examples of data commonly represented as networks include gene regulation networks (Trevio *et al.*, 2012), communication networks (Aral and Walker, 2012), co-authorship networks (Chen and Redner, 2010) and social networks (Newman and Girvan, 2004).

In this chapter, we consider a central problem in network analysis: discovering *overlapping communities* of connected nodes and *popular nodes* in the network, and using this latent structure to predict unobserved links (Liben-Nowell and Kleinberg, 2003). We take a Bayesian approach and present two probabilistic models of undirected networks. We focus on undirected networks revealing the modeling and algorithmic issues that arise even with the simpler assumption. The models we present in this chapter can be extended to directed networks.

Modeling assortativity through latent communities

In Section 6.1.1, we posit a model of undirected networks (Goldenberg *et al.*, 2010) where each node can belong to multiple communities. We use a subclass of the mixed-membership stochastic blockmodel (MMSB) (Airoldi *et al.*, 2008) that allows nodes to belong to multiple communities.

The assumption of mixed or multiple memberships for nodes is an important one. Classical community detection algorithms assume that each node belongs to a single community (Fortunato, 2010; Newman and Girvan, 2004; Nowicki and Snijders, 2001; Wiggins and Hofman, 2008; Clauset *et al.*, 2004). In real-world networks, each node will likely belong to multiple communities and its connections will reflect these multiple memberships (Derényi *et al.*, 2005; Ahn *et al.*, 2010). For example, in a large social network, a member may be connected to co-workers, friends from school, and neighbors. We need algorithms that discover overlapping communities to capture the heterogeneity of each node's connections. We will show in Chapter 7 that our MMSB based community model provides better fits to real network data than single community models (Nowicki and Snijders, 2001; Wang and Wong, 1987).

We use a sub-class of the MMSB model that assumes *assortativity*—a property of nodes to connect to nodes that are *similar* in some way. In particular, we assume that nodes connect when they belong to similar latent communities. In this thesis, We refer to this sub-class as "MMSB". As an example, consider protein-protein interactions (Airoldi *et al.*, 2008), which can be analyzed to reveal the functional roles of proteins. The MMSB model encodes the assumption that two proteins interact because they belong to similar groups, either through co-location or through cellular operations.

Capturing node popularity

The assortativity assumption alone may not be adequate in other contexts. For example, on the Internet a user's web page may link to another user due to a similar interest in skydiving. The same user's page may also link to popular web pages such as Google's search page. The competing explanation to assortativity is that nodes preferentially connect to popular nodes—the basis for preferential attachment (Jeong *et al.*, 2003). The resulting degree distributions from a preferential attachment process are known to satisfy empirically observed properties such as power laws (Papadopoulos *et al.*, 2012).

In Section 6.1.2 we present the assortative MMSB with node popularities (AMP). It captures the effect of both popularity and assortativity in forming links. To achieve this, we extend the assortative MMSB to capture the popularity of nodes in the network. Recent theoretical work (Papadopoulos *et al.*, 2012) has argued that optimizing the trade-offs between popularity and similarity best explains the evolution of many real networks.

There have been several research efforts to incorporate popularity into network models. Karrer et al. (Karrer and Newman, 2011) proposed the degree-corrected blockmodel that extends the classic stochastic blockmodels (Nowicki and Snijders, 2001) to incorporate node popularities. Krivitsky et al. (Krivitsky *et al.*, 2009) proposed the latent cluster random effects model that extends the latent space model (Hoff *et al.*, 2002) to include node popularities. Both models capture node similarity and popularity, but assume that unobserved similarity arises from each node participating in a single community. Finally, the Poisson community model (Ball *et al.*, 2011) is a probabilistic model of overlapping communities that implicitly captures degree-corrected mixed-memberships. However, the standard EM inference under this model drives many of the per-node community parameters to zero, which makes it ineffective for prediction or model metrics based on prediction (e.g., to select the number of communities).

In Section 6.2, we discuss the ability of the AMP to recover overlapping communities in the presence of skewed degree distributions.

Scalable approximate posterior inference

In Section 6.3, we analyze a network under the MMSB and the AMP by computing the posterior, the conditional distribution of the hidden community structure given the observed network.

The standard coordinate ascent algorithm for the MMSB (Airoldi *et al.*, 2008) iterates between computing about every pair of nodes and updating the inferred community structure. This algorithm is inefficient because it requires repeated computation about all pairs of nodes, which quickly becomes intractable for large networks. We develop efficient stochastic variational inference algorithms (see Chapter 2) for the MMSB and the AMP that iteratively subsamples the network and updates an estimate of the hidden communities. We explore several methods of subsampling.

While the MMSB is conditionally conjugate, the AMP is a nonconjugate model. We develop a scalable algorithm for posterior inference for the AMP, based on a nonconjugate variant of stochastic variational inference.

One of the main advantages of taking a probabilistic approach to network analysis is that the models and algorithms are reusable in more complex settings. Our strategy for analyzing networks easily extends to other probabilistic models, such as those taking into account node covariates. The approach we develop here opens the door to using sophisticated statistical models to analyze massive networks.

In Chapter 7, we will demonstrate the capabilities of our models on large, real networks and report on a study of large simulated networks where the community structure is known.

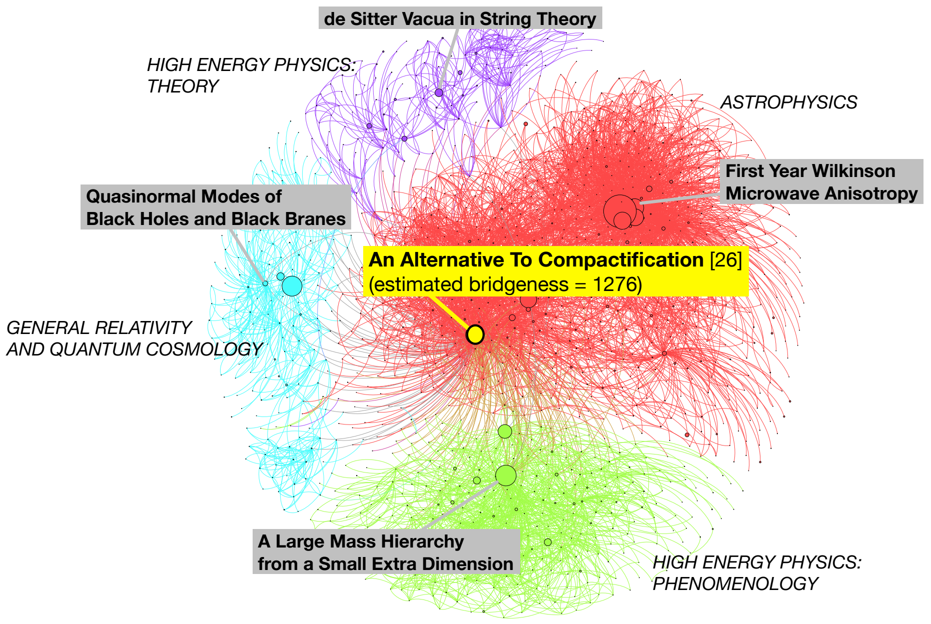

de Sitter Vacua in String Theory
HIGH ENERGY PHYSICS: THEORY
ASTROPHYSICS
First Year Wilkinson Microwave Anisotropy
Quasinormal Modes of Black Holes and Black Branes
An Alternative To Compactification [26] (estimated bridgeness = 1276)
GENERAL RELATIVITY AND QUANTUM COSMOLOGY
A Large Mass Hierarchy from a Small Extra Dimension
HIGH ENERGY PHYSICS: PHENOMENOLOGY

Figure 6.1: The discovered community structure in a subgraph of the arXiv citation network (Ginsparg, 2011). The figure shows the top four link communities that include citations to "An Alternative to Compactification" (Randall and Sundrum, 1999), an article that bridges several communities. We visualize the links between the articles, and show some highly-cited titles. Each community is labeled with its dominant subject area; nodes are sized by their bridgeness (Nepusz *et al.*, 2008), an inferred measure of their impact on multiple communities. This is taken from an analysis of the full 575,000 node network.

6.1 Finite network models

6.1.1 Modeling assortativity

In this section, we describe the assortative mixed-membership stochastic blockmodel (MMSB). The model assumes there are K communities and that each node a is associated with a vector of *community memberships* π_a. This vector is a distribution over the communities—it is positive and sums to one. For example, consider a social network and a member for whom half of her friends are from work and the other half are from her neighborhood. For this node, π_a would place half its mass on the work community and the other half on the neighborhood community.

To generate a network, the model considers each pair of nodes. For each pair $\{a, b\}$, it chooses a *community indicator* $z_{a\to b}$ from the ath node's community memberships π_a and then chooses a community indicator $z_{b\leftarrow a}$ from π_b. (Each indicator points to one of the K communities that its corresponding node is a member of.) If these indicators point to the same community then it connects nodes a and b with high probability; otherwise they are likely to be unconnected.

These assumptions capture that the connections between nodes can be explained by their memberships in multiple communities, even if we do not know where those communities lie. To see this we consider a single pair of nodes (a, b) and compute the probability that the model connects them, conditional on their community memberships. This computation requires that we marginalize out the value of the latent indicators $z_{a\rightarrow b}$ and $z_{a\leftarrow b}$.

Let β_k be the probability that two nodes are connected given that their community indicators are both equal to k. For now, assume that if the indicators point to different communities then the two nodes have zero probability of being connected. (In the full model, they will also have a small probability of being connected when the indicators are different, but this simplified version gives the intuition.) The conditional probability of a connection is

$$p(y_{ab} = 1 \,|\, \pi_a, \pi_b) = \textstyle\sum_{k=1}^{K} \pi_{ak}\pi_{bk}\beta_k. \tag{6.1}$$

The first two terms represent the probability that both nodes draw an indicator for the kth community from their memberships; the last term represents the conditional probability that they are connected given that they both drew that indicator. (The parameter β_k relates to how densely connected the kth community is.) The probability that they are connected will be high when π_a and π_b share high weight for at least one community, such as if the social network members attended the same school; it will be low if there is no overlap in their communities. The summation marginalizes out the communities, capturing that the model is indifferent to which communities the nodes have in common. The model captures *assortativity*—nodes with similar memberships will more likely link to each other (Newman, 2002; Hoff *et al.*, 2002).

We described the probability that governs a single connection between a pair of nodes. For the full network, the model assumes the following generative process:

1. For each community k, draw community strength $\beta_k \sim \text{Beta}(\eta)$.
2. For each node a, draw community memberships $\pi_a \sim \text{Dirichlet}(\alpha)$.
3. For each pair of nodes a and b, where $a < b$:
   (a) Draw community indicator $z_{a\rightarrow b} \sim \pi_a$
   (b) Draw community indicator $z_{a\leftarrow b} \sim \pi_b$
   (c) Draw the connection between them from

$$p(y_{ab} = 1 \,|\, z_{a\rightarrow b}, z_{a\leftarrow b}) = \begin{cases} \beta_{z_{a\rightarrow b}} & \text{if } z_{a\rightarrow b} = z_{a\leftarrow b} \\ \epsilon & \text{if } z_{a\rightarrow b} \neq z_{a\leftarrow b}. \end{cases}$$

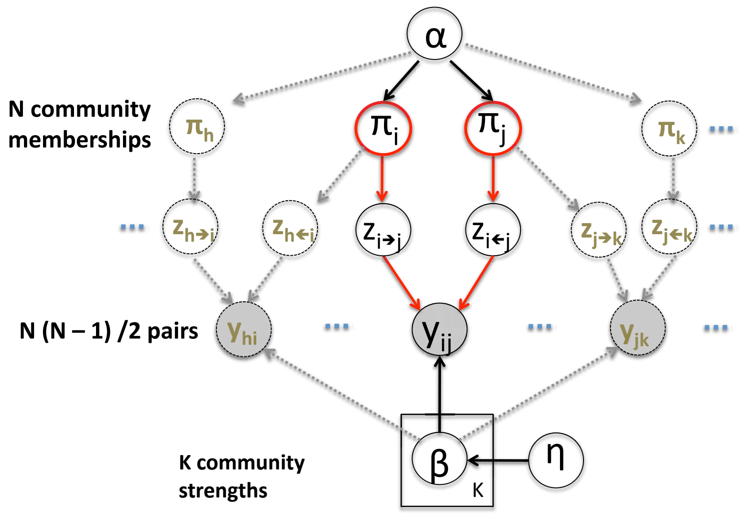


Figure 6.2: The assortative mixed-membership stochastic blockmodel (MMSB).

This defines a joint probability distribution over the N per-node community memberships $\boldsymbol{\pi}$, the per-pair community indicators $\boldsymbol{z}$, and the observed network $\boldsymbol{y}$ (both links and non-links).[1] The graphical model for the assortative MMSB is shown in Figure 6.2.

Given an observed network, the model defines a posterior distribution—the conditional distribution of the hidden community structure—that gives a decomposition of the nodes into K overlapping communities. The posterior distribution is

$$p(\boldsymbol{\pi}, \boldsymbol{z} \,|\, \boldsymbol{y}) = p(\boldsymbol{\pi}, \boldsymbol{z}, \boldsymbol{y})/p(\boldsymbol{y}). \tag{6.2}$$

In particular, the posterior will place higher probability on configurations of the community memberships that describe densely connected communities. With this posterior, we can investigate the posterior expectation of each node's memberships; and for each connected node pair, we can examine the posterior expectation of the community assignments to hypothesize why they are connected. (In this sense, our algorithm discovers link communities (Ahn *et al.*, 2010)).

As for many interesting Bayesian models, however, this posterior is intractable to compute (see Chapter 2). Further, existing approximation methods like MCMC (Robert and Casella, 2004) or variational inference (Wainwright and Jordan, 2008) are inefficient for real-world sized networks because they must iteratively consider all pairs of nodes (Nowicki and Snijders, 2001; Airoldi *et al.*, 2008). In Section 6.3, we develop an efficient stochastic variational inference algorithm for approx-

[1]We will use this as a model of an undirected graph, but it easily generalizes to the directed case.

imating the posterior with massive networks. Our algorithm will iterate between subsampling the network, analyzing the subsample, and updating the estimated community structure.

Finally, note that the model in Ball *et al.* (2011) is an interesting example of a probabilistic model for finding overlapping communities which, unlike others, accommodates an estimation algorithm that only involves the observed links between nodes. This approach is more efficient than other probabilistic models, especially on sparse networks.

Overlapping communities in the arXiv network

Our visualization in Figure 7.3 illustrates the MMSB posterior superimposed on a portion of the observed network. This is a subgraph of a network of 575,000 scientific articles on the arXiv preprint server (Ginsparg, 2011); each link denotes that an article cites or is cited by another article. Our algorithm analyzed the complete graph, discovering overlapping communities among the citations. It assigned multiple communities to each article and a single community to each link; we have colored the links accordingly. Many articles mostly link to other articles within their main community. However, the article "An Alternative to Compactification" (Randall and Sundrum, 1999) is different—it links to multiple communities, which suggests that it relates to multiple fields. Identifying nodes in large networks that bridge multiple communities is one way that our algorithm gives new insights into the structure of the network.

6.1.2 Modeling assortativity and preferential attachment

The MMSB model of Section 6.1.1 treats the links or non-links y_{ab} of a network as arising from interactions between nodes a and b. The probability that two nodes are linked is governed by the similarity of their community memberships and the strength of their shared communities. In the AMP, we capture the effect of both node popularity and assortativity in forming links.

In the AMP, we introduce latent variables θ_a to capture the popularity of each node a, i.e., its propensity to attract links independent of its community memberships. Under the AMP, the similarities between the nodes' community memberships and their respective popularities compete to explain the observations. The random effects on link generation is captured using a logit model

$$\text{logit}\ (p(y_{ab} = 1 | z_{a\to b}, z_{a\leftarrow b}, \boldsymbol{\theta}, \boldsymbol{\beta})) \equiv \theta_a + \theta_b + \textstyle\sum_{k=1}^{K} \delta_{ab}^k \beta_k, \tag{6.3}$$

where we define indicators $\delta_{ab}^k = z_{a\to b,k} z_{a\leftarrow b,k}$. The indicator δ_{ab}^k is one if both nodes assume the same community k.

Equation 6.3 is a log-linear model (McCullagh and Nelder, 1989). In log-linear models, the random component, i.e., the expected probability of a link, has a multiplicative dependency on the systematic components, i.e., the covariates. This model is also similar in the spirit of the random effects model (Hoff *et al.*, 2002)—the node-specific effect θ_a captures the popularity of individual nodes while the $\sum_{k=1}^{K} \delta_{ab}^{k} \beta_k$ term captures the interactions through latent communities. We can extend the predictor in Equation 6.3 to include observed node covariates, if any. We now define a hierarchical generative process for the observed link or non-link under the AMP:

1. Draw K community strengths $\beta_k \sim \mathcal{N}(\mu_0, \sigma_0^2)$.
2. For each node a:
    (a) Draw community memberships $\pi_a \sim \text{Dirichlet}(\alpha)$
    (b) Draw popularity $\theta_a \sim \mathcal{N}(0, \sigma_1^2)$.
3. For each pair of nodes a and b:
    (a) Draw interaction indicator $z_{a \to b} \sim \pi_a$
    (b) Draw interaction indicator $z_{a \leftarrow b} \sim \pi_b$
    (c) Draw the probability of a link $y_{ab} | z_{a \to b}, z_{a \leftarrow b}, \theta, \beta \sim \text{logit}^{-1}(z_{a \to b}, z_{a \leftarrow b}, \theta, \beta)$.

To simplify assumptions we can replace the vector of K latent community strengths $\boldsymbol{\beta}$ with a single community strength β. In Chapter 7, we demonstrate that this simpler model gives good predictive performance on small networks.

We analyze data with the AMP via the posterior distribution over the latent variables $p(\pi_{1:N}, \theta_{1:N}, \boldsymbol{z}, \beta_{1:K} | \boldsymbol{y}, \alpha, \mu_0, \sigma_0^2, \sigma_1^2)$, where $\theta_{1:N}$ represents the node popularities, and the posterior over $\pi_{1:N}$ represents the community memberships of the nodes. With an estimate of this latent structure, we can characterize the network in many useful ways.

Figure 6.3 gives an example. This is a subgraph of the netscience collaboration network (Newman, 2006) with $N = 1460$ nodes. We analyzed this network with $K = 100$ communities, using the stochastic inference algorithm for the AMP from Section 6.4. This results in posterior estimates of the community memberships and popularities for each node and posterior estimates of the community assignments for each link. With these estimates, we visualized the discovered community structure and the popular authors.

In general, with an estimate of this latent structure, we can study individual links, characterizing the extent to which they occur due to similarity between nodes and the extent to which they are an artifact of the popularity of the nodes.

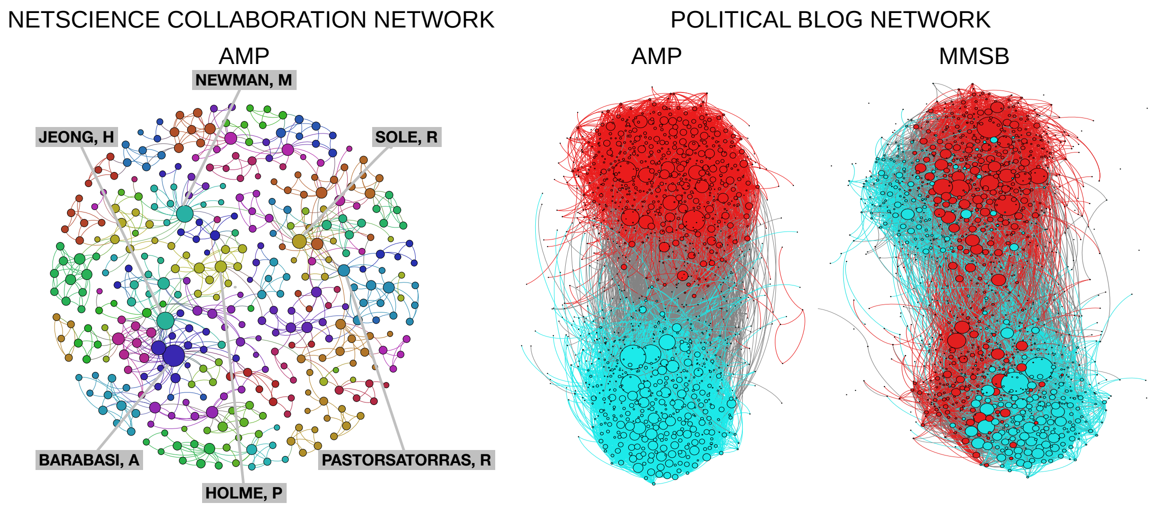


Figure 6.3: We visualize the discovered community structure and node popularities in a giant component of the netscience collaboration network (Newman, 2006) (Left). Each link denotes a collaboration between two authors, colored by the posterior estimate of its community assignment. Each author node is sized by its estimated posterior popularity and colored by its dominant research community. The network is visualized using the Fructerman-Reingold algorithm (Fruchterman and Reingold, 1991). Following Karrer and Newman (2011), we show an example where incorporating node popularities helps in accurately identifying communities (Right). The division of the political blog network (Adamic and Glance, 2005) discovered by the AMP corresponds closely to the liberal and conservative blogs identified in Adamic and Glance (2005); the MMSB has difficulty in delineating these groups.

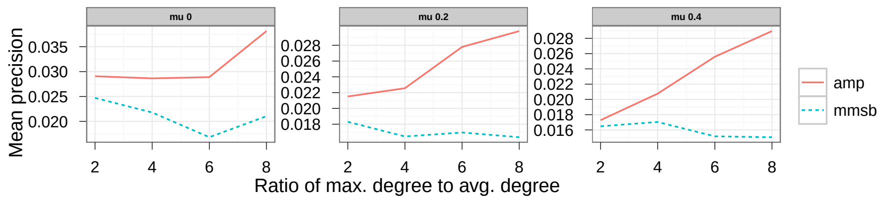


Figure 6.4: The AMP predicts significantly better than the MMSB on 12 LFR benchmark networks (Lancichinetti and Fortunato, 2009). Each plot shows 4 networks with increasing right-skewness in degree distribution. μ is the fraction of noisy links between dissimilar nodes—nodes that share no communities. The precision is computed at 50 recommendations for each node, and is averaged over all nodes in the network.

6.2 Statistical properties

In this section, we highlight the ability of the AMP to recover overlapping communities in the presence of skewed degree distributions.

Since the AMP is a "degree-corrected" mixed-membership stochastic blockmodel (Karrer and Newman, 2011), the model can better fit networks with skewed degree distributions. The skewness in degree distributions causes the community strength parameters of MMSB to overestimate or underestimate the linking patterns within communities. The per-node popularities in the AMP can capture the heterogeneity in node degrees, while learning the corrected community strengths. We demonstrate this using synthetic networks. We generated 12 LFR benchmark networks (Lancichinetti and Fortunato, 2009), each with 1000 nodes. Roughly 50% of the nodes were assigned to 4 overlapping communities, and the other 50% were assigned to single communities. We set a community size range of [200, 500] and a mean node degree of 10 with power-law exponent set to 2.0. Figure 6.4 shows that the MMSB performs poorly as the skewness is increased, while the AMP performs significantly better in the presence of both noisy links and right-skewness, both characteristics of real networks.

By accounting for node popularity, the AMP model can recover the underlying communities more accurately. Figure 6.3 shows that the AMP model delineates the division of the political blog network (Adamic and Glance, 2005) better than the MMSB.

6.3 Stochastic variational inference for the conjugate MMSB

In this section, we present the SVI algorithm for the MMSB, and describe subsampling strategies. The MMSB model of Section 6.1.1 is a conjugate model, and the terms in its variational objective are tractable. In contrast, the AMP model is a nonconjugate model (see Chapter 2); we will develop a novel nonconjugate variant of SVI algorithm for the AMP in the next section.

6.3.1 Overview

The variational objective for the MMSB contains a sum of terms for each pair of nodes (see Equation 6.6). The stochastic algorithm that we develop iteratively subsamples a subset of pairs and updates its current estimate of the community structure by following a scaled gradient computed only on that subset. (By "community structure" we mean the $N \times K$ parameters $\boldsymbol{\gamma}$, which describe the posterior distribution of each node's community memberships π.)

What is important about our algorithm is that it does not require analyzing all N^2 pairs at each iteration and that it is a valid stochastic optimization algorithm of the variational objective. It scales to massive networks.

We emphasize that our algorithm does not prune the network to make computation manageable (Chen and Redner, 2010). Rather, it repeatedly subsamples different subgraphs at each iteration. Further, our algorithm does not need to have the entire network available in order to make progress on estimating its communities. It gives a natural and principled approach for interleaving data collection on a network and estimation of its hidden community structure.

We now derive and present two algorithms:

- **Algorithm A** subsamples both links and non-links of a network, providing an algorithm that is flexible in terms of how we sample the subset of node pairs. We can analyze all the pairs associated with a sampled node; or we can use a subsampling technique that makes data collection easier, for example if the network is stored in a distributed way. Our approach here can be applied to sophisticated statistical models, and opens the door to analyzing massive networks with them.

- **Algorithm B** subsamples only the links of a network; therefore, it's faster and efficient than algorithm A. This algorithm is specific to the assortative MMSB; it exploits the assumption that when nodes assume different communities during an interaction there is a negligible chance of a link forming between them.

Naively applied, biased sampling strategies will lead to biases in posterior inference. In the following, we show how to correct for these biases and how using these strategies can lead to improved convergence speeds.

6.3.2 Algorithm A: subsample both links and non-links

Recall from Chapter 2 that in variational inference we define a family of distributions over the hidden variables $q(\boldsymbol{\beta}, \boldsymbol{\pi}, \boldsymbol{z})$ and find the member of that family that is closest to the true posterior. The mean-field variational family for the MMSB, with Beta priors placed over $\boldsymbol{\beta}$, is

$$q(\boldsymbol{\beta}, \boldsymbol{\pi}, \boldsymbol{z}) = \textstyle\prod_{k=1}^{K} q(\beta_k \,|\, \lambda_k) \prod_{n=1}^{N} q(\pi_n \,|\, \gamma_n) \prod_{i<j} q(z_{i\to j} \,|\, \phi_{i\to j})\, q(z_{i\leftarrow j} \,|\, \phi_{i\leftarrow j})\,. \qquad (6.4)$$

The variational distributions are

$$q(z_{i\to j} = k) = \phi_{i\leftarrow j,k}; \quad q(\pi_n) = \text{Dirichlet}(\pi_n; \gamma_n); \quad q(\beta_k) = \text{Beta}(\beta_k; \lambda_k). \qquad (6.5)$$

Recall from Chapter 2 that minimizing the KL divergence between q and the true posterior is equivalent to optimizing an *evidence lower bound* (ELBO) $\mathcal{L}$, a bound on the log likelihood of the observations. The ELBO is

$$
\begin{aligned}
\mathcal{L} = & \textstyle\sum_k \mathrm{E}\left[\log p(\beta_k|\eta_k)\right] - \textstyle\sum_k \mathrm{E}\left[\log q(\beta_k|\lambda_k)\right] \\
& + \textstyle\sum_n \mathrm{E}\left[\log p(\pi_n|\alpha)\right] - \textstyle\sum_n \mathrm{E}\left[\log q(\pi_n|\gamma_n)\right] \\
& + \textstyle\sum_{a,b} \mathrm{E}\left[\log p(z_{a\rightarrow b}|\pi_a)\right] + \mathrm{E}\left[\log p(z_{a\leftarrow b}|\pi_b)\right] \\
& - \textstyle\sum_{a,b} \mathrm{E}\left[\log q(z_{a\rightarrow b}|\phi_{a\rightarrow b})\right] - \mathrm{E}\left[\log q(z_{a\leftarrow b}|\phi_{a\leftarrow b})\right] \\
& + \textstyle\sum_{a,b} \mathrm{E}\left[\log p(y_{ab}|z_{a\rightarrow b}, z_{a\leftarrow b}, \boldsymbol{\beta})\right]. \qquad (6.6)
\end{aligned}
$$

The first line in Eq. 6.6 contains summations over the *global* terms: communities and nodes. They relate to the global variables, which are the community strengths $\boldsymbol{\lambda}$ and per-node memberships $\boldsymbol{\gamma}$. The remaining lines contain summations over all node pairs—the *local* terms. They depend on both the global and local variables, the latter being the indicator parameters $\boldsymbol{\phi}$.

In stochastic variational inference, we form noisy gradients by subsampling the network. This leads to a scalable algorithm because it avoids the expensive all-pairs sums in the ELBO. Existing SVI methods require the data be sampled uniformly to form noisy gradients (Hoffman *et al.*, 2013). We now develop an SVI algorithm that allows for non-uniform samples of links and non-links at each iteration. We then present SVI with link subsampling, an algorithm that subsamples only the links of a network.

Recall from Chapter 2 that stochastic variational inference iteratively updates the local and global parameters. At each iteration, it first subsamples the network. It then computes the optimal local parameters of the subset—the $(\phi_{i\rightarrow j}, \phi_{i\leftarrow j})$ for each sampled node pair (i, j)—given the current settings of the global parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\lambda}$. Finally, it updates the global parameters using a noisy natural gradient computed from the subsampled data and the optimized local parameters. The first phase is the local step; the second phase is the global step (Hoffman *et al.*, 2013).

The pseudocode of algorithm A is in Figure 6.5. The subsampling of the network in each iteration provides a way to plug in a variety of network sampling algorithms into the estimation procedure. However, to maintain a correct stochastic optimization algorithm of the variational objective, the subsampling method must be valid. That is, the natural gradients estimated from the subsample must be unbiased estimates of the true gradients.

```
1. Initialize $\boldsymbol{\gamma} = (\gamma_n)_{n=1}^N$, $\boldsymbol{\lambda} = (\lambda_k)_{k=1}^K$.
2. Subsample a set $\mathcal{S}$ of node pairs.
3. Local step.
    • For each pair $(i, j) \in \mathcal{S}$:
        – Compute the optimal indicator parameters $\hat{\phi}_{i\to j}$ and $\hat{\phi}_{i\leftarrow j}$.
4. Global step.
    • For each node $a$:
        – Compute the community membership natural gradients $\partial\gamma_a^t$ and update $\gamma_a$.
    • For each community $k$:
        – Compute the community strength natural gradients $\partial\lambda_k^t$ and update $\lambda_k$.
5. Repeat.
```

Figure 6.5: Algorithm A: an SVI algorithm for the MMSB that subsamples both links and non-links.

Computing stochastic gradients for algorithm A

The global step updates the global community strengths $\boldsymbol{\lambda}$ and community memberships $\boldsymbol{\gamma}$ with a stochastic gradient of the ELBO in Eq. 6.6. The ELBO contains summations over all $O(N^2)$ node pairs.

Pair-based sampling. Consider drawing a node pair (a, b) at random from a distribution $g(a, b)$ over the $M = N(N-1)/2$ node pairs. We can rewrite the ELBO as a random function of the variational parameters that includes the global terms and the local terms associated only with (a, b). The expectation of this random function is equal in objective to Eq. 6.6.

For example, the fifth term in Eq. 6.6 is rewritten as

$$\textstyle\sum_{a,b} \mathrm{E}\left[\log p(y_{ab}|z_{a\to b}, z_{a\leftarrow b}, \boldsymbol{\beta})\right] = \mathrm{E}\left[\frac{1}{g(a,b)}\mathrm{E}\left[\log p(y_{ab}|z_{a\to b}, z_{a\leftarrow b}, \boldsymbol{\beta})\right]\right]. \tag{6.7}$$

Evaluating the rewritten Equation 6.6 for a node pair sampled from g gives a noisy but unbiased estimate of the ELBO. Following Hoffman *et al.* (2013), the stochastic natural gradients computed from a sample pair (a, b) are

$$\partial\gamma_{a,k}^t = \alpha_k + \tfrac{1}{g(a,b)}\phi_{a\to b,k} - \gamma_{a,k}^{t-1} \tag{6.8}$$

$$\partial\lambda_{k,i}^t = \eta_{k,i} + \tfrac{1}{g(a,b)}\phi_{a\to b,k}\cdot\phi_{a\leftarrow b,k}\cdot y_{ab,q} - \lambda_{k,i}^{t-1}, \tag{6.9}$$

where $y_{ab,0} = y_{ab}$, and $y_{ab,1} = 1 - y_{ab}$. In practice, we sample a "mini-batch" S of pairs per update, to reduce noise.

The intuition behind the above update is the following. When a single pair (a, b) is sampled, we find a noisy natural gradient by computing the community memberships $\boldsymbol{\gamma}$ that would be optimal (given indicator parameters $\boldsymbol{\phi}$) if our entire network were a multigraph containing the interaction y_{ab} repeated $1/g(a, b)$ times.

Once the noisy gradient is computed, the global step follows it with an appropriate step-size,

$$\boldsymbol{\gamma} \leftarrow \boldsymbol{\gamma} + \rho_t \partial \boldsymbol{\gamma}^t; \quad \boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} + \rho_t \partial \boldsymbol{\lambda}^t. \tag{6.10}$$

Similarly to the other SVI algorithms that we describe in this thesis, we require that $\sum_t \rho_t^2 < \infty$ and $\sum_t \rho_t = \infty$ for convergence to a local optimum (Robbins and Monro, 1951). We set $\rho_t \triangleq (\tau_0 + t)^{-\kappa}$, where $\kappa \in (0.5, 1]$ is the learning rate and $\tau_0 \geq 0$ downweights early iterations.

Set-based sampling. Our algorithm has assumed that the subset of node pairs S are sampled independently. We can relax this assumption by defining a distribution over predefined sets of pairs. These sets can be defined using information about the pairs, such as network topology, which lets us take advantage of more sophisticated sampling strategies. For example, we can define a set for each node that contains the node's adjacent links or non-links. At each iteration, we sample one of these sets at random.

We set two constraints to ensure that set-based sampling results in unbiased gradients. First, the union of the sets s must be the total set of all node pairs, U: $U = \cup_i s_i$. Second, every pair (a, b) must occur in some constant number of sets $c \geq 1$. With these conditions satisfied, we can again rewrite Equation 6.6 as the sum over its global terms and an expectation over the local terms. Let $h(t)$ be a distribution over the sets. For example, the fifth term in Equation 6.6 can be written as

$$\begin{aligned} &\textstyle\sum_{a,b} \mathrm{E}\left[\log p(y_{ab} | z_{a\rightarrow b}, z_{a\leftarrow b}, \boldsymbol{\beta})\right] = \\ &\qquad \mathrm{E}\left[\tfrac{1}{c}\tfrac{1}{h(t)} \textstyle\sum_{(a,b)\in s_t} \mathrm{E}\left[\log p(y_{ab} | z_{a\rightarrow b}, z_{a\leftarrow b}, \boldsymbol{\beta})\right]\right]. \end{aligned} \tag{6.11}$$

Under set-based sampling, the stochastic gradients of the ELBO are

$$\partial \gamma_{a,k}^t = \alpha_k + \tfrac{1}{c}\tfrac{1}{h(t)} \textstyle\sum_{(a,b)\in s_t} \phi_{a\rightarrow b,k} - \gamma_{a,k}^{t-1} \tag{6.12}$$

$$\partial \lambda_{k,i}^t = \eta_{k,i} + \tfrac{1}{c}\tfrac{1}{h(t)} \textstyle\sum_{(a,b)\in s_t} \phi_{a\rightarrow b,k} \cdot \phi_{a\leftarrow b,k} \cdot y_{ab,q} - \lambda_{k,i}^{t-1}, \tag{6.13}$$

where $y_{ab,0} = y_{ab}$, and $y_{ab,1} = 1 - y_{ab}$. The updates are the same as in Equation 6.10. We now describe the steps of algorithm A in Figure 6.5 in detail.

A detailed description of Algorithm A

Initializing variational parameters. The algorithm of Figure 6.5 requires initial settings of the global variational parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\lambda}$. There are many ways to initialize these parameters. We set the community strength parameters $\boldsymbol{\lambda}$ from "false observations" by dividing the links and nodes equally among the communities and adding a small random offset drawn from a Gamma distribution with mean 1. We initialize the community memberships $\boldsymbol{\gamma}$ randomly.

The local step. The local step optimizes the indicators parameters $\boldsymbol{\phi}$ with respect to a subsample of the network. Recall that there is a indicator parameter for each node pair—$\phi_{a\rightarrow b}$ and $\phi_{a\leftarrow b}$—representing the posterior approximation of which communities are active in determining whether there is a link. We optimize these parameters in parallel. The update for $\phi_{a\rightarrow b}$ given $y_{a,b}$ is

$$\begin{aligned} \phi^t_{a\rightarrow b,k}|y = 0 &\propto \exp\{\mathrm{E}\left[\log \pi_{a,k}\right] + \phi^{t-1}_{a\leftarrow b,k}\mathrm{E}\left[\log(1-\beta_k)\right] + (1-\phi^{t-1}_{a\leftarrow b,k})\log(1-\epsilon)\} \\ \phi^t_{a\rightarrow b,k}|y = 1 &\propto \exp\{\mathrm{E}\left[\log \pi_{a,k}\right] + \phi^{t-1}_{a\leftarrow b,k}\mathrm{E}\left[\log \beta_k\right] + (1-\phi^{t-1}_{a\leftarrow b,k})\log \epsilon\}. \end{aligned} \tag{6.14}$$

This is natural gradient ascent with a step-size of one.

Assessing convergence. We measure convergence of Algorithm 1 by computing the link prediction accuracy on a held-out set of node pairs. In our experiments, we set aside two validation sets and a test set, each having $h\%$ of the network links and an equal number of non-links. In the experiments on real networks, we set $h = 5\%$. The links and non-links were chosen from the network uniformly at random. We use a separate validation set to choose learning parameters and study the sensitivity to the number of communities K.

A "50%-links" validation set poorly represents the severe class imbalance between links and non-links in real-world networks. For example, links form only 0.0039% of the node pairs in the arXiv network listed in Table 7.2. However, a validation set matching the network sparsity would have too few links. We can compute the validation log likelihood at network sparsity by reweighting the average link and non-link log likelihood (estimated from the 50%- links validation set) by their respective proportions in the network and adding them. We used the validation log likelihood at network sparsity for the SVI algorithm with informative set sampling. Since link sampling focusses on the links of the network, we used the validation log likelihood on the "50%-links" to determine convergence.

We stop training on a network (the *training* set) when the average change in expected validation log likelihood at network sparsity is less than 0.001%.

Under the MMSB, we approximate the predictive likelihood using point estimates of the posterior community memberships of nodes $\hat{\boldsymbol{\pi}}$ and the posterior community strengths $\hat{\boldsymbol{\beta}}$; these point estimates are computed as the mean of the variational posterior parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\lambda}$, respectively. The predictive likelihood is

$$\begin{aligned} p(y_{ab}|y^{\text{obs}}) &\approx p(y_{ab}|\hat{\pi}_a, \hat{\pi}_b, \hat{\boldsymbol{\beta}}) \qquad (6.15) \\ &= \sum_{z_{a\to b}} \sum_{z_{a\leftarrow b}} p(y_{ab}|z_{a\to b}, z_{a\leftarrow b}, \hat{\boldsymbol{\beta}}) p(z_{a\to b}|\hat{\pi}_a) p(z_{a\leftarrow b}|\hat{\pi}_b). \end{aligned}$$

We refer the reader to Gopalan and Blei (2013) for convergence results on the arXiv network and the Google network. We used perplexity at network sparsity. Perplexity is defined using the average predictive log likelihood of a held-out set of node pairs H,

$$\text{perplexity}(y^{\text{test}}) = \exp\left\{ -\frac{\sum_{a,b\in H} \log p(y_{ab}|y^{\text{obs}})}{|H|} \right\}. \qquad (6.16)$$

Perplexity is a measure of model fitness (lower numbers are better).

Hyperparameters and learning parameters. SVI requires setting hyperparameters of the model and learning rates of the algorithm. We set the node membership forgetting rate (κ) to 0.5 and the community strength forgetting rate to 0.9. We set the offset $\tau_0 = 1024$, but found that $\tau_0 = 1$ works better on some networks. We set Dirichlet hyperparameters $\alpha = \frac{1}{10\cdot K}$ or $\alpha = \frac{1}{K}$, where K is the number of communities. Note that in the link sampling algorithm, we set the learning rate to one. On real networks, the prior on the community strengths was set using a uniform assignment of links and nodes to communities. On synthetic networks, we set the prior on the community strengths to Beta$(1, 1)$.

Computational complexity. The local step of the SVI algorithm can be computed in $O(sK)$ operations, where s is the number of node pairs sampled in each iteration and K is the number of communities. Due to the assortativity assumptions in our model, the local step is not quadratic in K as is typical for the MMSB (Airoldi *et al.*, 2008). The time for the global step of the SVI algorithm is $O(NK)$ operations per iteration, where N is the number of nodes. In the SVI algorithm with link sampling, with the mini-batch set to all links, the computational complexity is $O(MK)$ operations per iteration, where M is the number of links. The SVI algorithm with link sampling does not require subsampling non-links and converges much faster than other subsampling methods.

Setting the number of communities. Probabilistic models of community detection require setting the number of communities, and in typical applications we will want to set this number based on the data. In our empirical study, we addressed this model selection problem in two ways. One was by evaluating the predictive performance of the model for varying numbers of communities. We held out a portion of the network $\mathbf{y}_{\text{held-out}}$ and calculated $p(\mathbf{y}_{\text{held-out}} \mid \mathbf{y}_{\text{observed}})$; a better model will assign higher probability to the held out set. This reflects a *predictive approach* to model selection, and has good statistical properties (Geisser and Eddy, 1979). (We note that non-probabilistic methods for detecting overlapping communities usually cannot provide a mechanism for predicting unseen pieces of the network.) A second way was to set the number of communities as part of the initialization procedure of the variational distribution. This procedure is described in Gopalan and Blei (2013).

Subsampling strategies for algorithm A

Algorithm A is flexible about how the subset of pairs is sampled, as long as the expectation of the stochastic gradient is equal to the true gradient. We can choose the distribution over pairs to sample from independently or choose the distribution over sets.

- **Random pair sampling.** The simplest method is to sample node pairs uniformly at random. This method is an instance of independent pair sampling, with $g(a, b)$ (used in Eq. 6.7) equal to $\frac{1}{N(N-1)/2}$.
- **Random node sampling.** This method focuses on local neighborhoods of the network. A set consists of all the pairs that involve one of the N nodes. At each iteration, we sample a set uniformly at random from the N sets, so $h(t) = 1/N$. Since each pair involves two nodes, each link or non-link appears in two sets and so $c = 2$. Following Equation 6.12 and Equation 6.13, we compute the stochastic gradients from a sampled node a as

$$\partial\gamma_{a,k}^{t} = \alpha_k + \tfrac{N}{2} \textstyle\sum_{(a,b)} \phi_{a\to b,k} - \gamma_{a,k}^{t-1} \tag{6.17}$$

$$\partial\lambda_{k,i}^{t} = \eta_{k,i} + \tfrac{N}{2} \textstyle\sum_{(a,b)} \phi_{a\to b,k} \cdot \phi_{a\leftarrow b,k} \cdot y_{ab,q} - \lambda_{k,i}^{t-1}, \tag{6.18}$$

  where $y_{ab,0} = y_{ab}$, and $y_{ab,1} = 1 - y_{ab}$. In practice, we sample a "mini-batch" of nodes per update, to reduce noise.

- **Informative set sampling.** The idea behind this method is to sample a set of pairs around each node with a bias towards pairs that help estimation. This is a type of set-based sampling.

  For each node, we define an "informative set" consisting of all its links and a small number of non-links. For each node, we also define m "non-informative sets" that partition its non-links. Since the number of non-links associated with each node is large, dividing them into many sets allows the computation in each iteration to be fast. At each iteration, we select a node uniformly at random and choose the informative set with high probability by flipping a ξ-biased coin. (We pick $\xi << 1$.) Otherwise, with low probability we select one of the m non-informative sets. To compute Eq. 6.11 we note that $c = 2$ and the distribution over sets is

$$h(t) \propto \begin{cases} (1-\xi)\frac{1}{N} & \text{if the set is informative} \\ \xi \frac{1}{Nm} & \text{if the set is non-informative.} \end{cases} \tag{6.19}$$

6.3.3 Algorithm B: subsample only links

Many real networks are sparse and only a small fraction of their node pairs are links. (See Table 7.2.) As the number of nodes increases, subsampling non-links becomes increasingly inefficient. Here we consider *link-based variational inference* and *link sampling*, a subsampling approach that involves only the links in the network. We develop this algorithm by assuming that a node's non-links are explained by the same communities that a node exhibits while generating links.

We specify the variational family in a particular way to focus on the links. It differs from the family in Equation 6.4 in the variational indicator parameters for the links. The new family specifies a indicator parameter for the joint distribution over the pair of node community indicators of each link. The indicator parameters for the non-links remain the same as in Equation 6.4. We then constrain the non-link indicator parameters of each node to equal the mean of the link indicator parameters of that node. In the following discussion, *links(a)* is the set of links of node a in the training set, and *links* are the set of all links in the training set. We define the sets for non-links similarly.

In particular, we use the following mean-field family in link-based variational inference,

$$\begin{aligned} q(\boldsymbol{\pi}, \boldsymbol{z}, \boldsymbol{\beta}) = & \textstyle\prod_{n=1}^{N} q\left(\pi_n \mid \gamma_n\right) \prod_{(i,j)\in \text{links}} q\left(z_{i\to j}, z_{i\leftarrow j} \mid \phi_{ij}\right) \\ & \textstyle\prod_{(i,j)\in \text{nonlinks}} q\left(z_{i\to j} \mid \phi_{i\to j}\right) q\left(z_{i\leftarrow j} \mid \phi_{i\leftarrow j}\right) \\ & \textstyle\prod_{k=1}^{K} q\left(\beta_k \mid \lambda_k\right). \end{aligned} \tag{6.20}$$

We constrain the indicator parameters of each non-link (i,l) of a node i,

$$\phi_{i\to l,k} = \frac{\sum_{(i,j)\in\text{links}(i)} \sum_{l=1}^{K} \phi_{ij}^{kl}}{d_i} = \frac{\sum_{(i,j)\in\text{links}(i)} \phi_{ij}^{kk}}{d_i} = \bar{\phi}_{i,k} \tag{6.21}$$

where d_i is the degree of node i in the training set, $d_i = \sum_j y_{ij}$.

The simplification in Equation 6.21 arises because $\sum_{k\neq l} \phi_{ij}^{kl} = 0$. When $k \neq l$, the community strength parameters are the non-diagonal entries of the blockmodel, each set to ϵ. Since $\epsilon \to 0$ by our modeling assumption of homophily, $\phi_{ij}^{kl} \propto \exp\{-\infty\}$. Notice that since $\sum_{k=1}^{K} \phi_{ij}^{kk} = 1$, $\bar{\phi}_i$ is normalized.

The ELBO in the link-based variational inference is a function of the variational parameters $(\boldsymbol{\phi}_{\text{links}}, \bar{\phi}, \boldsymbol{\gamma}, \boldsymbol{\lambda})$. The $\boldsymbol{\phi}_{\text{links}}$ are the $M \times K$ matrix of indicator parameters defined over the links, where M is the number of links in the training set. The $\bar{\phi}$ are the $N \times K$ matrix of the per-node mean indicator parameters. We can compute the local step update for ϕ_{ab}, given a link $y_{ab} = 1$, while fixing the other parameters:

$$\phi_{ab}^{kk} \propto \exp\{E_q \log \pi_{ak} + E_q \log \pi_{bk} + E_q \log \beta_k\}. \tag{6.22}$$

The full natural gradient of the ELBO with respect to the node's community memberships is

$$\begin{aligned} \partial\gamma_{a,k}^{t} &= \alpha_k + \textstyle\sum_{(a,b)\in\text{links}(a)} \phi_{ab}^{kk} + \sum_{(a,b)\in\text{nonlinks}(a)} \phi_{a\to b,k} - \gamma_{a,k}^{t-1} \\ &= \alpha_k + \textstyle\sum_{(a,b)\in\text{links}(a)} \phi_{ab}^{kk} + c_a \bar{\phi}_{a,k} - \gamma_{a,k}^{t-1}, \end{aligned} \tag{6.23}$$

where c_a is the number of non-links of node a in the training set. The full natural gradient of the ELBO with respect to the community strengths is,

$$\begin{aligned} \partial\lambda_{k,0}^{t} =& \eta_0 + \textstyle\sum_{(a,b)\in\text{links}} \phi_{ab}^{kk} + \lambda_{k,0}^{t-1} \\ \partial\lambda_{k,1}^{t} =& \eta_1 + \textstyle\sum_{(a,b)\in\text{nonlinks}} \phi_{a\to b,k}\phi_{a\leftarrow b,k} - \lambda_{k,1}^{t-1}. \end{aligned} \tag{6.24}$$

We can rewrite Equation 6.24 as a function of only the link indicator parameters,

$$\begin{aligned} &\textstyle\sum_{(a,b)\in\text{nonlinks}} \phi_{a\to b,k}\phi_{a\leftarrow b,k} \\ &= \textstyle\sum_{(a,b)\in\text{nonlinks}} \bar{\phi}_{a,k}\bar{\phi}_{b,k} \\ &= \textstyle\sum_n \bar{\phi}_{n,k} \sum_n \bar{\phi}_{n,k} - \sum_n (\bar{\phi}_{n,k})^2 - \sum_{(a,b)\in\text{links}} \bar{\phi}_{a,k}\bar{\phi}_{b,k}. \end{aligned} \tag{6.25}$$

We have expressed the full natural gradients of the community memberships and community strengths as a function of $\boldsymbol{\phi}_{\text{links}}$ and $\bar{\phi}$. We now describe a stochastic variational inference (SVI) algorithm that iterates only over the links.

Our link subsampling method extends random node sampling. The structure of the algorithm is similar to Figure 6.5, with a subsampling step, local steps and global steps. At each iteration, we sample a node uniformly at random and observe all its training links. In practice, we sample a mini-batch of nodes. In the local step, we iterate over the links and compute the optimal link indicator parameters using Equation 6.22. We then compute the mean indicator parameters of the sampled nodes using Equation 6.21.

As with the the previous sampling methods, we consider the stochastic optimization of the global community strengths $\boldsymbol{\lambda}$ and the global community memberships $\boldsymbol{\gamma}$. Previously, we obtained community membership gradients with respect to the entire community membership vector $\boldsymbol{\gamma}$ of dimension NK. While this allows for flexible sampling, it requires updates to all nodes in every iteration.

In link sampling, we optimize the community memberships of each node separately, using distinct learning rates. Further, when we sample a node we observe all links of a sampled node in the training set. We can therefore update the community memberships of the sampled node using the full natural gradients in Equation 6.23.

Since many networks are sparse, including the real datasets and the synthetic networks analyzed in the article, the link sampling algorithm scales to such networks even when the mini-batch in each iteration is the set of all links in the training set. In this case, the full natural gradients in Equation 6.23 and Equation 6.24 are used in the global step.

In our study on synthetic networks, we set the mini-batch to the entire set of links and used a learning rate of one. This leads to good convergence of the variational objective. Further, we rescaled γ during an initial phase as follows,

$$\gamma_{a,k} = \gamma_{a,k} * \frac{\sum_{(i,j)\in links} 1}{\sum_{(i,j)\in links} \phi_{ij}^{kk}}. \tag{6.26}$$

The rescaling of γ in Equation 6.26 ensures that each community makes an equal contribution to the observations. This avoids small communities with high community strengths and unused communities during the early iterations. The initial phase is run until the heldout likelihood on a heldout set of node pairs no longer improves. At this point, the inference continues without scaling until the algorithm converges. (This can be interpreted as a form of annealing, a technique that is sometimes used in variational inference.)

As we demonstrate in the empirical study, the SVI algorithm with link sampling recovers true communities with high accuracy, and scales to networks with millions of nodes.

Further subsampling and optimization can be applied to improve the efficiency of the SVI algorithm with link sampling. For instance, we can apply informative set sampling to the links. We maintain two dynamic sets of links: links whose corresponding indicator parameters have "converged", and links that have not converged. Each iteration, we sample links with a bias towards links that have not converged (see Equation 6.19).

6.4 Stochastic variational inference for the nonconjugate AMP

In this section, we present a novel stochastic nonconjugate variational inference algorithm for the AMP model of Section 6.1.2. Our goal is to compute the posterior distribution $p(\pi_{1:N}, \theta_{1:N}, \boldsymbol{z}, \beta_{1:K} | \boldsymbol{y}, \alpha, \mu_0 \sigma_0^2, \sigma_1^2)$. As with the MMSB, exact inference is intractable; we use variational inference (Jordan *et al.*, 1999).

Traditionally, variational inference is a coordinate ascent algorithm. However, the AMP presents two challenges. First, in variational inference the coordinate updates are available in closed form only when all the nodes in the graphical model satisfy conditional conjugacy. The AMP is not conditionally conjugate. To see this, note that the Gaussian priors on the popularity θ and the community strengths β are not conjugate to the conditional likelihood of the data. Second, coordinate ascent algorithms iterate over all the $O(N^2)$ node pairs making inference intractable for large networks.

We address these challenges by deriving a stochastic gradient algorithm that optimizes a tractable lower bound of the variational objective (Hoffman *et al.*, 2013). Our algorithm avoids the $O(N^2)$ computational cost per iteration by subsampling a "mini-batch" of random nodes and a subset of their interactions in each iteration.

In variational inference, we define a family of distributions over the hidden variables $q(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\pi}, \boldsymbol{z})$ and find the member of that family that is closest to the true posterior. We use the mean-field family, with the following variational distributions:

$$\begin{aligned} q(z_{a\to b} = i, z_{a\leftarrow b} = j) &= \phi_{ab}^{ij}; \quad & q(\pi_n) &= \text{Dirichlet}(\pi_n; \gamma_n); \\ q(\beta_k) &= \mathcal{N}(\beta_k; \mu_k, \sigma_\beta^2); & q(\theta_n) &= \mathcal{N}(\theta_n; \lambda_n, \sigma_\theta^2). \end{aligned} \tag{6.27}$$

The posterior over the joint distribution of link community assignments per node pair (a, b) is parameterized by the *per-interaction memberships* $\phi_{a\rightarrow b}$ [2], the community memberships by γ, the community strength distributions by μ and the popularity distributions by λ.

Minimizing the KL divergence between q and the true posterior is equivalent to optimizing an *evidence lower bound* (ELBO) $\mathcal{L}$, a bound on the log likelihood of the observations. We obtain this bound by applying Jensen's inequality (Jordan *et al.*, 1999) to the data likelihood. The ELBO is

$$\begin{aligned}\mathcal{L} = {} & \textstyle\sum_n \mathrm{E}\left[\log p(\pi_n|\alpha)\right] - \sum_n \mathrm{E}\left[\log q(\pi_n|\gamma_n)\right] \\ & + \textstyle\sum_n \mathbb{E}_q[\log p(\theta_n|\sigma_1^2)] - \sum_n \mathbb{E}_q[\log q(\theta_n|\lambda_n, \sigma_\theta^2)] \\ & + \textstyle\sum_k \mathbb{E}_q[\log p(\beta_k|\mu_0, \sigma_0^2)] - \sum_k \mathbb{E}_q[\log q(\beta_k|\mu_k, \sigma_\beta^2)] \\ & + \textstyle\sum_{a,b} \mathrm{E}\left[\log p(z_{a\rightarrow b}|\pi_a)\right] + \mathrm{E}\left[\log p(z_{a\leftarrow b}|\pi_b)\right] - \mathrm{E}\left[\log q(z_{a\rightarrow b}|\phi_{a\rightarrow b})\right] \\ & + \textstyle\sum_{a,b} \mathrm{E}\left[\log p(y_{ab}|z_{a\rightarrow b}, z_{a\leftarrow b}, \boldsymbol{\beta})\right]. \end{aligned} \tag{6.28}$$

Notice that the first three lines in Equation 6.28 contains summations over communities and nodes; we call these *global terms.* They relate to the global parameters which are $(\boldsymbol{\gamma}, \boldsymbol{\lambda}, \boldsymbol{\mu})$. The remaining lines contain summations over all node pairs; we call these *local terms.* They relate to the local parameters which are the ϕ_{ab}. The distinction between the global and local parameters is important—the updates to global parameters depends on all (or many) local parameters, while the updates to local parameters for a pair of nodes only depends on the relevant global and local parameters in that context.

Estimating the global variational parameters is a challenging computational problem. Coordinate ascent inference must consider each pair of nodes at each iteration, but even a single pass through the $O(N^2)$ node pairs can be prohibitive. Unlike the MMSB, the AMP is not conditionally conjugate. Nevertheless, by carefully manipulating the variational objective, we can develop a scalable stochastic variational inference algorithm for the AMP.

Lower bounding the variational objective To optimize the ELBO with respect to the local and global parameters we need its derivatives. The data likelihood terms in the ELBO can be written as

$$\mathbb{E}_q[\log p(y_{ab}|z_{a\rightarrow b}, z_{a\leftarrow b}, \boldsymbol{\theta}, \boldsymbol{\beta})] = y_{ab}\mathbb{E}_q[x_{ab}] - \mathbb{E}_q[\log(1+\exp(x_{ab}))], \tag{6.29}$$

[2]Following Kim *et al.* (2013), we use a structured mean-field assumption.

```
1. Initialize $\boldsymbol{\gamma} = (\gamma_n)_{n=1}^N$, $\boldsymbol{\lambda} = (\lambda_n)_{n=1}^N$, $\boldsymbol{\mu} = (\mu_k)_{k=1}^K$.
2. Subsample a mini-batch $\mathcal{S}$ of nodes. Let $\mathcal{P}$ be the set of node pairs in $\mathcal{S}$.
3. Local step.
    • For each pair $(a, b) \in \mathcal{P}$,
        – Compute the optimal indicator parameters $\phi_{a\to b}$ using Equation 6.36 and Equation 6.37.
4. Global step.
    • For each node $a \in \mathcal{S}$:
        – Compute the natural gradients in Equation 6.31 and update memberships $\gamma_a$
        – Compute the stochastic gradients in Equation 6.32 and update popularities $\lambda_a$.
    • For each community $k$:
        – Compute stochastic gradients in Equation 6.34 and update strengths $\mu_k$.
    • Set $\rho_a(t) = (\tau_0 + t_a)^{-\kappa}; t_a \leftarrow t_a + 1$, for each node $a \in \mathcal{S}$.
    • Set $\rho'(t) = (\tau_0 + t)^{-\kappa}; t \leftarrow t + 1$.
5. Repeat subsampling, local and global steps.
```

Figure 6.6: Stochastic nonconjugate variational inference for the AMP.

where we define $x_{ab} \equiv \theta_a + \theta_b + \sum_{k=1}^K \beta_k \delta_{ab}^k$. The terms in Equation 6.29 cannot be expanded analytically. To address this issue, we further lower bound $-\mathbb{E}_q[\log(1 + \exp(x_{ab}))]$ using Jensen's inequality (Jordan *et al.*, 1999),

$$\begin{aligned} -\mathbb{E}_q[\log(1 + \exp(x_{ab}))] &\geq -\log[\mathbb{E}_q(1 + \exp(x_{ab}))] \\ &= -\log[1 + \mathbb{E}_q[\exp(\theta_a + \theta_b + \textstyle\sum_{k=1}^K \beta_k \delta_{ab}^k)]] \\ &= -\log[1 + \exp(\lambda_a + \sigma_\theta^2/2)\exp(\lambda_b + \sigma_\theta^2/2) s_{ab}], \end{aligned} \tag{6.30}$$

where we define $s_{ab} \equiv \sum_{k=1}^K \phi_{ab}^{kk} \exp\{\mu_k + \sigma_\beta^2/2\} + (1 - \sum_{k=1}^K \phi_{ab}^{kk})$. In simplifying Equation 6.30, we have used that $q(\theta_n)$ is a Gaussian. Using the mean of a log-normal distribution, we have $\mathbb{E}_q[\exp(\theta_n)] = \exp(\lambda_n + \sigma_\theta^2/2)$. A similar substitution applies for the terms involving β_k in Equation 6.30.

We substitute Equation 6.30 in Equation 6.28 to obtain a tractable lower bound $\mathcal{L}'$ of the ELBO $\mathcal{L}$ in Equation 6.28. This allows us to develop a coordinate ascent algorithm that iteratively updates the local and global parameters to optimize this lower bound on the ELBO.

Computing stochastic gradients for the AMP

We optimize the ELBO with respect to the global variational parameters using stochastic gradient ascent. Stochastic gradient algorithms follow noisy estimates of the gradient with a decreasing step-size. If the expectation of the noisy gradient equals to the gradient and if the step-size decreases according to a certain schedule, then the algorithm converges to a local optimum (Robbins and Monro, 1951). Subsampling the data to form noisy gradients scales inference as we avoid the expensive all-pairs sums in Equation 6.28.

The global step updates the global community memberships $\boldsymbol{\gamma}$, the global popularity parameters $\boldsymbol{\lambda}$ and the global community strength parameters $\boldsymbol{\mu}$ with a stochastic gradient of the lower bound on the ELBO $\mathcal{L}'$.

We use natural gradients for the community memberships, but use distinct stochastic optimizations for the memberships and popularity parameters of each node and maintain a separate learning rate for each node. This restricts the per-iteration updates to nodes in the current mini-batch.

Since the variational objective is a sum of terms, we can cheaply compute a stochastic gradient by first subsampling a subset of terms and then forming an appropriately scaled gradient. We use a variant of the random node sampling method. At each iteration we sample a node uniformly at random from the N nodes in the network. (In practice we sample a "mini-batch" S of nodes per update to reduce noise (Hoffman *et al.*, 2013).) While a naive method will include all interactions of a sampled node as the observed pairs, we can leverage network sparsity for efficiency; in many real networks, only a small fraction of the node pairs are linked. Therefore, for each sampled node, we include as observations all of its links and a small uniform sample of m_0 non-links.

Let $\partial\gamma_a^t$ be the natural gradient of $\mathcal{L}'$ with respect to γ_a, and $\partial\lambda_a^t$ and $\partial\mu_k^t$ be the gradients of $\mathcal{L}'$ with respect to λ_a and μ_k, respectively. Following Airoldi *et al.* (2008), we have

$$\partial\gamma_{a,k}^t = -\gamma_{a,k}^{t-1} + \alpha_k + \textstyle\sum_{(a,b)\in\text{links(a)}} \phi_{ab}^{kk}(t) + \sum_{(a,b)\in\text{nonlinks(a)}} \phi_{ab}^{kk}(t), \tag{6.31}$$

where links(a) and nonlinks(a) correspond to the set of links and non-links of a in the training set. Notice that an unbiased estimate of the summation term over non-links in Equation 6.31 can be obtained from a subsample of the node's non-links. Therefore, the gradient of $\mathcal{L}'$ with respect to the membership parameter γ_a, computed using all of the nodes' links and a subsample of its non-links, is a noisy but unbiased estimate of the natural gradient in Equation 6.31.

The gradient of the approximate ELBO with respect to the popularity parameter λ_a is

$$\partial\lambda_a^t = -\tfrac{\lambda_a^{t-1}}{\sigma_1^2} + \textstyle\sum_{(a,b)\in \text{links(a)} \,\cup\, \text{nonlinks(a)}} (y_{ab} - r_{ab}s_{ab}), \tag{6.32}$$

where we define r_{ab} as

$$r_{ab} \equiv \tfrac{\exp\{\lambda_a+\sigma_\theta^2/2\}\exp\{\lambda_b+\sigma_\theta^2/2\}}{1+\exp\{\lambda_a+\sigma_\theta^2/2\}\exp\{\lambda_b+\sigma_\theta^2/2\}s_{ab}}. \tag{6.33}$$

Finally, the stochastic gradient of $\mathcal{L}'$ with respect to the global community strength parameter μ_k is

$$\partial\mu_k^t = \tfrac{\mu_0-\mu_k^{t-1}}{\sigma_0^2} + \tfrac{N}{2|S|}\textstyle\sum_{(a,b)\in \text{links(S)} \,\cup\, \text{nonlinks(S)}} \phi_{ab}^{kk}(y_{ab} - r_{ab}\exp\{\mu_k + \sigma_\beta^2/2\}). \tag{6.34}$$

As with the community membership gradients, notice that an unbiased estimate of the summation term over non-links in Equation 6.32 and Equation 6.34 can be obtained from a subsample of the node's non-links. To obtain an unbiased estimate of the true gradient with respect to μ_k, the summation over a node's links and non-links must be scaled by the inverse probability of subsampling that node in Equation 6.34. Since each pair is shared between two nodes, and we use a mini-batch with S nodes, the summations over the node pairs are scaled by $\frac{N}{2|S|}$ in Equation 6.34.

We can interpret the gradients in Equation 6.32 and Equation 6.34 by studying the terms involving r_{ab} in Equation 6.32 and Equation 6.34. In Equation 6.32, $(y_{ab} - r_{ab}s_{ab})$ is the residual for the pair (a, b), while in Equation 6.34, $(y_{ab} - r_{ab}\exp\{\mu_k + \sigma_\beta^2/2\})$ is the residual for the pair (a, b) conditional on the latent community assignment for both nodes a and b being set to k. Further, notice that the updates for the global parameters of node a and b, and the updates for $\boldsymbol{\mu}$ depend only on the diagonal entries of the indicator variational matrix ϕ_{ab}.

We can similarly obtain stochastic gradients for the variational variances σ_β and σ_θ; however, in our experiments we found that fixing them gave good results (see Chapter 7).

A detailed description of SVI for the AMP

The global step. The global step for the global parameters follows the noisy gradient with an appropriate step-size:

$$\gamma_a \leftarrow \gamma_a + \rho_a(t)\partial\gamma_a^t; \quad \lambda_a \leftarrow \lambda_a + \rho_a(t)\partial\lambda_a^t; \quad \mu \leftarrow \mu + \rho'(t)\partial\mu^t. \tag{6.35}$$

We maintain separate learning rates ρ_a for each node a, and only update the γ and λ for the nodes in the mini-batch in each iteration. There is a global learning rate ρ' for the community strength parameters $\boldsymbol{\mu}$, which are updated in every iteration. For each of these learning rates ρ, we require that $\sum_t \rho(t)^2 < \infty$ and $\sum_t \rho(t) = \infty$ for convergence to a local optimum (Robbins and Monro, 1951). We set $\rho(t) \triangleq (\tau_0 + t)^{-\kappa}$, where $\kappa \in (0.5, 1]$ is the learning rate and $\tau_0 \geq 0$ downweights early iterations.

The local step The local step optimizes the indicator parameters $\boldsymbol{\phi}$ with respect to a subsample of the network. There is a indicator parameter of dimension $K \times K$ for each node pair—$\phi_{a \rightarrow b}$—representing the posterior approximation of which pair of communities are active in determining the link or non-link. The coordinate ascent update for $\phi_{a \rightarrow b}$ is

$$\phi_{ab}^{kk} \propto \exp\Big\{\mathbb{E}_q[\log \pi_{a,k}] + \mathbb{E}_q[\log \pi_{b,k}] + y_{ab}\mu_k - r_{ab}(\exp\{\mu_k + \sigma_\beta^2/2\} - 1)\Big\} \tag{6.36}$$

$$\phi_{ab}^{ij} \propto \exp\Big\{\mathbb{E}_q[\log \pi_{a,i}] + \mathbb{E}_q[\log \pi_{b,j}]\Big\}, i \neq j, \tag{6.37}$$

where r_{ab} is defined in Equation 6.33. We present the full stochastic variational inference in Figure 6.6.

Initialization and convergence for the AMP We initialize the community memberships $\boldsymbol{\gamma}$ using approximate posterior memberships from the link-sampling based SVI algorithm for the MMSB. We initialized popularities $\boldsymbol{\lambda}$ to the logarithm of the normalized node degrees added to a small random offset, and initialized the strengths $\boldsymbol{\mu}$ to zero. We assess convergence using the procedure for the MMSB outlined in Section 6.3.

6.5 Discussion

In this chapter, we presented network models that capture basic ways in which nodes form links—nodes connecting to similar nodes, and nodes connecting to popular nodes. We presented the assortative MMSB, and developed a scalable SVI algorithm. We presented both set-based and pair-based subsampling methods. The link-sampling method, which subsamples only the links of the network, is the most efficient: the corresponding SVI algorithm needs to only iterate over the links. We overcome the nonconjugacy in the AMP models, and again developed a SVI algorithm. In all of our algorithms, we interleave subsampling the network with re-estimating its community structure.

These models have several natural extensions. First, the MMSB and the AMP models are based on the assumption that nodes assume a single latent community during interactions. Subsequently, each node is associated with normalized mixed-memberships. However, in social networks an interaction may be made stronger by multiple shared similarities between two people. An interesting venue for future work is to explore models that can aggregate the effect of multiple shared communities between nodes in explaining links between them.

Second, we restricted our models to undirected networks, but it is straightforward to consider directed networks. In this case, the models have distinct receiver and sender memberships in link formation. Finally, we can use Bayesian nonparametric assumptions to infer the number of communities within the analysis (Kim *et al.*, 2012). In general, with the ideas presented here, we can use sophisticated statistical models to analyze massive real-world networks.

In the next chapter, we evaluate the algorithms presented here on large real-world networks and on synthetic networks where ground truth communities are known.

Chapter 7

Scalable overlapping community detection

In this chapter, we study the stochastic variational inference (SVI) algorithms for the MMSB and the AMP models developed in Chapter 6 for their ability to recover community structure on large real-world and synthetic networks.

In Section 7.1 we demonstrate that the MMSB accurately identifies overlapping communities on synthetic data with millions of nodes and tens of millions of links. In Section 7.2 we discover overlapping communities on three large real-world networks and identify nodes that bridge these communities.

Finally, in Section 7.3, using nine real world networks, we compare the AMP and the MMSB models. We demonstrate that by incorporating preferential attachment in addition to assortativity, the AMP predicts better than the MMSB and provides a better fit to networks. However, the improved performance comes at a cost. The inference for the nonconjugate AMP model scales to tens of thousands of nodes, while the inference for the conjugate MMSB model scales to millions of nodes. We end with a discussion in Section 7.4.

7.1 Recovering ground truth communities

In this section, we perform a benchmark comparison of the results from the MMSB algorithm of Figure 6.5 on synthetic networks where the overlapping communities are known. We used the "benchmark" tool (Lancichinetti and Fortunato, 2009) to synthesize networks with the number of

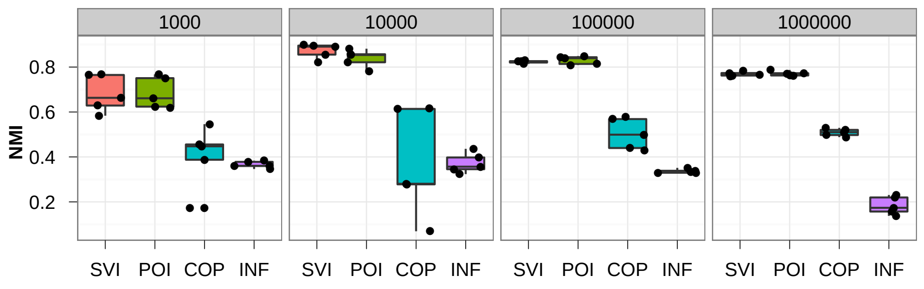


Figure 7.1: The performance of scalable algorithms on synthetic networks with overlapping communities. The numbers of nodes in each network span one thousand to one million, and for each network size we generated 5 networks. Our stochastic inference algorithm (SVI) outperforms scalable alternatives, the INFOMAP algorithm (INF) and the COPRA algorithm (COP), while performing as well as the Poisson algorithm (POI). We measure accuracy with normalized mutual information (NMI) (Lancichinetti and Fortunato, 2009). We also compared to many other methods that could not scale up to one million nodes; see the supplementary materials for a full table of results.

nodes ranging from one thousand to one million.

We compared our algorithm to the best existing algorithms for detecting overlapping communities (Ball *et al.*, 2011; Derényi *et al.*, 2005; Ahn *et al.*, 2010; McDaid and Hurley, 2010; Gregory, 2010; Lancichinetti *et al.*, 2011; Viamontes Esquivel and Rosvall, 2011). Each algorithm analyzes the (unlabeled) network and returns both the number of communities and community assignments for each node. In our algorithm we chose the number of communities in an initialization phase of variational inference. (The details are in the supplement.) Note that the Poisson algorithm also requires setting the number of communities. We used the same number of communities derived from our initialization procedure. A better algorithm better recovers the true community structure. We measured closeness to the truth with the normalized mutual information (NMI) (Lancichinetti and Fortunato, 2009), which measures the strength of the relationship between the true and discovered labels.

Most methods could not scale to one-million node networks. The four that did were our algorithm, the Poisson algorithm (Ball *et al.*, 2011), INFOMAP (Viamontes Esquivel and Rosvall, 2011), and COPRA (Gregory, 2010). Figure 7.1 shows the NMI for these methods on twenty synthetic networks, five each of 1000 nodes, 10,000 nodes, 100,000 nodes, and 1,000,000 nodes. We now describe our experiments in detail.

Generating synthetic networks

We want the synthetic networks to match the properties of real networks—skewed community and node degree distributions, significant community overlap (Derényi *et al.*, 2005; Ahn *et al.*, 2010), and a large fraction of nodes with multiple memberships.

We ran experiments to evaluate the performance of the algorithms on benchmark networks with and without *noisy* links. Notice that the significant community overlap naturally avoids well-separated clusters. The inclusion of noisy links tests the algorithm's ability to identify overlapping communities even when a significant fraction of a node's links are to other nodes sharing no communities.

For the experiments on networks without noise, we generated 20 LFR benchmark networks (Lancichinetti and Fortunato, 2009) varying in size from 1,000 to 1,000,000 nodes. Half of the nodes in each network have memberships in $m = 4$ communities. We set the average degree of nodes as $15 \times m$, similar to the experiments in McDaid and Hurley (2010). The LFR benchmarks give the node degrees and community sizes power laws; the degree distribution and community size distribution exponents were set to the default values of 2.0 and 1.0, respectively. Research on scale-free networks (Aiello *et al.*, 2000) have assumed the maximum degree to vary as $k_{\max} \sim N^{\frac{1}{\alpha}}$, where α is the power law exponent for node degrees. We set $k_{\max} = \sqrt{N}$. We varied the minimum and maximum community sizes as $(20\frac{N}{1000}, 50\frac{N}{1000})$. Since community sizes are typically small (Leskovec *et al.*, 2009) we restricted the minimum and maximum community size to 2000 and 5000 respectively. This results in about ≈ 750 ground truth communities when $N = 1,000,000$ and ≈ 30 communities when $N = 1,000$. Finally, we set the *mixing parameter* μ (Lancichinetti and Fortunato, 2009) to 0 in our experiments on networks without noise. The mixing parameter is the fraction of a node's links that connect to nodes sharing no communities.

For the experiments on networks with noisy links, we varied μ in steps of 0.2 from 0 to 0.8. We fixed the number of nodes at $10,000$, and kept the other settings the same as the preceding experiment. We generated 25 LFR benchmark networks and included only the candidate algorithms that successfully scaled to $1,000,000$ nodes in the preceding experiment.

Competing algorithms

On each network, we ran the following algorithms:

1. The COPRA label propagation algorithm (Gregory, 2010).

2. The INFOMAP algorithm based on flow compression (Viamontes Esquivel and Rosvall, 2011).

3. The MOSES seed expansion algorithm (McDaid and Hurley, 2010).

4. The Poisson EM algorithm (Ball *et al.*, 2011).

5. The OSLOM algorithm for finding statistically significant communities (Lancichinetti *et al.*, 2011).

6. The Clique percolation algorithm (Derényi *et al.*, 2005).

7. The Link clustering algorithm (Ahn *et al.*, 2010).

We used the author's source code for all algorithms. We ran the SVI algorithm with the link sampling method. For each run, we measured the normalized mutual information (NMI) (Lancichinetti and Fortunato, 2009) between the inferred community structure and the true community structure. For the algorithms that find communities at various resolutions— Clique percolation, Link clustering and COPRA—we varied the parameters as described below, and kept the best NMI score. For the SVI and the Poisson EM algorithm, we ran the algorithms until convergence on networks with up to 100,000 nodes.

A note on assessing convergence

On the million node networks, the SVI and the Poisson EM algorithm can take a long time for convergence in likelihood, while their NMI scores have typically "converged" quickly. This is primarily due to the large number of links (approximately 54 million links) in these synthetic networks. We instrumented the author's source code for the Poisson EM algorithm and the SVI algorithm to periodically report the accuracy scores when provided with ground truth communities. We gave both algorithms a computational budget of 24 hours and recorded the NMI scores attained by them. The Poisson EM algorithm's NMI score had typically "converged" at this point, even if the likelihood did not. (We note that in other applications of EM, such as probabilistic latent semantic indexing, "early stopping" is an effective form of regularization.)

Table 7.1: Accuracy results on 20 LFR benchmark networks (Lancichinetti and Fortunato, 2009) measured using normalized mutual information (Lancichinetti and Fortunato, 2009). The networks were generated with mixing parameter set to 0. The four algorithms that scale to a million nodes are the SVI algorithm, the Poisson EM algorithm (Ball *et al.*, 2011), INFOMAP (Viamontes Esquivel and Rosvall, 2011), and COPRA (Gregory, 2010). The SVI algorithm performs better than INFOMAP and COPRA and is as accurate as the Poisson EM algorithm.

Nodes	Replication	SVI	COPRA	INFOMAP	MOSES	POISSON	OSLOM	CLIQUE	LC
1,000	1	0.58	0.55	0.38	0.47	0.62	0.44	0.93	0.15
1,000	2	0.77	0.45	0.36	0.49	0.77	0.41	0.85	0.16
1,000	3	0.66	0.46	0.36	0.53	0.66	0.49	0.96	0.17
1,000	4	0.63	0.17	0.38	0.52	0.62	0.46	0.78	0.15
1,000	5	0.76	0.39	0.35	0.55	0.75	0.48	0.85	0.20
10,000	1	0.90	0.28	0.35	0.56	0.85	0.18	0.22	0.01
10,000	2	0.90	0.28	0.32	0.55	0.88	0.16	0.13	0.01
10,000	3	0.82	0.07	0.36	0.54	0.78	0.19	0.23	0.01
10,000	4	0.86	0.61	0.44	0.54	0.82	0.17	-	0.02
10,000	5	0.89	0.62	0.40	0.56	0.86	0.17	-	0.00
100,000	1	0.82	0.57	0.34	0.35	0.85	-	-	-
100,000	2	0.83	0.44	0.33	0.33	0.81	-	-	-
100,000	3	0.81	0.50	0.35	0.34	0.81	-	-	-
100,000	4	0.82	0.43	0.33	0.35	0.84	-	-	-
100,000	5	0.83	0.58	0.33	0.35	0.84	-	-	-
1,000,000	1	0.76	0.52	0.22	-	0.76	-	-	-
1,000,000	2	0.77	0.50	0.16	-	0.76	-	-	-
1,000,000	3	0.78	0.53	0.17	-	0.77	-	-	-
1,000,000	4	0.76	0.49	0.23	-	0.79	-	-	-
1,000,000	5	0.77	0.51	0.14	-	0.77	-	-	-

Algorithm settings for the LFR experiments

The Clique percolation algorithm identifies communities from a series of adjacent k-cliques. In our experiments we varied k from 4 to 7, a typical range for LFR experiments (McDaid and Hurley, 2010; Gregory, 2010). The Link clustering algorithm defines a similarity function over nodes sharing a link, and uses hierarchical clustering to find hierarchical community structures (Ahn *et al.*, 2010). Since the dendrogram can be partitioned in multiple ways, the algorithm uses a measure of the quality of a link partition, called the partition density D. We varied D from 0.1 to 0.4—the range we found to be best—in steps of 0.1. The COPRA algorithm is a fast heuristic based on label propagation and includes a overlap parameter that we varied from 2 to 10, in steps of 2. This is a typical range (Gregory, 2010). The OSLOM (Lancichinetti *et al.*, 2011), MOSES (McDaid and Hurley, 2010) and the INFOMAP (Viamontes Esquivel and Rosvall, 2011) algorithms were run with parameters set to default values.

For the SVI and the Poisson EM algorithm (Ball *et al.*, 2011), we set K to the value chosen by the initial phase of our variational algorithm. The author's software for most of the algorithms we compare to generate *community assignments*, the discovered mapping between nodes and communities. The mapping is used to compute the accuracy when given the ground truth communities.

We extended both the SVI and the Poisson EM algorithm to generate the community assignment files. In both algorithms, we assigned a link to a community if the approximate posterior probability of link assignment to a community exceeded a threshold t. We took the best NMI values obtained from thresholds $t = 0.5$ and $t = 0.9$. For the experiments on networks without noise, we assigned each node associated with a link to the same community as the link. For the experiments with noisy links, we required at least 3 links of a node to be assigned to a community prior to assigning the node to that community. We added this setting to both algorithms to control sensitivity to noise. (Notice in Figure 7.2 that both algorithms continue to show a high accuracy on networks without noise ($\mu = 0$) with the threshold set to 3 links.)

Open source software

We implemented the SVI algorithm and the various subsampling variants in C++. Our source code is available from: github.premgopalan.com/svinet.

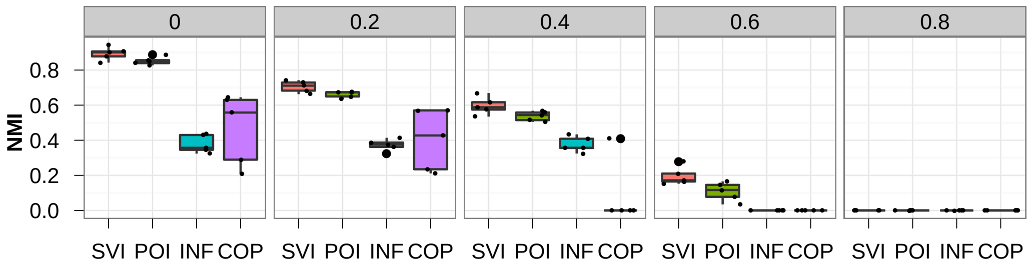


Figure 7.2: Our stochastic inference algorithm (SVI) outperforms COPRA (COP) (Gregory, 2010) and INFOMAP (INF) (Viamontes Esquivel and Rosvall, 2011) and is as accurate as the Poisson EM algorithm (POI) (Ball *et al.*, 2011) in discovering ground truth communities in 25 LFR benchmark networks with *noisy* links. Each panel shows the performance of the algorithms on five replications of the random network generated with 10,000 nodes and a fixed mixing parameter (Lancichinetti and Fortunato, 2009). The mixing parameter is the fraction of a node's links that connect to nodes sharing no communities. From left to right, the panels correspond to increasing noise.

Results

Table 7.1 shows the NMI results on the networks without noise. Most algorithms could not scale to one-million node networks. The four that did were the Poisson EM, the SVI, the COPRA and the INFOMAP algorithms. The SVI algorithm performs better than the COPRA and the INFOMAP algorithms and is as accurate as the Poisson EM algorithm on the one-million node network. On smaller networks, the SVI algorithm performs as well as the Poisson EM algorithm; it performs second to Clique percolation on the one-thousand node networks. However, the Clique percolation algorithm does not scale beyond the 10,000 node networks.

Figure 7.2 shows the NMI results on the networks with noisy links. We find that the SVI algorithm performs better than two of the three other scalable alternatives—the COPRA and the INFOMAP algorithms—and is as accurate as the Poisson EM algorithm.

7.2 Exploring real-world communities

In this section, we demonstrate how the MMSB algorithm of Figure 6.5 can be used to study massive real-world networks. We ran the SVI algorithm with informative set sampling on the real networks in Table 7.2. We pre-processed the networks to associate each node with "informative" and "non-informative" sets of node pairs.

We analyzed two citation networks: a network of 575,000 scientific articles from the arXiv pre-print server (Ginsparg, 2011) and a network of 3,700,000 patents from the United States patent network (Leskovec *et al.*, 2005). In these networks, a link indicates that one document cites another.

Table 7.2: Real-world networks analyzed in the article and the supplement

Data set	Number of nodes	Number of links	% links	Type	Source
arXiv	576,000	6,640,000	0.0039%	citation	(Ginsparg, 2011)
Google	875,000	4,320,000	0.0011%	hyperlink	(Leskovec *et al.*, 2009)
US Patents	3,700,000	16,500,000	0.00023%	citation	(Leskovec *et al.*, 2005)

We also analyzed a large network of 875,000 webpages from Google (Leskovec *et al.*, 2009).[1] In all networks, we treated the directed links as undirected—the presence of a link is evidence of similarity between the nodes and is independent of direction. (This is common in hyperlink graph analysis (Fortunato, 2010).) These networks are much larger than what can easily be analyzed with previous approaches to computing with mixed-membership stochastic blockmodels (Airoldi *et al.*, 2008). (Though we note that several efficient methods have recently been developed for block models without overlapping communities (Decelle *et al.*, 2011a,b; Amini *et al.*, 2012).)

We analyze a network by setting the number of communities K and running the stochastic inference algorithm.[2] This results in posterior estimates of the community memberships for each node and posterior estimates of the community assignments for each node pair (i.e., for each pair of nodes, estimates of which communities governed whether they are connected). With these estimates, we visualize the network according to the discovered communities.

Scientific articles from arXiv

The arXiv network (Ginsparg, 2011) contains scientific articles and citations between them. Our large subset of the arXiv contains 575,000 physics papers. We ran stochastic inference to discover 200 communities. This took a few hours of computation.[3]

Figure 7.3 illustrates a subgraph of the arXiv network and demonstrates the structure that our algorithm uncovered. In the model, each node i contains community memberships θ_i and each link (i, j) is assigned to one of the K communities. In the figure, we colored each link according to the peak of the approximate posterior $p(z_{i \to j} \mid \boldsymbol{y})$. This suggests within which communities and to what degree each paper has had an impact. (We note that most of the links attached to highly cited articles are incoming links, so visualizing these links reveals the communities influenced by the paper.)

[1]This data did not contain the descriptions of the nodes that are required to visualize the communities. Our quantitative analyses of this network is in the supplementary materials.

[2]We assess convergence and set the number of communities by looking at predictive probability on a held out subgraph. We also used the predictive probability to confirm that the mixed-membership model gives a better fit than the single-membership model (Nowicki and Snijders, 2001; Wiggins and Hofman, 2008). The details of these results are in the supplementary materials. We will release our software tool on publication.

[3]In detail, it took a few hours when using the link sampling method of subsampling pairs. See details in the supplement.

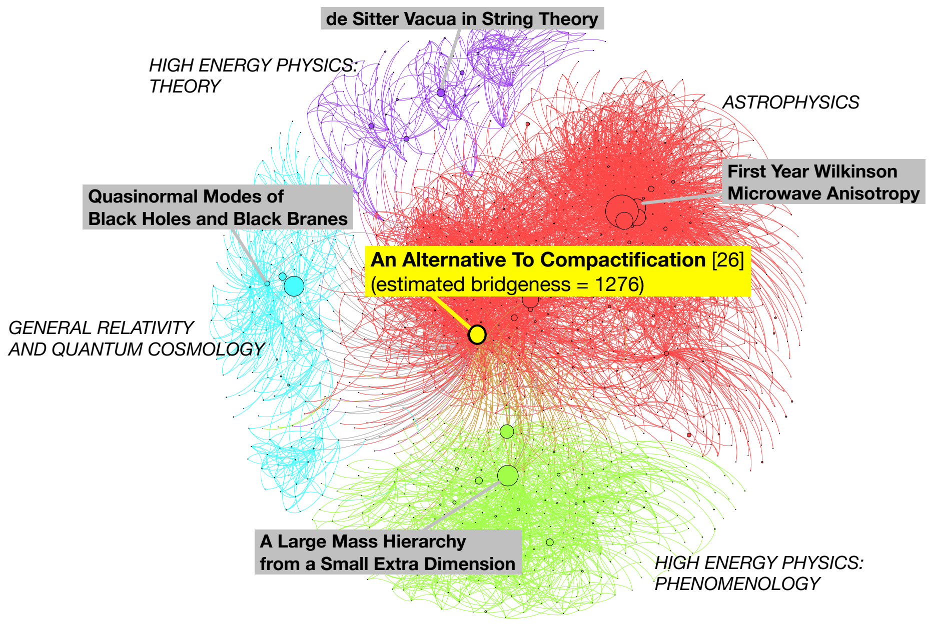

de Sitter Vacua in String Theory
HIGH ENERGY PHYSICS: THEORY
ASTROPHYSICS
First Year Wilkinson Microwave Anisotropy
Quasinormal Modes of Black Holes and Black Branes
An Alternative To Compactification [26] (estimated bridgeness = 1276)
GENERAL RELATIVITY AND QUANTUM COSMOLOGY
A Large Mass Hierarchy from a Small Extra Dimension
HIGH ENERGY PHYSICS: PHENOMENOLOGY

Figure 7.3: The discovered community structure in a subgraph of the arXiv citation network (Ginsparg, 2011). The figure shows the top four link communities that include citations to "An Alternative to Compactification" (Randall and Sundrum, 1999), an article that bridges several communities. We visualize the links between the articles, and show some highly-cited titles. Each community is labeled with its dominant subject area; nodes are sized by their bridgeness (Nepusz *et al.*, 2008), an inferred measure of their impact on multiple communities. This is taken from an analysis of the full 575,000 node network.

The central article in Figure 7.3 is the highly cited article "An Alternative to Compactification" (Randall and Sundrum, 1999), which was published in 1999. The article proposes a simple explanation to one of the most important problems in physics: Why is the weak force 10^{32} times stronger than gravity? The paper's external tag (given by the authors) suggests it is primarily a theoretical paper. It has had, however, an impact on a diverse array of problems including certain astrophysics puzzles regarding the structure of the universe (Khoury *et al.*, 2001) and the confrontation between general relativity and experiment (Will, 2006).

In analyzing the full network of citations, our algorithm has captured how this article has played a role in multiple subfields. It assigned it to membership in nine communities and gave it a high posterior bridgeness score (Nepusz *et al.*, 2008), a measure of how strongly it bridges multiple communities.[4] In the subgraph of Figure 7.3, the link colors correspond to the research communities associated with the links. We visualize the four top communities that link to this article: "High Energy Physics: Theory," "High Energy Physics: Phenomenology," "General Relativity and Quan-

[4]We compute the degree-corrected bridgeness (Nepusz *et al.*, 2008), which relates to a normalized distance between the community memberships of a node and the uniform distribution over communities. This is usually a function of known community memberships. However, since the community structure is hidden, we use the algorithm in the previous section to approximate the posterior and then compute the expected bridgeness.

Table 7.3: Top 10 articles in the arXiv network by estimated bridgeness (Nepusz *et al.*, 2008). Notice that some articles have higher bridgeness but a smaller citation count than others

Title	# Citations	Estimated bridgeness
Maps of Dust IR Emission...	5946	2893.7
First Year Wilkinson Microwave...	5707	2270.7
Wilkinson Microwave Anisotropy...	4488	1907.1
Big Bang Nucleosynthesis	2896	1882.9
The Cosmological Parameters 2006	2472	1703.9
Five-Year Wilkinson Microwave...	2804	1485.9
A Large Mass Hierarchy...	3644	1426.7
The Large N Limit of Superconformal...	3914	1378.4
An Alternative to Compactification	2803	1275.8

tum Cosmology," and "Astrophysics."[5] We emphasize that the citations alone cannot reveal the role of an article in its citation graph—we executed this analysis by first discovering the communities with our algorithm and then using those discovered communities to compute quantities, like bridgeness (Nepusz *et al.*, 2008) and link color, that require community assignments.

As an example of a different kind of article, consider "The Cosmological Constant - the Weight of the Vacuum" (Padmanabhan, 2003). This article has 1,117 citations in the dataset, on the same order as Randall and Sundrum (1999). It discusses the theoretical and cosmological aspects of the cosmological constant. Our algorithm finds that this article has a lower bridgeness, and membership in only two communities. Both communities are dominated by the "Astrophysics" subject tag, with the other significant tag being "General Relativity and Quantum Cosmology." Detecting these two kinds of articles highlights an advantage of this type of analysis. By discovering the hidden community structure, we can separate articles (of similar citation count) that have had interdisciplinary impact from those with impact within their particular fields.

We have illustrated a small subgraph of this large network, centered around a specific article. Across the whole network, we can use the posterior bridgeness to filter and find a collection of articles that have had interdisciplinary impact. Table 7.2 shows the top ten papers in the arXiv network by posterior bridgeness. The top scientific articles in Table 7.2 have a wide impact, as they concern data, parameters or theory applied in various sub-fields of physics. For example, the top article, "Maps of Dust IR Emission for Use in Estimation of Reddening and CMBR Foregrounds" (Schlegel *et al.*, 1998) constructs an accurate full sky map of the dust temperature useful in the estimation of cosmic microwave background radiation. This filtering demonstrates the practical potential for unsupervised analysis of large networks. The posterior bridgeness score, a function of the discovered communities, helps us focus on a class of nodes that is otherwise difficult to find.

[5]Naming and interpreting communities is a difficult problem in unsupervised community detection. For visual convenience, we examine the external tags given to the articles and name each community by its most common tag. Note the algorithm does not have access to the tags.

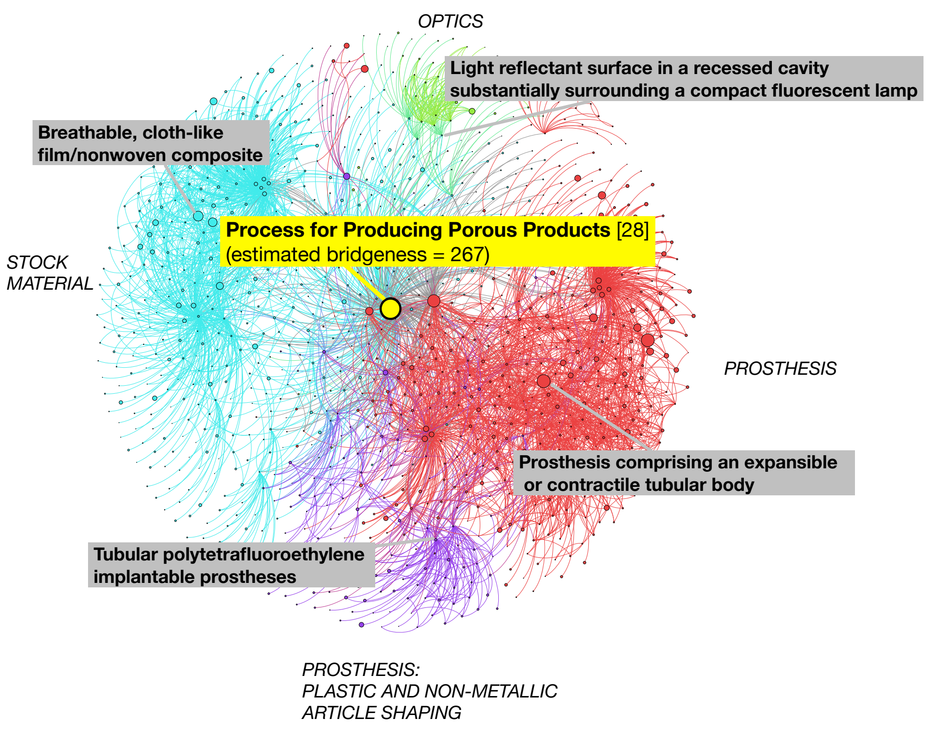

OPTICS
Light reflectant surface in a recessed cavity substantially surrounding a compact fluorescent lamp
Breathable, cloth-like film/nonwoven composite
Process for Producing Porous Products [28] (estimated bridgeness = 267)
STOCK MATERIAL
PROSTHESIS
Prosthesis comprising an expansible or contractile tubular body
Tubular polytetrafluoroethylene implantable prostheses
PROSTHESIS: PLASTIC AND NON-METALLIC ARTICLE SHAPING

Figure 7.4: The discovered community structure in a subgraph of the U.S. Patents network (Leskovec *et al.*, 2005). The figure shows subgraphs of the top four communities that include citations to "Process for producing porous products." (Gore, 1976). We visualize the links between the patents and show titles of some of the highly-cited patents. Each community is labeled with its dominant classification; nodes are sized by their bridgeness (Nepusz *et al.*, 2008); the local network is visualized using the Fruchterman Reingold algorithm (Fruchterman and Reingold, 1991). This is taken from an analysis of the full 3.7M node network.

United States patents

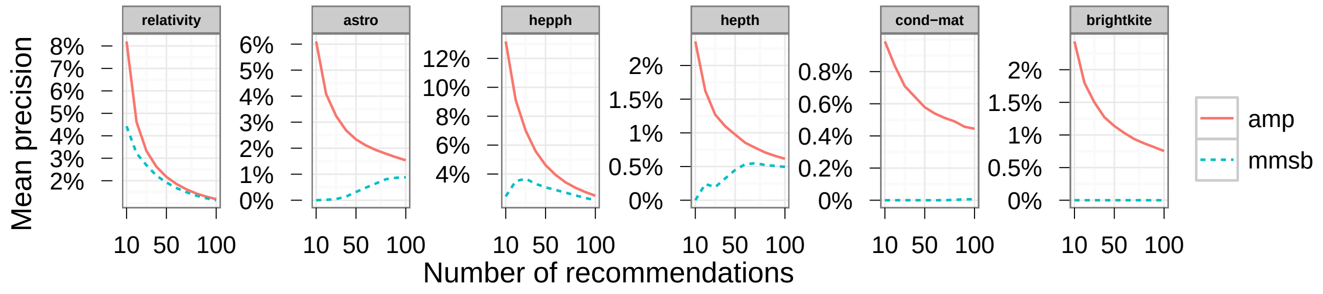
The National Bureau of Economic Research maintains a large data set of United States patents (Leskovec *et al.*, 2005). It contains 3,700,000 patents granted between 1975 and 1999 and the citations between them. We analyzed this network, setting the number of communities to 1000.

Figure 7.4 illustrates a subgraph of the patents data that reveals overlapping community structure around "Process for Producing Porous Products" (Gore, 1976). This patent was issued in 1976 and describes an efficient process for producing highly porous materials from tetrafluoroethylene polymers. It has influenced the design of many everyday materials, such as waterproof laminate, adhesives, printed circuit boards, insulated conductors, dental floss, and strings of musical instruments. Our algorithm assigned it a high posterior bridgeness and membership in 39 communities. The classification tags of the citing patents confirm that it has influenced several areas of patents: Synthetic Resins or Natural Rubbers, Prosthesis, Stock Material, Plastic and Non-Metallic Article Shaping, Adhesive bonding, Conductors and Insulators, and Web or Sheet. Figure 7.4 illustrates the top communities for this patent, found by our algorithm.

We also studied a patent with a comparable number of citations but with significantly lower bridgeness. “Osmotic dispensing device for releasing beneficial agent” (Eckenhoff *et al.*, 1987) concerns a novel osmotic dispenser for continually administering agents, e.g., opthalmic drugs. It has 339 citations, comparable to the 441 of Gore (1976), but a much lower bridgeness score. Our algorithm assigned it to 7 communities, with the classification tags mostly restricted to “Drug: Bio-Affecting and Body Treating Compositions” and “Surgery.”

Figure 7.5: Network data sets. N is the number of nodes, d is the percent of node pairs that are links and P is the mean perplexity over the links and nonlinks in the held-out test set.

DATA SET	N	$d(\%)$	P_{AMP}	P_{MMSB}	TYPE	SOURCE
US AIR	712	1.7%	**2.75 ± 0.04**	3.41 ± 0.15	TRANSPORT	(RITA, 2010)
POLITICAL BLOGS	1224	1.9%	**2.97 ± 0.03**	3.12 ± 0.01	HYPERLINK	(ADAMIC AND GLANCE, 2005)
NETSCIENCE	1450	0.2%	**2.73 ± 0.11**	3.02 ± 0.19	COLLAB.	(NEWMAN, 2006)
RELATIVITY	4158	0.1%	**3.69 ± 0.18**	6.53 ± 0.37	COLLAB.	(LESKOVEC *et al.*, 2008)
HEP-TH	8638	0.05%	**12.35 ± 0.17**	23.06 ± 0.87	COLLAB.	(LESKOVEC *et al.*, 2008)
HEP-PH	11204	0.16%	**2.75 ± 0.06**	3.310 ± 0.15	COLLAB.	(LESKOVEC *et al.*, 2008)
ASTRO-PH	17903	0.11%	**5.04 ± 0.02**	5.28 ± 0.07	COLLAB.	(LESKOVEC *et al.*, 2008)
COND-MAT	36458	0.02%	**10.82 ± 0.09**	13.52 ± 0.21	COLLAB.	(NEWMAN, 2006)
BRIGHTKITE	56739	0.01%	**10.98 ± 0.39**	41.11 ± 0.89	SOCIAL	(LESKOVEC *et al.*, 2008)



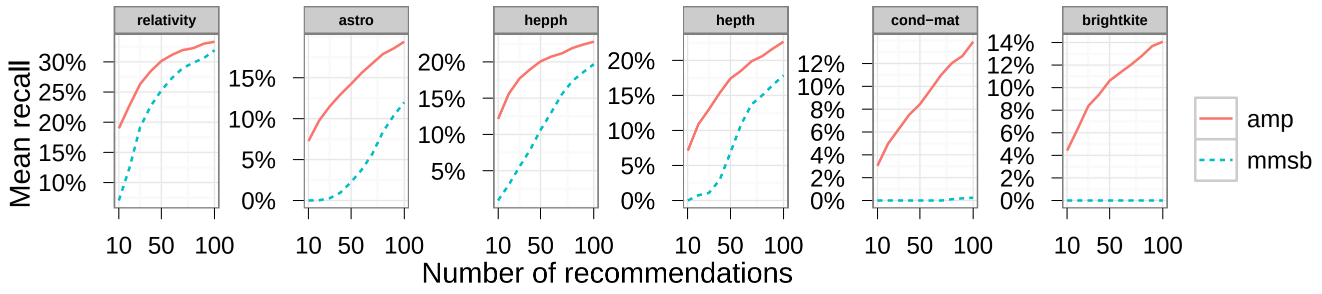


Figure 7.6: The AMP model outperforms the MMSB model in predictive accuracy on real networks. Both models were fit using stochastic variational inference (Hoffman *et al.*, 2013). For the data sets shown, the number of communities K was set to 100 and hyperparameters were set to the same values across data sets. The perplexity results are based on five replications. A single replication is shown for the mean precision and mean recall.

7.3 Comparing the network models

We use the predictive approach to evaluating model fitness (Geisser and Eddy, 1979), comparing the predictive accuracy of AMP (Section 6.4) to the MMSB with link sampling (Section 6.3.3). In all data sets, we found that AMP gave better fits to real-world networks. Our networks range in size from 712 nodes to 56,739 nodes. Some networks are sparse, having as little as 0.01% of all pairs as links, while others have up to 2% of all pairs as links. Our data sets contain four types of networks: hyperlink, transportation, collaboration and social networks. We implemented the SVI algorithm for the AMP from Figure 6.6 in 4,800 lines of C++ code. [6]

Evaluation metrics

We used perplexity, mean precision and mean recall in our experiments to evaluate the predictive accuracy of the algorithms. We computed the link prediction accuracy using a test set of node pairs that are not observed during training. The test set consists of 10% of randomly selected links and non-links from each data set. During training, these test set observations are treated as zeros. We approximate the predictive distribution of a held-out node pair y_{ab} under the AMP using posterior estimates $\hat{\theta}$, $\hat{\beta}$ and $\hat{\pi}$.

Perplexity is the exponential of the average predictive log likelihood of the held-out node pairs. For mean precision and recall, we generate the top n pairs for each node ranked by the probability of a link between them. The ranked list of pairs for each node includes nodes in the test set, as well as nodes in the training set that were non-links. We compute precision-at-m, which measures the fraction of the top m recommendations present in the test set; and we compute recall-at-m, which captures the fraction of nodes in the test set present in the top m recommendations. We vary m from 10 to 100. We then obtain the mean precision and recall across all nodes. [7]

Hyperparameters

For the stochastic AMP algorithm, we set the "mini-batch" size $S = N/100$, where N is the number of nodes in the network and we set the non-link sample size $m_0 = 100$. We set the number of communities $K = 2$ for the political blog network and $K = 20$ for the US air; for all other networks, K was set to 100. We set the hyperparameters $\sigma_0^2 = 1.0$, $\sigma_1^2 = 10.0$ and $\mu_0 = 0$, fixed the variational variances at $\sigma_\theta = 0.1$ and $\sigma_\beta = 0.5$ and set the learning parameters $\tau_0 = 65536$ and $\kappa = 0.5$. We set the Dirichlet hyperparameter $\alpha = \frac{1}{K}$ for the AMP and the MMSB.

[6] Our software is available at github.com/premgopalan/sviamp.

[7] Precision and recall are better metrics than ROC AUC on highly skewed data sets (Davis and Goadrich, 2006).

Results

Figure 7.5 compares the AMP and the MMSB stochastic algorithms on a number of real data sets. The AMP definitively outperforms the MMSB in predictive performance. All hyperparameter settings were held fixed across data sets. The first four networks are small in size, and were fit using the AMP model with a single community strength parameter. All other networks were fit with the AMP model with K community strength parameters. As N increases, the gap between the mean precision and mean recall performance of these algorithms appears to increase. Without node popularities, MMSB is dependent entirely on node memberships and community strengths to predict links. Since K is held fixed, communities are likely to have more nodes as N increases, making it increasingly difficult for the MMSB to predict links. For the small US air, political blogs and netscience data sets, we obtained similar performance for the replication shown in Figure 7.5. For the AMP the mean precision at 10 for US Air, political blogs and netscience were 0.087, 0.07, 0.092, respectively; for the MMSB the corresponding values were 0.007, 0.0, 0.063, respectively.

7.4 Discussion

In this chapter, we studied the stochastic variational inference algorithms for the AMP and the MMSB models of Chapter 6. We demonstrated that these algorithms can scale to large network sizes and recover overlapping communities in benchmark networks. The MMSB algorithm scales better than the AMP algorithm, but the AMP model outperforms the MMSB in predictive performance by capturing node popularities.

Our exploration of large overlapping communities and "bridging" nodes in real collaboration networks has presented the kinds of new scientific analyses these inferences enable. We leave as future work the exercise of incorporating node covariates in the AMP models, and fitting large network data sets where such metadata is available.

In the previous chapters, we have analyzed sparse discrete data sets of user behavior, text and networks. In the next chapter, we consider the problem of fitting probabilistic models to massive data sets from genetic variations. Unlike the previous data sets, these data sets are dense, presenting a new inferential challenge; unlike several of the algorithms we have developed so far, we cannot exploit data sparsity for scalability.

Part III

Genetic variation

Chapter 8

Scalable inference of genetic variation

The goal of population genetics is to quantitatively understand variation of genetic polymorphisms among individuals. Researchers have developed sophisticated statistical methods to capture the complex population structure that underlies observed genotypes in humans. The number of humans that have been densely genotyped across the genome has grown significantly in recent years. In aggregate about 1M individuals have been densely genotyped to date, and if we could analyze this data then we would have a nearly complete picture of human genetic variation. Existing state-of-the-art methods, however, cannot scale to data of this size.

In this chapter, we develop a new algorithm TeraStructure to fit Bayesian models of genetic variation in human populations on tera-sample-sized data sets (10^{12} observed genotypes, e.g., 1M individuals at 1M SNPs). Terastructure is a stochastic variational inference algorithm (see Section 2.2.4) for the PSD model we present in Section 8.2. It iterates between subsampling locations of the genome and updating an estimate of the latent population structure. We develop this algorithm in Section 8.3.

In Section 8.4, we demonstrate that on real and simulated data sets of up to 10K individuals, TeraStructure is twice as fast as existing methods and recovers the latent population structure with equal accuracy. On genomic data simulated at the tera-sample-size scales, TeraStructure continues to be accurate and is the only method that can complete its analysis.

8.1 Introduction

The quantitative characterization of genetic polymorphisms in human populations plays a key role in understanding evolution, migration, and trait variation. Genetic variation of humans is highly structured in that frequencies of genetic polymorphisms depend strongly on ancestry and evolutionary forces that vary among individuals. Therefore, to comprehensively understand human genetic variation, we must also understand the underlying structure of human populations.

Over the last fifteen years, scientists have successfully used genome-wide Bayesian models of genetic polymorphisms to infer the latent structure embedded in an observed population. The *PSD model* of Pritchard, Stephens and Donnelly (Pritchard *et al.*, 2000) has become a standard tool both for exploring hypotheses about human genetic variation and for taking latent structure into account in downstream analyses.

Modern genetics, however, cannot take full advantage of the PSD model and related probabilistic models. The reason is that the existing solutions to the core computational problem—the problem of estimating the latent ancestral structure given a collection of observed genetic data—cannot handle the scale of modern datasets. For example, the sample sizes of genome-wide association studies now routinely involve tens of thousands of people. Public and private initiatives have managed to measure genome-wide genetic variation on hundreds of thousands of individuals. Taken together, we now have dense genome-wide genotype data on the order of a million individuals. Fitting probabilistic models on these data would provide an unprecedented characterization of genetic variation and the structure of human populations. But, as we show in our study, this analysis is not possible with the current state of the art.

To this end, we develop TeraStructure, an algorithm for analyzing data sets of up to 10^{12} genotypes. It is an SVI algorithm (Hoffman *et al.*, 2013) which iterates between subsampling observed single nucleotide polymorphism (SNP) genotypes, analyzing the subsample, and updating its estimate of the hidden ancestral populations. TeraStructure provides a statistical estimate of the PSD model, that we review in Section 8.2. The PSD model captures the heterogenous mixtures of ancestral populations that are inherent in a data set of observed human genomes.

8.2 The PSD Model

We present our model and algorithm for unphased genotype data, though it easily generalizes to phased data. (Most massive population genetics data sets are unphased.) In unphased data, each observation $x_{i,\ell} \in \{0,1,2\}$ denotes the observed allele for individual i at location ℓ. The data are coded for how many major alleles are present: $x_{i\ell} = 0$ indicates two minor alleles; $x_{i,\ell} = 2$ indicates two major alleles; and $x_{i,\ell} = 1$ indicates one major and one minor allele. In this last case we do not code which allele came from the mother and which from the father. This is what it means for the data to be unphased.

Formally, the PSD model assumes that there are K ancestral populations, each characterized by its minor allele frequencies β_k for each of the SNPs. Further, it assumes that each individual in the sample exhibits those populations with different proportions θ_i. Finally, it assumes that each SNP genotype ℓ in each individual i is drawn from an ancestral population that itself is drawn from the individual-specific proportions. The data are assumed drawn from the following model:

$$\begin{aligned}
\beta_{k,\ell} &\sim \text{Beta}(a,b) \\
\theta_i &\sim \text{Dirichlet}(c) \\
x_{i,\ell} &\sim \text{Binomial}\left(2, \textstyle\sum_k \theta_{i,k}\beta_{k,\ell}\right).
\end{aligned}$$

This is the model for unphased data in Pritchard *et al.* (2000).

8.2.1 The PSD posterior

The PSD model turns the problem of estimating ancestral population structure into one of posterior inference, i.e., estimating a conditional distribution. The assumed genomic structure—the population proportions for each individual and the allele frequencies for each population—are hidden random variables in the model; the collection of individuals at a collection of SNPs $x = \{x_{i,\ell}\}$ are observed random variables. The main computational problem for the PSD model is to estimate the posterior distribution of the hidden population structure given the data, $p(\beta, \theta \,|\, x)$. With this posterior, or posterior means of the hidden variables, population geneticists can explore the latent structure of their data and correct for ancestry in downstream analyses.

For example, Figure 8.1 illustrates the posterior expected population proportions, computed from our algorithm, for the 1718 individuals of the 1000-Genomes data set. Figure 8.1 illustrates these posterior estimates at three values of the latent number of populations K, at $K = 7$, $K = 8$ and $K = 9$. This data set contains over 3 billion observations. Though the model is not aware of the country-of-origin for each individual, our algorithm uncovered population structure consistent with the major geographical regions. Some of the groups of individuals identify a specific region

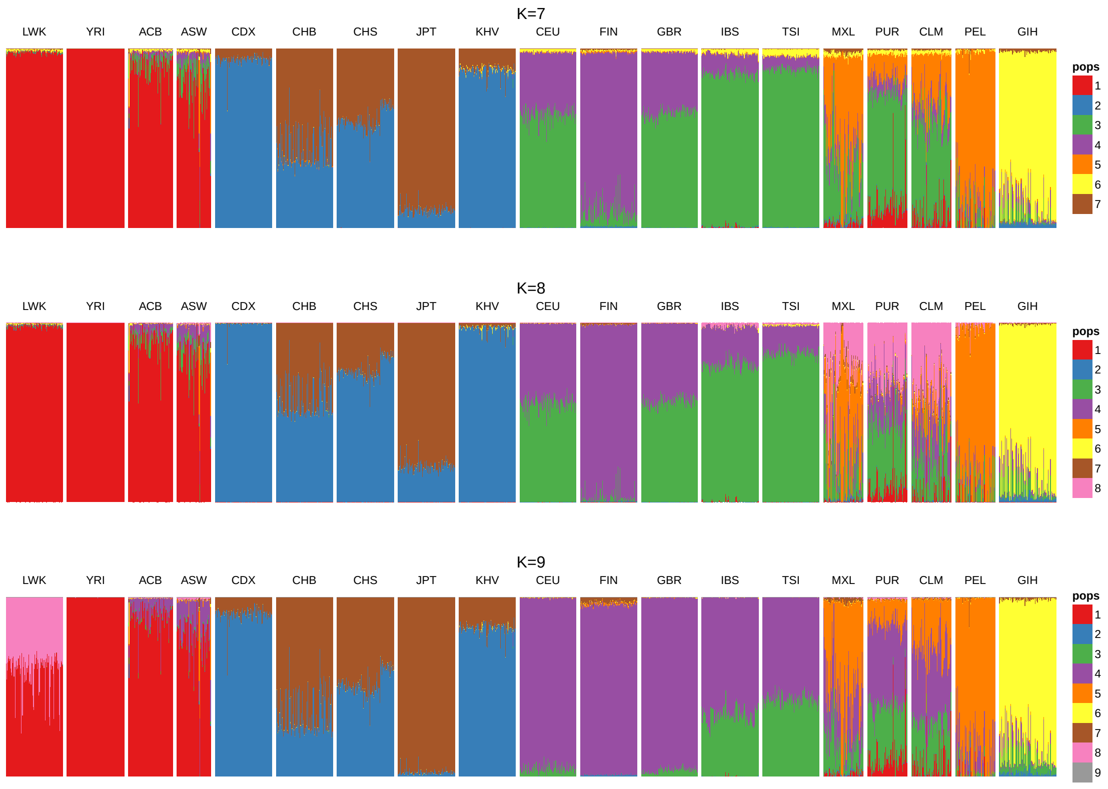


Figure 8.1: Population structure inferred from the TGP data set using the TeraStructure algorithm at three settings for the number of populations K. The visualization of the θ's in the Figure shows patterns consistent with the major geographical regions. Some of the clusters identify a specific region (e.g. red for Africa) while others represent admixture between regions (e.g. green for Europeans and Central/South Americans). The presence of clusters that are shared between different regions demonstrates the more continuous nature of the structure. The new cluster from $K = 7$ to $K = 8$ matches structure differentiating between American groups. For $K = 9$, the new cluster is unpopulated.

(e.g., red for Africa) while others represent admixture between regions (e.g., green for Europeans and Central/South Americans).

Like many modern Bayesian models, this posterior is not tractable to compute: the original algorithm for using the PSD model (Pritchard *et al.*, 2000) and subsequent innovations (Alexander *et al.*, 2009; Raj *et al.*, 2013) are all methods for approximating it. However, as we mentioned above, none of these existing methods can analyze the kind of massive genetic data that is available today. The reason is that each one requires repeatedly iterating through the entire data set to form its approximation. With massive data sets, this is not a practical methodology.

8.3 An SVI algorithm for the PSD model

In this section, we develop a SVI algorithm for the PSD model that has a significantly computational structure from prior work (Alexander *et al.*, 2009; Raj *et al.*, 2013). The algorithm is illustrated in Figure 8.3. At each iteration, it maintains an estimate of the population proportions for each person and the allele frequencies for each population.[1] It repeatedly iterates between the following steps: (a) sample a SNP from the data, $x_{\cdot,\ell}$, the measured genotypes at a single site in the genome across all people, (b) analyze how the current estimates of the ancestral populations explain the genotypes at that SNP, and (c) update the estimates of the latent structure—both the ancestral allele frequencies and per-individual population proportions.

It is the subsampling step of the inner loop that allows TeraStrucure to scale to massive genetic data. Rather than scan the entire population at each iteration, it iteratively subsamples a SNP, analyzes the subsample, and updates its estimate. On small data sets, this leads to faster estimates that are as good as those obtained by the slower procedures. More importantly, it lets us scale the PSD model up to sample sizes that are orders of magnitude greater than what the current state of the art can handle.

8.3.1 Variational Inference

We discussed the main ideas behind variational inference and SVI in Chapter 2. We first parameterize individual distributions for each latent variable in the model, i.e., a distribution for each set of per-population allele frequencies $q(\beta_k)$ and a distribution for each individual's population proportions $q(\theta_i)$. We then fit these distributions so that their product is close to the true posterior, where

[1] We describe and illustrate these quantities as though they are estimates. More technically, the algorithm stores parameterized approximate posteriors to them.

closeness is measured by Kullback-Leibler divergence. Thus we do Bayesian inference by solving the following optimization problem,

$$q^*(\beta, \theta) = \arg\min_q \mathrm{KL}(\prod_k q(\beta_k) \prod_i q(\theta_i) || p(\beta_1, \ldots, \beta_K, \theta_1, \ldots, \theta_n \,|\, x)). \tag{8.1}$$

To finish specifying the objective, we must set the form of $q(\cdot)$. The form of the variational distribution is set to make the problem tractable, that is, for the objective of Equation 8.1 to be computable (as well as its gradients). We define a family of distributions over the hidden variables $q(\cdot)$ indexed by a set of free parameters ν. As with other applications in this thesis, we choose $q(\cdot)$ to be the *mean-field family*, the family where each variable is independent and governed by its own parametric distribution,

$$q(\beta, \theta) = \left(\prod_{k=1}^{K} \prod_{\ell=1}^{L} q(\beta_{k,\ell} \,|\, \hat{\beta}_{k,\ell}) \right) \prod_{i=1}^{N} q(\theta_i \,|\, \hat{\theta}_i). \tag{8.2}$$

Our notation is that $\hat{\theta}_i$ is the variational parameter for the ith individual's population proportions θ_i and $\hat{\beta}_{k,\ell}$ is the variational parameter for the distribution of alleles in population k at location ℓ. Further, we set the form of each factor to be the same form as the prior. Thus $q(\theta_{i,\ell} \,|\, \hat{\theta}_{i,\ell})$ are Dirichlet distributions and $q(\beta_{k,\ell} \,|\, \hat{\beta}_{k,\ell})$ are Beta distributions. These decisions come from the general theory around mean-field variational inference in exponential families (Ghahramani and Beal, 2001; Bishop, 2006). We discuss these decisions in Section 8.3.2.

The objective function of Equation 8.1 is not computable. (It is not computable for the same reason that exact Bayesian inference is intractable—it requires computing the marginal probability of the data.) Thus variational inference optimizes an alternative objective that is equal to the negative KL up to an unknown additive constant,

$$\mathcal{L}(\nu) = \mathbb{E}_q[\log p(\beta, \theta, x)] - \mathbb{E}_q[\log q(\beta, \theta \,|\, \nu)]. \tag{8.3}$$

This variational family is flexible – it represents different individuals with different population proportions. In Figure 8.1 we plotted the variational expectation of each individuals population parameters distribution $\mathrm{E}\left[\theta_i \,|\, \hat{\theta}_i\right]$.

With these components—the objective of Equation 8.3 and the variational family of Equation 8.2—we have turned the inference problem for the PSD model into an optimization problem.

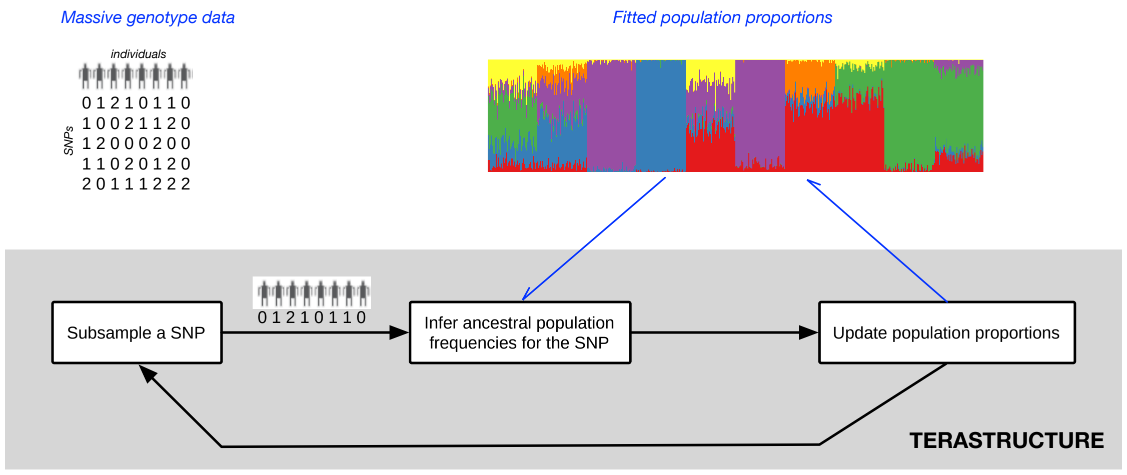


Figure 8.2: A schematic diagram of stochastic variational inference for the Pritchard-Stephens-Donnelly (PSD) model. The algorithm maintains an estimate of the latent population proportions for each individual. At each iteration it samples SNP measurements from the large database, infers the per-population frequencies for that SNP, and updates its idea of the population proportions. This is much more efficient than algorithms that must iterate across all SNPs at each iteration.

8.3.2 Stochastic Variational Inference

TeraStructure solves this optimization problem with *stochastic variational inference* (Hoffman *et al.*, 2013). Recall from Chapter 2 that SVI is an adaptation of the classical stochastic optimization algorithm (Robbins and Monro, 1951) to variational inference (see Chapter 2). Specifically, we optimize the KL divergence by following noisy realizations of its derivatives, where the noise emerges from our subsampling the data at each iteration. The noisy derivatives are much cheaper to compute than the true derivatives, which require iterating over the entire data set.

Global and local parameters

Before we develop our algorithm, we use the conditional dependencies in our graphical model to divide our variational parameters into *local* and *global* (Hoffman *et al.*, 2013).

In each iteration we subsample allele measurements for all individuals at a SNP location l. Our sampled observations are $x_{1:N,l}$. Under the PSD model, given individual proportions θ the sample $x_{1:N,l}$ and the allele frequencies $\beta_{1:K,l}$ are conditionally independent of all other observations and allele frequencies $\beta_{1:K,-l}$. Thus, the allele frequencies $\beta_{1:K,l}$ are local to the observations $x_{1:N,l}$. The per-individual population proportions θ_i, however, are not local to the observation; they govern the distribution of observations at all SNP locations. Therefore, the θ are global variables. Following Hoffman *et al.* (2013), we extend this notion of global and local sets to the variational

```
1. For all users $i \in 1, \cdots, N$, initialize the population proportions $\theta_i$ randomly (see Sec-
   tion 8.3.2)
2. repeat
3.   Sample a location $l$ and all observations $x_{.,l}$ at that location
4.   For $k \in 1, \cdots, K$, initialize $(\hat{\beta}_{lk,0}, \hat{\beta}_{lk,1})$ at location $l$ to $(a, b)$,
5.   Local step: repeat
6.     For $k \in 1, \cdots, K$ and $i \in 1, \cdots, N$ set
           $\phi_{il,k} \propto \exp\Big\{\mathrm{E}[\log\theta_{i,k}] + \mathrm{E}[\log\beta_{k,l}]\Big\}$
           $\xi_{il,k} \propto \exp\Big\{\mathrm{E}[\log\theta_{i,k}] + \mathrm{E}[\log(1-\beta_{k,l})]\Big\}$
7.     For $k \in 1, \cdots, K$ set the Beta parameters at location $l$
           $\hat{\beta}_{lk,0} = a + \sum_{i=1}^{N} x_{i,l}.\phi_{il,k}$
           $\hat{\beta}_{lk,1} = b + \sum_{i=1}^{N} (c - x_{i,l})\xi_{il,k}$                    (8.4)
8.   until local parameters $\phi_{.,l}$, $\xi_{.,l}$ and $\hat{\beta}_{.,l}$ converge
9.   Global step: For $i \in 1, \cdots, N$ set the population proportions
           $\hat{\theta}^{t}_{i,k} = (1-\rho_t)\hat{\theta}^{(t-1)}_{i,k} + \rho_t L(\alpha_k + x_{i,l}\phi_{il,k} + (c - x_{i,l})\xi_{il,k})$     (8.5)
10.  Set the step-size $\rho_t = (\tau_0 + t)^{-\kappa}$
11. until convergence criteria is met
```

Figure 8.3: Stochastic variational inference for the PSD model.

parameters. Given observations $x_{l,1:N}$ at the location l, the $\hat{\theta}$ are the global variational parameters; the $\hat{\beta}_{1:K,l}$ are the local variational parameters.

In stochastic variational inference (Hoffman *et al.*, 2013), we iteratively update local and global parameters. In each iteration, we first subsample a SNP location l and compute optimal local parameters for the sample, given the current settings of the global parameters. We then update the global parameters using a stochastic natural gradient (Amari, 2001) of the variational objective computed from the subsampled data and the local parameters.

We will now develop our algorithm by first obtaining closed forms updates for our local and global variational parameters. For the local parameters, we will derive optimal coordinate updates; for the global parameters, we will derive the stochastic natural gradient update.

Computing the optimal local parameters

Given the global variables θ, we can optimize local parameters $\beta_{\cdot,l}$ in closed form under certain assumptions. These assumptions involve the *complete conditionals* of the hidden variables in the model, and the variational family. A complete conditional is the conditional distribution of a latent variable given the observations and the other latent variables in the model (Ghahramani and Beal, 2001). If the complete conditional of a variable is in the same family as its prior, and the corresponding variational distribution is in the same family, then we can optimize its variational parameter by setting it to the expected natural parameter (under q) of the complete conditional.

If the complete conditional of each latent variable is in the same exponential family as its prior distribution, then the model is *conditionally conjugate.*

The complete conditional for the $\beta_{k,l}$ at a sampled location ℓ are

$$\begin{aligned} p(\beta_{k,l}|\beta_{-k,l},\theta,x) &\propto p(\beta_{k,l}|a,b)\prod_{i=1}^{N} p(x_{i,l}|\theta_i,\beta_{1:K,l}) \\ &\propto \exp\Big\{(a-1)\log\beta_{k,l}+(b-1)\log(1-\beta_{k,l}) \\ &\qquad +\sum_n x_{n,l}\log\sum_k \theta_{n,k}\beta_{k,l}+\sum_n (c-x_{n,l})\log(1-\sum_k\theta_{n,k}\beta_{k,l})\Big\}. \end{aligned} \tag{8.6}$$

The complete conditional in Equation 8.6 is not in the exponential family because the expectation of the second and third log-of-summation terms with respect to the variational family q are intractable. Therefore, the PSD model of Section 8.2 is not conditionally conjugate.

To overcome the nonconjugacy in the model, we introduce multinomial approximations using the zeroth order delta method for moments (Bickel and Doksum, 2007; Wang and Blei, 2013). These approximations provide a lower bound to these intractable terms in the evidence lower bound. In particular, we introduce auxiliary K-multinomial distributions $q(\phi)$ and $q(\xi)$,

$$\begin{aligned} \log\left(\textstyle\sum_k \theta_{i,k}\beta_{k,l}\right) &\geq \textstyle\sum_k \phi_{il,k}\log\frac{\theta_{i,k}\beta_{k,l}}{\phi_{il,k}}, \\ \log(1-\textstyle\sum_k \theta_{i,k}\beta_{k,l}) &\geq \textstyle\sum_k \xi_{il,k}\log\frac{\theta_{i,k}(1-\beta_{k,l})}{\xi_{il,k}}. \end{aligned} \tag{8.7}$$

Notice that these distributions $q(\phi)$ and $q(\xi)$ approximate only the conditionals of the allele frequencies local to the sampled location l. Therefore, the parameters to these distributions are also local to l.

Substituting the lower bounds from Equation 8.7 in Equation 8.6, the complete conditional is

$$p(\beta_{k,l}|\beta_{-k,l},\theta,x) \propto \text{Beta}\Big(a+\sum_{i=1}^{N} y_{i,l}\phi_{il,k}, b+\sum_{i=1}^{N}(c-y_{i,l})\xi_{il,k}\Big). \tag{8.8}$$

Our approximation has effectively placed the complete conditional of allele frequency $\beta_{k,l}$ in the exponential family. By choosing the variational distribution $q(\beta_{k,\ell}\,|\,\hat{\beta}_{k,\ell})$ from Equation 8.2 to be the Beta distribution, the same family as the prior distribution, we satisfy the conditions for a closed form coordinate update for the local parameters $\hat{\beta}_{1:K,l}$. The optimal $\hat{\beta}_{k,l}$ is the expected natural parameter of the complete conditional in Equation 8.8 (Hoffman *et al.*, 2013).

Another perspective on the approximations in Equation 8.7 is they lead to a computationally efficient lower bound on the objective of Equation 8.3.

Computing stochastic gradient updates

We now turn to the stochastic optimization of the population proportions parameter $\hat{\theta}_n$ using the subsampled observations $x_{1:N,l}$ at location l. We compute noisy estimates of the natural gradient (Amari, 2001) of the variational objective with respect to $\hat{\theta}_n$, and we follow these estimates with a decreasing step-size. Following Hoffman *et al.* (2013), we can compute the natural gradient of Equation 8.3 with respect to the global variational parameter $\hat{\theta}_i$ by first computing the coordinate update and then subtracting the current setting of the parameters.

To compute the coordinate update for $\hat{\theta}_i$, we write down its complete conditional. For the population proportions θ_i, the complete conditional is

$$\begin{aligned} p(\theta_i|\beta,x) &\propto p(\theta_i|c)\prod_{l=1}^{L}\prod_{k=1}^{K} p(\beta_{k,l}|a,b)\prod_{l=1}^{L} p(x_{i,l}|\theta_i,\beta_l) \\ &\propto \exp\Big\{\sum_k (c-1)\log\theta_{i,k} \\ &\qquad + \sum_l x_{i,l}\log\sum_k \theta_{i,k}\beta_{k,l} + \sum_l (c-x_{i,l})\log(1-\sum_k \theta_{i,k}\beta_{k,l})\Big\}. \end{aligned} \tag{8.9}$$

Similar to the complete conditionals of the local variables in Equation 8.6, the complete conditional in Equation 8.9 is not in the exponential family. We use the multinomial approximations in Equation 8.7 to bring the complete conditional into the exponential family, and in the same family as the prior distribution over the population proportions:

$$p(\theta_i|\beta, x) \propto \text{Dirichlet}\Big(\alpha_k + \sum_{l=1}^{L}(x_{i,l}\phi_{il,k} + (c - x_{i,l})\xi_{il,k})\Big). \tag{8.10}$$

Following Hoffman *et al.* (2013), the stochastic natural gradient of the variational objective with respect to the global parameter $\hat{\theta}_i$, using L replicates of $x_{i,l}$ is

$$\frac{\partial \mathcal{L}(\nu)}{\partial \theta_i} = \alpha + L(x_{i,l}\phi_{il,k} + (c - x_{i,l})\xi_{il,k}) - \theta_i. \tag{8.11}$$

Notice we have used the expected natural parameter from the complete conditional in Equation 8.10 in Equation 8.11. We arrive at this form of the natural gradient, by premultipling the gradient by the inverse Fisher information, and replacing the summation over all SNP locations in Equation 8.11 with a summation over L replications from the sampled location. Equation 8.11 is a noisy natural gradient of a lower bound on the variational objective.

To optimize the variational objective with respect to the population proportions $\hat{\theta}_i$, we use the natural gradients in Equation 8.11 in a Robbins-Monro algorithm (Hoffman *et al.*, 2013). At each iteration we update the global variational parameter with a noisy gradient computed from the SNP observations at location l. The step-size at iteration t is ρ_t, and is set using the schedule

$$\rho_t = (t + \tau)^{-\kappa}. \tag{8.12}$$

This satisfies the Robbins-Monro conditions on the step-size, and guarantees convergence to a local optimum of the variational objective.

The stochastic algorithm

The full algorithm is shown in Figure 8.3. For each iteration, we first subsample a SNP location l and compute optimal local parameters $(\phi_{1:N,l}, \xi_{1:N,l}, \hat{\beta}_{1:K,l})$ for the sample, given the current settings of the global parameters. We then update the global parameters $\hat{\theta}_i$ of all individuals using stochastic natural gradients of the variational objective computed from the subsampled data and local parameters.

Memory efficient computation

During training the stochastic variational inference algorithm is only required to keep the variational population proportions $\hat{\theta}_{i,k}$ for all individuals $i \in 1, \cdots, N$ in memory. For a given location, the optimal local parameters $(\phi_{1:N,l}, \xi_{1:N,l}, \hat{\beta}_{1:K,l})$ can be computed using the local optimization steps—steps 6 to 9—in Figure 8.3. This drastically cuts the memory needed. The memory requirement is therefore $O(NK)$ where N is the number of individuals and K is the number of latent ancestral populations. The algorithm also results in a compact fitted model state. The fitted model state comprises of the estimated $\hat{\theta}_i$ for all individuals. The allele frequencies $\hat{\beta}_{1:K,l}$ can be *queried* for any given location l using the local step.

Linear scaling in the number of threads

We can compute the local steps and the global steps in parallel across T threads. First, we "map" the individuals into T disjoint sets, and each thread is responsible for computation on one of these sets. Notice that each thread can independently compute the local parameters $(\phi_{n,l}, \xi_{n,l})$ for any individual n that it owns. This corresponds to step 6 of the algorithm in Figure 8.3. Further, the sums required in step 7 can also be computed in parallel. The "reduce" step consists of aggregating the per-thread sums in step 7, and estimating the new Beta parameters. This is an $O(T + K)$ operation. Since T and K are small constants, our reduce step is inexpensive. The global step in step 9 can also be computed in parallel.

Given T threads, the computational complexity of the stochastic algorithm is $O(\frac{NK}{T})$. The algorithm is dominated by the parallel computation in steps 6 and 9, which scale linearly in the number of threads T. By increasing T, we can scale our algorithm almost linearly in the number of threads.

Initializing variational parameters

We initialize the population proportions randomly using $\theta_{ik} \sim \text{Gamma}(100, 0.01)$. Within the local step, we initialize $(\hat{\beta}_{lk,0}, \hat{\beta}_{lk,1})$ at location l to the prior parameters (a, b).

Assessing convergence using a validation set

We hold out a *validation set* of genotype observations, and evaluate the predictive accuracy on that set to assess convergence of the stochastic algorithm in Figure 8.3 (Geisser and Eddy, 1979). These observations are treated as missing during training.

The validation set is chosen with computational efficiency in mind. We will periodically evaluate the heldout log likelihood on this set (the *validation log likelihood*) to determine convergence of the algorithm in Figure 8.3. By choosing individuals from a small fraction of total locations L, we ensure that this periodic computation is only required to recompute the optimal $\hat{\beta}_{\cdot,k}$ for those locations.

The TeraStructure algorithm stops when the change in validation log likelihood is less than 0.0001%. We measure this change over $100,000$ iterations for the data sets with $N >= 100,000$; we measure this change across $10,000$ iterations for smaller data sets.

For the validation set, we uniformly sample at random 0.5% of the L locations, and at each location we uniformly sample at random and keep aside observed alleles for r individuals. The number of per-location held out individuals r is set to $N/100$ for large N ($N > 2000$) and otherwise to $N/10$. This allows for a reasonably small fraction of individuals to be held out from each location. Further, r is limited to a maximum of 1000 individuals for any N.

8.4 Empirical study on massive data sets

We applied TeraStructure to both real and simulated data sets to study and demonstrate its good performance. We compared it to ADMIXTURE (Alexander *et al.*, 2009) and fastSTRUCTURE (Raj *et al.*, 2013), the two algorithms for estimating the PSD model that work on modestly sized data. In our comparisons, we timed all the algorithms under equivalent computational conditions. On simulated data, where the truth is known, we measured the quality of the resulting fits by computing the KL divergence between the estimated models and the truth. On the real data sets, where the truth is not known, we measured model fitness by predictive log likelihood of held-out measurements (Methods). The smaller the KL divergence and the larger the predictive likelihood, the better a method performs.

Real Data sets

We first analyzed two real data sets: the Human Genome Diversity Panel (HGDP) data set (Cann *et al.*, 2002; Cavalli-Sforza, 2005) and the 1000 Genomes Project (TGP) (Abecasis, 2012). After preprocessing, HGDP consisted of 940 individuals at 642K SNPs for a total of 604 million observed genotypes and TGP consisted of 1,718 people at 1,854,622 SNPs for a total of 3.2 billion observed genotypes. The preprocessing for HGDP consisted of removing individuals not in the "H952" set (Rosenberg *et al.*, 2006), which leaves us with only the individuals without first or second degree relatives in the data. The preprocessing for TGP consisted of removing related individuals using

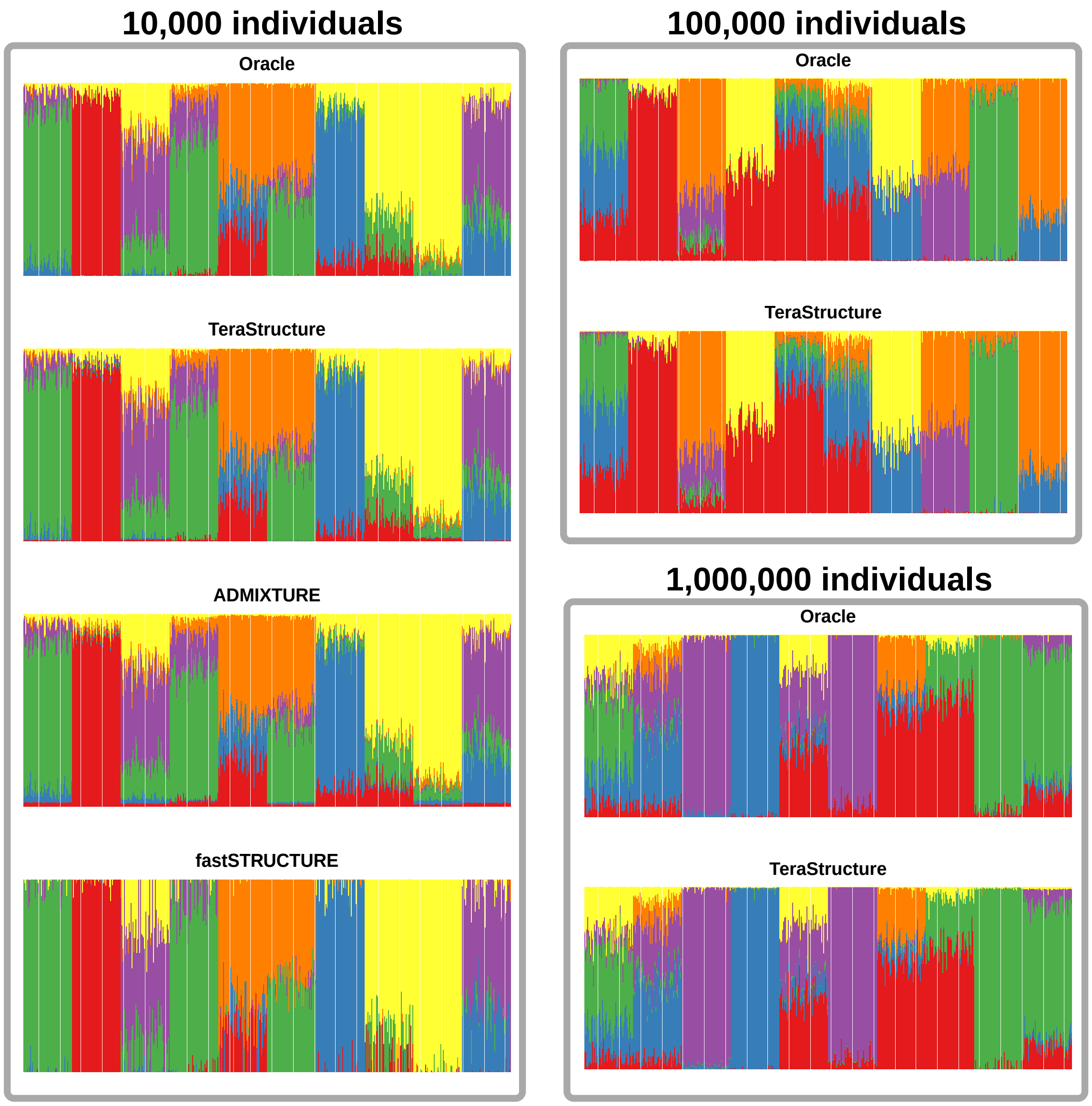


Figure 8.4: TeraStructure recovers the ground truth per-individual population proportions on the synthetic data sets with high accuracy. Each panel shows a visualization of the ground truth θ_i^* and the inferred $\mathrm{E}\left[\theta_i | \hat{\theta}_i\right]$ for all individuals in a data set. The current state-of-the-art algorithms cannot complete their analyses of 100,000 and 1,000,000 individuals. TeraStructure is able to analyze data of this size and gives highly accurate estimates.

the sample information provided by the 1000 Genomes Project. Further, we removed individuals for 95% genotyping completeness and removed the SNPs with lower than 1% minor allele frequency.

Synthetic data sets

The goal of our study on synthetic data sets is to demonstrate scalability to tera-sized data sets—one million observed genotypes from one million individuals—while maintaining high accuracy in recovering ground truth per-individual population proportions θ and allele frequencies β. To this end, we generated synthetic genotype data using the PSD model (Pritchard *et al.*, 2000). A specification of the the per-individual population proportions and the population allele frequencies is our "ground truth". To generate realistic synthetic data, we made the individual θ_i's visually similar to the proportions obtained from fitting our model to the TGP data set. We modeled our allele frequencies β_ℓ from real data.

In our simulation, the process of drawing an individual i's proportions θ_i has two levels. At the first level, we drew S points in the K-simplex from a symmetric Dirichlet distribution, $q_s \sim \text{Dirichlet}(\alpha)$. Each of the S points represents a "region" of individuals, and each individual was assigned to one of the regions such that the regions are equally sized. Then, we drew the population proportions of each individual, $\theta_i \sim \text{Dirichlet}(\gamma q_{s,1}, \ldots, \gamma q_{s,K})$. Thus, each region has a fixed q_s and the proportion of individuals from that region are governed by the same scaled q_s parameter. The parameter q_s controls the sparsity of the θ_i, while the parameter γ controls how similar admixture proportions are within each group. For all simulations, we set $S = 50$, $\alpha = 0.2$, and $\gamma = 50$.

Each β_ℓ at a SNP location ℓ, consists of K independent draws from a Beta distribution with parameters following that of the Balding-Nichols Model (Balding and Nichols, 1995), i.e. $\beta_\ell \sim \text{Beta}(\frac{1-F_\ell}{F_\ell} p_\ell, \frac{1-F_\ell}{F_\ell}(1-p_\ell))$ where p_ℓ is the marginal allele frequency and F_ℓ is the Wright's F_{ST} at location ℓ. The paired parameters p_ℓ and F_ℓ were estimated from the HGDP data set described earlier. For each pair, we chose a random complete SNP from the HGDP data and set the allele frequency p_ℓ to the observed frequency. The Wright's F_{ST} F_ℓ was set to the Weir & Cockerham $\mathrm{F_{ST}}$ estimate (Weir and Cockerham, 1984) with 5 discrete subpopulations, following analysis of the HGDP study in Rosenberg *et al.* (2006). We simulated data with 1,000,000 SNPs and four different scales of individuals: 1,000, 10,000, 100,000 and 1,000,000. With 1 million individuals and 1 million SNPS, the number of observations is tera-sampled-sized, i.e., 10^{12} observations.

Metrics

On real data sets, we computed the predictive accuracy on a *test set* of observed alleles by computing the heldout log likelihood under the PSD model. The test set is chosen to enable a fair comparison to other algorithms. We hold out alleles for 0.5% of the N individuals from each location $l \in 1, \cdots, L$. A better predictive accuracy corresponds to a better fit to the data (Geisser and Eddy, 1979). We approximate the predictive distribution of a heldout SNP using posterior estimates of $\hat{\theta}$ and $\hat{\beta}$.

On synthetic data sets, we measured the accuracy in recovering the ground truth population proportions. We computed the Kullback Leibler divergence (Kullback and Leibler, 1951) of the Multinomial governed by the variational posterior estimate $\hat{\theta}$ to the true population proportions θ^* for each individual. We then compared the median KL divergence across all individuals.

Hyperparameters

We set the Dirichlet parameter c to $\frac{1}{K}$ to enforce a sparse prior on the per-individual population proportions. We set the learning rate parameters, τ_0 to 1 and κ to 0.5, to allow learning rapidly in the early iterations. Finally, we set the hyperparameters a and b to 1 to enforce a uniform prior on the population parameters $\beta_{1:L,1:K}$. We used the same hyperparameter settings in all of our experiments.

Open source software

Our software is implemented in C++ and has 5400 lines of code. It uses the POSIX Threading library for multi-threaded computation. It inputs genotype data in text or PLINK format (Purcell *et al.*, 2007) and outputs the population proportions $\mathrm{E}\left[\hat{\theta}\right]$. An additional software tool runs local variational inference to write out the expected allele frequency Beta parameters corresponding to a list of locations. To run this program, one provides the $\hat{\theta}$, and a list of SNP locations. Our software is available at *http://github.com/premgopalan/popgen*.

Computing hardware

All experiments were run on a single multicore machine with two Intel Xeon E5-2680v2 processors with 10 cores each and running at 2.8GHz. The maximum memory required for our experiments is 10GB.

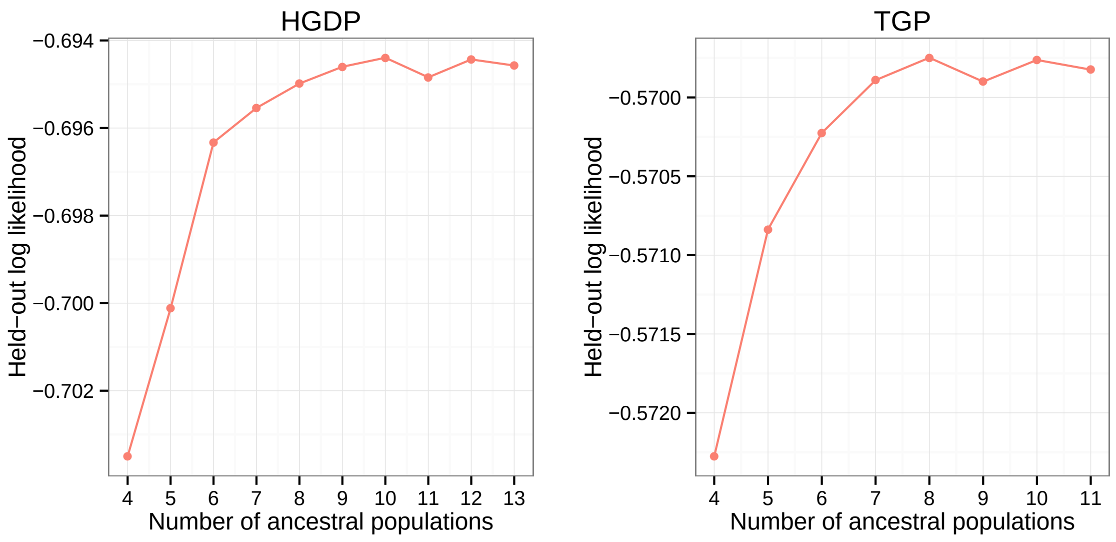


Figure 8.5: Predictive log likelihood as a function of the number of ancestral populations on the Human Genome Diversity Panel (HGDP) and 1000 Genomes Project (TGP) data sets. The HGDP data peaks at 10 population, and the TGP data peaks at 8 populations.

Data set	N	Mean predictive log likelihood		
		TeraStructure	ADMIXTURE	fastSTRUCTURE
HGDP	940	-0.71	-0.71	-0.71
TGP	1718	-0.60	-0.60	-0.61

Table 8.1: The predictive accuracy of TeraStructure is comparable to the ADMIXTURE (Alexander *et al.*, 2009) and the fastSTRUCTURE (Raj *et al.*, 2013) algorithms, implying a similar model fit. The mean test log likelihood under the model fits is shown. We generated 5 test sets at random and computed the mean over these heldout sets. N is the number of individuals in the data set. The number of ancestral populations is set to $K = 10$ for HGDP and $K = 8$ for TGP.

Data set	Replication	N	L	Median per-individual KL divergence		
				TeraStructure	ADMIXTURE	fastSTRUCTURE
syn10K	1	10,000	1,000,000	**0.016**	0.020	6.68
syn10K	2	10,000	1,000,000	**0.009**	0.019	5.15
syn10K	3	10,000	1,000,000	**0.020**	0.022	4.49
syn100K	1	100,000	1,000,000	**0.006**	–	–
syn100K	2	100,000	1,000,000	**0.013**	–	–
syn100K	3	100,000	1,000,000	**0.009**	–	–
syn1M	1	1,000,000	1,000,000	**0.015**	–	–

Table 8.2: The accuracy of the algorithms on synthetic data. TeraStructure is the only algorithm that was able to complete its analysis on the synthetic data sets with $N = 100,000$ individuals and $N = 1,000,000$ individuals. On these massive data sets, TeraStructure found a highly accurate fit to the data (see also Figure 8.4). On smaller synthetic data, TeraStructure finds a fit to the data that is closer to the ground truth than either of the other methods. The number of ancestral populations is set to the number of ground truth ancestral populations: K=6.

8.4.1 Results

In previous work, ADMIXTURE and fastSTRUCTURE have been shown to perform reasonably well on real data sets of the size corresponding to TGP and HGDP (Alexander *et al.*, 2009; Raj *et al.*, 2013). In applying all three algorithms to these data, we found that TeraStructure equalled the predictive log likelihood on held-out data obtained by the competing methods (see Table 8.1). TeraStructure also completed its estimation in a significantly shorter period of time (Table 8.3).

We then studied the algorithms on synthetic data. We designed these data sets be similar to real genetic data sets, but at sizes that push the limits of what is available today. We simulated data sets consisting of 10,000 individuals, 100,000 individuals, and 1M individuals, each with 1M SNP genotypes per individual. On these data we know the true individual proportions, and we can visualize how well each algorithm reconstructs them (Figure 8.4). We found that ADMIXTURE and fastSTRUCTURE were only able to analyze the 10,000-individual set, on which TeraStructure was both 2-3 times faster and more accurate (Tables 8.3 and 8.2). More importantly, TeraStructure was the only algorithm that was able to analyze the larger data sets of 100,000 individuals and 1M individuals, and again with high accuracy (Figure 8.4 and Table 8.2).

TeraStructure uses a convergence criterion to decide when to stop iterating, as described in Section 8.3.2. This lets us measure how many SNPs were subsampled before the algorithm had learned the structure of the population. On the HGDP and TGP data, we found that TeraStructure needed to sample $\sim$ 90% and $\sim$ 50% of the SNPs, respectively, before converging (Table 8.3). On the tera-sample-sized data set of 1M individuals by 1M SNPs, TeraStructure sampled $\sim$ 50% of the

Data set	N	L	S	Time (hours)		
				TeraStructure	ADMIXTURE	fastSTRUCTURE
HGDP	940	644,258	0.9	< 1	< 1	12
TGP	1718	1,854,622	0.5	3	3	21
syn10K	10,000	1,000,000	1.0	9	28	216
syn100K	100,000	1,000,000	0.7	158	–	–
syn1M	1,000,000	1,000,000	0.5	509	–	–

Table 8.3: The running time of all algorithms on both real and synthetic data. TeraStructure is the only algorithm that can scale beyond $N = 10,000$ individuals to the synthetic data sets with $N = 100,000$ individuals and $N = 1,000,000$ individuals. S is the fraction of SNP locations subsampled, with repetition, during training; L is the number of SNP locations. $S * L$ also equals the number of training iterations of the outer loop in the algorithm of Figure 8.3 prior to convergence, since we subsample one SNP location in each iteration. The TeraStructure and ADMIXTURE algorithms were run with ten parallel threads, while fastSTRUCTURE, which does not have a threading option, was run with a single thread. Even under the best-case assumption of ten times speedup due to parallel computation, the TeraStructure algorithm is twice as fast as both ADMIXTURE and fastSTRUCTURE algorithms on the data set with $N = 10,000$ individuals. On the real data sets, TeraStructure is faster than the other algorithms. In contrast to other methods, TeraStructure iterated over the SNP locations at most once on all data sets.

SNPs before converging.

When analyzing data with the PSD model, we must choose the number of ancestral populations K. For real data, TeraStructure addressed this model selection problem using a predictive approach (Geisser and Eddy, 1979). We held out a set of genome locations for each individual and computed the average predictive log likelihood under the model for varying numbers of ancestral populations. The best choice of K is the one that assigns the highest probability to the held-out set. Our sensitivity analysis revealed that $K = 8$ had the highest validation likelihood on the TGP data, while $K = 10$ had the highest on the HGDP data (Figure 8.5). On the real data sets, we used the optimal values of K (Table 8.1); on simulated data sets, we set K to the number of ground truth ancestral populations (Table 8.3).

8.5 Discussion

Genomic studies are growing and it is vital that our statistical algorithms can scale to trillions or more data points. Algorithms that require multiple iterations over the entire data fail in this setting. But methods like TeraStructure, methods that repeatedly take strategic subsamples of the data to iteratively build a global picture of its latent structure, are positioned to succeed. We have shown that TeraStructure can accurately fit a rich probabilistic model of population genetic structure on data sets with a million individuals and 10^{12} observed genotypes. This is a jump of orders of magnitude beyond the capabilities of current state-of-the-art algorithms. Using TeraStructure to analyze real data sets of the tera-sample-size will provide the most comprehensive analyses to date of the global population genetics of humans.

TeraStructure's computational flow is simple: it iterates between subsampling observed SNPs from the data, analyzing the subsample, and updating its estimate of the hidden ancestral populations. The main ideas in this chapter can be adapted to many Bayesian models that are used in modern genetics research, such as HMMs, phylogenetic trees, and others.

Unlike the previous chapters, genetic variation data sets are dense. In our real data sets, roughly 50% of the entries of the discrete matrix of measured alleles were non-zero. Our algorithms derive scalability from subsampling within the stochastic variational inference framework, and from using multiple threads for the trivially parallellizable local step in Figure 8.3.

One direction of future work is to consider more efficient subsampling. The algorithm of Figure 8.3 subsamples a single location at each iteration, but includes the allele measurements from all individuals as observations. A improved strategy would be to subsample from both individuals and the SNP locations. Further, we may consider informative subsampling strategies, as explored in the context of network models in Chapter 6. One strategy would be to subsample the locations that have the most variability in allele measurements.

We have presented probabilistic models and scalable inference algorithms for several types of discrete data: user behavior, text, networks and genetic variation. In the next chapter, we summarize the ideas behind our models and inference algorithms and point to directions of future work.

Chapter 9

Contributions and future research

The preceding chapters developed statistical models and inference algorithms for learning from massive discrete data sets. We now summarize the principal contributions underlying our results, and outline avenues for future research.

9.1 Summary of methods and contributions

In this thesis, we have developed statistical inference algorithms for analyzing massive discrete data sets derived from user consumption, network interactions and genetic variations. We used directed graphical models to describe our modeling assumptions about the data, and developed variational inference or stochastic variational inference algorithms. We identify three principal contributions underlying our results.

- *Scaling intractable models.* For several models in this thesis a straightforward application of classic inference methods such as Markov Chain Monte Carlo, variational inference or even stochastic variational inference fails to scale them to large data sets. The AMP model of network popularity (Chapter 6) and the Bayesian model of genetic variations (Chapter 8) are nonconjugate models. In both cases, we develop approximate posterior inference for a tractable lower bound of the variational objective.

- *Subsampling under SVI.* We studied new non-uniform subsampling methods in this thesis. In the assortative MMSB model (Airoldi *et al.*, 2008) of network communities of Chapter 6, the Markov blanket of each node includes the variables associated with all other nodes. Under this challenging setting, we developed scalable SVI algorithms that subsample network pairs non-uniformly. We presented the link sampling method that relied on positing a variational family conditional on the data (see Section 6.3.3), and informative set sampling (see Section 6.3.2) that subsampled network pairs with a bias towards pairs that help estimation.

- *Modeling insights.* Using posterior predictive checks we demonstrated that the hierarchical Poisson factorization model of Section 3.3.2 captures the long-tailed user consumption activity well. Further, the additive PF models provide sparse latent representations of users and items. In particular, we demonstrated that coupling the latent spaces across article content and article readership in the CTPF model allows user preferences for articles to be interpreted as affinity to latent topics. We used these latent representations in exploratory analysis of large data sets in Chapter 4. In Chapter 5, we showed that the stick-breaking construction of the Gamma process, derived from a corresponding construction for the Dirichlet process, results in efficient mean-field inference for the Bayesian nonparametric Poisson factorization model.

In conclusion, we have developed scalable inference for a range of sophisticated models: Bayesian nonparametric models (Section 5.1), generalized linear models (Section 6.1.2), Bayesian hierarchical models (Section 3.3.2). On real-world data sets and synthetic data sets, we have demonstrated that our algorithms yield good predictive performance and provide an exploratory tool for the hidden structure in the data.

9.2 Suggestions for future research

We conclude by discussing open research directions suggested by our approaches to scaling latent variable models to large data sets.

Two broad areas of future work include learning algorithms for streaming data sets (Broderick *et al.*, 2013b) and using our models as building blocks in more sophisticated models, for example, Bayesian nonparametric models or a recommendation model that combines network, text, user and item covariates and user ratings. We now enumerate a few specific suggestions for future work.

- *Better subsampling methods for the PSD model.* In Chapter 8 we have shown that TeraStructure can accurately fit a rich probabilistic model of population genetic structure on data sets with a million individuals and 10^{12} observed genotypes. The algorithm of Figure 8.3 subsamples a single location at each iteration, but includes the allele measurements from all individuals as observations. A improved strategy would be to subsample from both individuals and the SNP locations. Further, we may consider informative subsampling strategies, as explored in the context of network models in Chapter 6. One strategy would be to subsample the locations that have the most variability in allele measurements.

- *Modeling covariates.* One of the main advantages of taking a probabilistic approach to network analysis (Chapter 6) is that the models and algorithms are reusable in more complex settings. Our strategy for analyzing networks easily extends to other probabilistic models, such as those taking into account node covariates. For example, the AMP model of networks (Section 6.1.2) can be extended to incorporate observed and hidden node covariates, but we have only studied hidden node variables in Chapter 6. Recent research from Krivitsky *et al.* (2009) and Kim and Leskovec (2011) will be informative and relevant to this step.

- *Posterior predictive checks.* For what types of data do our models and algorithms work best? The MMSB and the AMP models are based on the assumption that nodes assume a single latent community during interactions. Subsequently, each node is associated with normalized mixed-memberships. However, in social networks an interaction may be made stronger by multiple shared similarities between two people. An interesting venue for future work is to explore models that can aggregate the effect of multiple shared communities between nodes in explaining links between them.

  Posterior predictive checks (Rubin, 1984; Gelman *et al.*, 1996) are effective tools for such model assessment. The idea behind a PPC is to simulate a complete data set from the posterior predictive distribution—the distribution over data that the posterior induces—and then compare the generated data set to the true observations. A good model will produce data that captures the important characteristics of the observed data.

- *Comparison to Gaussian MF with downweighted zeros.* As noted in Chapter 4, a comparison of HPF to Gaussian MF with downweighted zeros (Hu *et al.*, 2008) is important future work. A related effort is to similarly extend the APF models to capture greater uncertainty around missing/zero ratings.

While we have focussed on specific applications in modeling user behavior, network interactions and human genetic variation, the statistical methods developed in this thesis extend to a variety of discrete data sets. Examples include finding overlapping communities of functionally similar chromosomes from chromosome folding data or the study of fMRI data. On advanced computing architectures, our algorithms can likely analyze much larger data sets. Further advances in this area are needed to make scalable Bayesian data analysis a standard tool in the scientist's toolbox.

Bibliography

Abecasis, G. R. (2012). An integrated map of genetic variation from 1,092 human genomes. *Nature*, **491**(7422), 56–65.

Adamic, L. A. and Glance, N. (2005). The political blogosphere and the 2004 U.S. election: divided they blog. In *LinkKDD*, LinkKDD '05, page 3643, New York, NY, USA. ACM.

Agarwal, D. and Chen, B. (2010). fLDA: Matrix factorization through latent Dirichlet allocation. In *Proceedings of the third ACM international conference on web search and data mining*, pages 91–100. ACM.

Ahn, Y., Bagrow, J. P., and Lehmann, S. (2010). Link communities reveal multiscale complexity in networks. *Nature*, **466**(7307), 761–764.

Aiello, W., Chung, F., and Lu, L. (2000). A random graph model for massive graphs. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, pages 171–180.

Airoldi, E., Blei, D., Fienberg, S., and Xing, E. (2008). Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, **9**, 1981–2014.

Alexander, D. H., Novembre, J., and Lange, K. (2009). Fast model-based estimation of ancestry in unrelated individuals. *Genome research*, **19**(9), 1655–1664.

Amari, S. (1982). Differential geometry of curved exponential families-curvatures and information loss. *The Annals of Statistics*, **10**(2), 357–385.

Amari, S. (2001). Information geometry on hierarchy of probability distributions. *IEEE Transactions on Information Theory*, **47**(5), 1701–1711.

Amini, A., Chen, A., Bickel, P., and Levina, E. (2012). Pseudo-likelihood methods for community detection in large sparse networks. *arXiv preprint arXiv:1207.2340*.

Aral, S. and Walker, D. (2012). Identifying influential and susceptible members of social networks. *Science*, **337**(6092), 337–341.

Balding, D. and Nichols, R. (1995). A method for quantifying differentiation between populations at multi-allelic loci and its implications for investigating identity and paternity. *Genetica*, **96**(1-2), 3–12.

Ball, B., Karrer, B., and Newman, M. (2011). Efficient and principled method for detecting communities in networks. *Physical Review E*, **84**(3).

Beal, M. J. and Ghahramani, Z. (2003). The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. *Bayesian Statistics*, **7**.

Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **35**(8), 1798–1828.

Bertin-Mahieux, T., Ellis, D. P., Whitman, B., and Lamere, P. (2011). The million song dataset. In *ISMIR*.

Bickel, P. and Doksum, K. (2007). *Mathematical Statistics: Basic Ideas and Selected Topics*, volume 1. Pearson Prentice Hall, Upper Saddle River, NJ, 2nd edition.

Bishop, C. (2006). *Pattern Recognition and Machine Learning*. Springer New York.

Blei, D. (2014). Build, compute, critique, repeat: Data analysis with latent variable models. *Annual Review of Statistics and Its Application*, **1**, 203–232.

Blei, D. and Lafferty, J. (2007). A correlated topic model of Science. *Annals of Applied Statistics*, **1**(1), 17–35.

Blei, D., Ng, A., and Jordan, M. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, **3**, 993–1022.

Bottou, L. and LeCun, Y. (2004). Large scale online learning. *Advances in Neural Information Processing Systems 16*.

Braun, M. and McAuliffe, J. (2010). Variational inference for large-scale models of discrete choice. *Journal of the American Statistical Association*, **105**(489), 324–335.

Broderick, T., Mackey, L., Paisley, J., and Jordan, M. I. (2013a). Combinatorial clustering and the beta negative binomial process. *arXiv preprint arXiv:1111.1802*.

Broderick, T., Boyd, N., Wibisono, A., Wilson, A. C., and Jordan, M. (2013b). Streaming variational bayes. In *Advances in Neural Information Processing Systems*, pages 1727–1735.

Bryant, M. and Sudderth, E. B. (2012). Truly nonparametric online variational inference for hierarchical Dirichlet processes. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2708–2716.

Buntine, W. (2002). Variational extentions to EM and multinomial PCA. In *European Conference on Machine Learning*.

Cann, H. M., de Toma, C., Cazes, L., Legrand, M. F., Morel, V., Piouffre, L., Bodmer, J., Bodmer, W. F., Bonne-Tamir, B., Cambon-Thomsen, A., Chen, Z., Chu, J., Carcassi, C., Contu, L., Du, R., Excoffier, L., Ferrara, G. B., Friedlaender, J. S., Groot, H., Gurwitz, D., Jenkins, T., Herrera, R. J., Huang, X., Kidd, J., Kidd, K. K., Langaney, A., Lin, A. A., Mehdi, S. Q., Parham, P., Piazza, A., Pistillo, M. P., Qian, Y., Shu, Q., Xu, J., Zhu, S., Weber, J. L., Greely, H. T., Feldman, M. W., Thomas, G., Dausset, J., and Cavalli-Sforza, L. L. (2002). A human genome diversity cell line panel. *Science*, **296**(5566), 261–262.

Canny, J. (2004). GaP: A factor model for discrete data. In *ACM SIGIR*.

Cavalli-Sforza, L. L. (2005). The Human Genome Diversity Project: Past, Present and Future. *Nat. Rev. Genet.*, **6**(4), 333–340.

Çinlar, E. (1975). *Introduction to stochastic processes*. Prentice-Hall, Englewood Cliffs, NJ.

Cemgil, A. T. (2009). Bayesian inference for nonnegative matrix factorisation models. *Computational Intelligence and Neuroscience*, **2009**.

Chen, P. and Redner, S. (2010). Community structure of the physical review citation network. *Journal of Informetrics*, **4**(3), 278–290.

Clauset, A., Newman, M. E. J., and Moore, C. (2004). Finding community structure in very large networks. *Physical Review E*, **70**(6), 066111.

Clauset, A., Shalizi, C. R., and Newman, M. E. (2009). Power-law distributions in empirical data. *SIAM review*, **51**(4), 661–703.

Davis, J. and Goadrich, M. (2006). The relationship between precision-recall and ROC curves. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 233–240, New York, NY, USA. ACM.

Decelle, A., Krzakala, F., Moore, C., and Zdeborová, L. (2011a). Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, **84**(6).

Decelle, A., Krzakala, F., Moore, C., and Zdeborová, L. (2011b). Inference and phase transitions in the detection of modules in sparse networks. *Physical Review Letters*, **107**(6), 65701.

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

Derényi, I., Palla, G., and Vicsek, T. (2005). Clique percolation in random networks. *Physical Review Letters*, **94**(16), 160202.

Dror, G., Koenigstein, N., and Koren, Y. (2012a). Web-scale media recommendation systems. *Proceedings of the IEEE*, **100**(9), 2722–2736.

Dror, G., Koenigstein, N., Koren, Y., and Weimer, M. (2012b). The Yahoo! music dataset and KDD-cup '11. *Journal of Machine Learning Research*, **18**, 8–18.

Dunson, D. B. and Herring, A. H. (2005). Bayesian latent variable models for mixed discrete outcomes. *Biostatistics*, **6**(1), 11–25.

Eckenhoff, J. B., Cortese, R., and Landrau, F. A. (1987). Self controlled release device for administering beneficial agent to recipient. United States patent US 4,692,336.

Efron, B. (2013). Empirical bayes modeling, computation, and accuracy.

Ferguson, T. S. (1973). A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*.

Feynman, R. P. (1972). *Statistical mechanics: a set of lectures*. W. A. Benjamin.

Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, **486**(35), 75–174.

Fruchterman, T. M. J. and Reingold, E. M. (1991). Graph drawing by force-directed placement. *Softw. Pract. Exper.*, **21**(11), 1129–1164.

Gantner, Z., Drumond, L., Freudenthaler, C., and Schmidt-Thieme, L. (2012). Bayesian personalized ranking for non-uniformly sampled items. *JMLR W&CP*.

Gaussier, E. and Goutte, C. (2005). Relation between PLSA and NMF and implications. In *ACM SIGIR*, pages 601–602.

Geisser, S. and Eddy, W. (1979). A predictive approach to model selection. *Journal of the American Statistical Association*, pages 153–160.

Gelman, A., Meng, X., and Stern, H. (1996). Posterior predictive assessment of model fitness via realized discrepancies. *Statistica Sinica*, **6**, 733–807.

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

Ghahramani, Z. and Beal, M. (2001). Propagation algorithms for variational Bayesian learning. In *Neural Information Processing Systems*, pages 507–513.

Ginsparg, P. (2011). Arxiv at 20. *Nature*, **476**(7359), 145–147.

Goldenberg, A., Zheng, A. X., Fienberg, S. E., and Airoldi, E. M. (2010). A survey of statistical network models. *Found. Trends Mach. Learn.*, **2**(2), 129–233.

Gopalan, P. K. and Blei, D. M. (2013). Efficient discovery of overlapping communities in massive networks. *Proceedings of the National Academy of Sciences*, **110**(36), 14534–14539.

Gore, R. W. (1976). Process for producing porous products. United States patent US 3,953,566.

Greene, W. (2005). Censored data and truncated distributions.

Gregory, S. (2010). Finding overlapping communities in networks by label propagation. *New Journal of Physics*, **12**(10), 103018.

Griffiths, T. L. and Ghahramani, Z. (2011). The Indian buffet process: an introduction and review. *Journal of Machine Learning Research*, **12**, 1185–1224.

Hastings, W. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, **57**, 97–109.

Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 230–237, New York, NY, USA. ACM.

Hoff, P., Raftery, A., and Handcock, M. (2002). Latent space approaches to social network analysis. *Journal of the American Statistical Association*, **97**(460), 1090–1098.

Hoffman, M. (2012). Poisson-uniform nonnegative matrix factorization. In *ICASSP*.

Hoffman, M., Blei, D., and Bach, F. (2010a). On-line learning for latent Dirichlet allocation. In *Neural Information Processing Systems*.

Hoffman, M., Blei, D., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *Journal of Machine Learning Research*, **14**(1303–1347).

Hoffman, M. D., Blei, D. M., and Cook, P. R. (2010b). Bayesian nonparametric matrix factorization for recorded music. In J. Frnkranz and T. Joachims, editors, *ICML*, pages 439–446. Omnipress.

Hong, L., Doumith, A. S., and Davison, B. D. (2013). Co-factorization machines: modeling user interests and predicting individual decisions in twitter. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 557–566. ACM.

Hu, Y., Koren, Y., and Volinsky, C. (2008). Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 263–272. IEEE.

Inouye, D., Ravikumar, P., and Dhillon, I. (2014). Admixture of Poisson MRFs: A topic model with word dependencies. In *ICML*, pages 683–691.

Jaakkola, T. and Jordan, M. (1996). A variational approach to Bayesian logistic regression models and their extensions. In *International Workshop on Artificial Intelligence and Statistics*.

Jack, Kris anwd Hammerton, J., Harvey, D., Hoyt, J. J., Reichelt, J., and Henning, V. (2010). Mendeleys reply to the datatel challenge. *Procedia Computer Science*, **1**(2), 1–3.

Jeong, H., Nda, Z., and Barabsi, A. L. (2003). Measuring preferential attachment in evolving networks. *EPL (Europhysics Letters)*, **61**(4), 567.

Johnson, N., Kemp, A., and Kotz, S. (2005). *Univariate Discrete Distributions*. John Wiley & Sons.

Jordan, M., Ghahramani, Z., Jaakkola, T., and Saul, L. (1999). Introduction to variational methods for graphical models. *Machine Learning*, **37**, 183–233.

Jordan, M. I. (2011). The era of big data. *The ISBA Bulletin*, **18**(2).

Karrer, B. and Newman, M. E. J. (2011). Stochastic blockmodels and community structure in networks. *Phys. Rev. E*, **83**, 016107.

Khoury, J., Ovrut, B. A., Steinhardt, P. J., and Turok, N. (2001). Ekpyrotic universe: Colliding branes and the origin of the hot big bang. *Physical Review D*, **64**(12), 123522.

Kim, D., Voelker, G., and Saul, L. (2012). A variational approximation for topic modeling of hierarchical corpora.

Kim, D. I., Gopalan, P., Blei, D. M., and Sudderth, E. B. (2013). Efficient online inference for bayesian nonparametric relational models. In *Neural Information Processing Systems*.

Kim, M. and Leskovec, J. (2011). Modeling social networks with node attributes using the multiplicative attribute graph model. In *Uncertainty in Artificial Intelligence*.

Knowles, D. A. and Ghahramani, Z. (2011). Nonparametric Bayesian sparse factor models with application to gene expression modelling. *Annals of Applied Statistics*, **5**(2B), 1534–1552.

Koren, Y. (2008). Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *ACM SIGKDD*, pages 426–434. ACM.

Koren, Y., Bell, R., and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, **42**(8), 30–37.

Krivitsky, P., Handcock, M., Raftery, A., and Hoff, P. (2009). Representing degree distributions, clustering, and homophily in social networks with latent cluster random effects models. *Social Networks*, **31**, 204–213.

Kullback, S. and Leibler, R. (1951). On information and sufficiency. *The Annals of Mathematical Statistics*, **22**(1), 79–86.

Kurihara, K., Welling, M., and Vlassis, N. (2006). Accelerated variational Dirichlet process mixtures. In *Advances in Neural Information Processing Systems (NIPS)*.

Kushner, H. and Yin, G. (1997). *Stochastic Approximation Algorithms and Applications*. Springer New York.

Lancichinetti, A. and Fortunato, S. (2009). Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical Review E*, **80**(1).

Lancichinetti, A., Radicchi, F., Ramasco, J. J., and Fortunato, S. (2011). Finding statistically significant communities in networks. *PLoS ONE*, **6**(4), e18961.

Lee, D. and Seung, H. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, **401**(6755), 788–791.

Leskovec, J., Kleinberg, J., and Faloutsos, C. (2005). Graphs over time: Densification laws, shrinking diameters and possible explanations. In *KDD*.

Leskovec, J., Lang, K. J., Dasgupta, A., and Mahone, M. W. (2008). Community structure in large networks: Natural cluster sizes and the absence of large well-defined cluster. In *Internet Mathematics*.

Leskovec, J., Lang, K. J., Dasgupta, A., and Mahoney, M. W. (2009). Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, **6**(1), 29–123.

Liang, P., Petrov, S., Jordan, M. I., and Klein, D. (2007). The infinite PCFG using hierarchical Dirichlet processes. In *Empirical Methods in Natural Language Processing*, pages 688–697.

Liben-Nowell, D. and Kleinberg, J. (2003). The link prediction problem for social networks. In *Proceedings of the twelfth international conference on Information and knowledge management*, CIKM '03, pages 556–559, New York, NY, USA. ACM.

Ma, H., Liu, C., King, I., and Lyu, M. R. (2011). Probabilistic factor models for web site recommendation. In *ACM SIGIR*, pages 265–274. ACM Press.

MacKay, D. (1992). Information-based objective functions for active data selection. *Neural computation*, **4**(4), 590–604.

Marlin, B., Zemel, R. S., Roweis, S., and Slaney, M. (2012). Collaborative filtering and the missing at random assumption. *arXiv preprint arXiv:1206.5267*.

Marlin, B. M. and Zemel, R. S. (2009). Collaborative prediction and ranking with non-random missing data. pages 5–12.

McCullagh, P. and Nelder, J. A. (1989). *Generalized Linear Models, Second Edition*. Chapman and Hall/CRC, 2 edition.

McDaid, A. F. and Hurley, N. J. (2010). Using model-based overlapping seed expansion to detect highly overlapping community structure. *arXiv:1011.1970*.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, M., and Teller, E. (1953). Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, **21**, 1087–1092.

Miller, K. (2011). *Bayesian Nonparametric Latent Feature Models*. Ph.D. thesis, EECS Department, University of California, Berkeley, CA. UCB/EECS-2011-78.

Neal, R. (1993). Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto.

Nepusz, T., Petróczi, A., Négyessy, L., and Bazsó, F. (2008). Fuzzy communities and the concept of bridgeness in complex networks. *Physical Review E*, **77**(1), 016107.

Newman, M. (2003). The structure and function of complex networks. *SIAM Review*, **45**, 167–256.

Newman, M. E. J. (2002). Assortative mixing in networks. *Physical Review Letters*, **89**(20), 208701.

Newman, M. E. J. (2006). Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, **74**(3), 036104.

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

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

Padmanabhan, T. (2003). Cosmological constantthe weight of the vacuum. *Physics Reports*, **380**(56), 235–320.

Papadopoulos, F., Kitsak, M., Serrano, M. ., Bogu, M., and Krioukov, D. (2012). Popularity versus similarity in growing networks. *Nature*, **489**(7417), 537–540.

Paquet, U. and Koenigstein, N. (2013). One-class collaborative filtering with random graphs. In *WWW*.

Porteous, I., Bart, E., and Welling, M. (2008). Multi-HDP: A nonparametric Bayesian model for tensor factorization. In D. Fox and C. P. Gomes, editors, *AAAI*, pages 1487–1490. AAAI Press.

Porteous, I., Asuncion, A. U., and Welling, M. (2010). Bayesian matrix factorization with side information and dirichlet process mixtures. In M. Fox and D. Poole, editors, *AAAI*. AAAI Press.

Pritchard, J., Stephens, M., and Donnelly, P. (2000). Inference of population structure using multilocus genotype data. *Genetics*, **155**, 945–959.

Purcell, S., Neale, B., Todd-Brown, K., Thomas, L., Ferreira, M. A., Bender, D., Maller, J., Sklar, P., De Bakker, P. I., Daly, M. J., *et al.* (2007). Plink: a tool set for whole-genome association and population-based linkage analyses. *The American Journal of Human Genetics*, **81**(3), 559–575.

Raj, A., Stephens, M., and Pritchard, J. (2013). Variational inference of population structure in large SNP datasets. *bioRxiv*, (doi:10.1101/001073).

Randall, L. and Sundrum, R. (1999). An alternative to compactification. *Physical Review Letters*, **83**(23), 4690–4693.

Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. (2009a). BPR: Bayesian personalized ranking from implicit feedback. In *Uncertainty in Artificial Intelligence*, pages 452–461.

Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. (2009b). BPR: Bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461.

RITA (2010). U.S. Air Carrier Traffic Statistics, Bur. Trans. Stats.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, **22**(3), 400–407.

Robert, C. and Casella, G. (2004). *Monte Carlo Statistical Methods*. Springer Texts in Statistics. Springer-Verlag, New York, NY.

Rosenberg, N. A., Mahajan, S., Gonzalez-Quevedo, C., Blum, M. G., Nino-Rosales, L., Ninis, V., Das, P., Hegde, M., Molinari, L., Zapata, G., *et al.* (2006). Low levels of genetic divergence across geographically and linguistically diverse populations from india. *PLoS genetics*, **2**(12), e215.

Rubin, D. (1984). Bayesianly justifiable and relevant frequency calculations for the applied statistician. *The Annals of Statistics*, **12**(4), 1151–1172.

Salakhutdinov, R. and Mnih, A. (2008a). Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *ICML*, pages 880–887. ACM.

Salakhutdinov, R. and Mnih, A. (2008b). Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 880–887, New York, NY, USA. ACM.

Salakhutdinov, R. and Mnih, A. (2008c). Probabilistic matrix factorization. *Advances in Neural Information Processing Systems*, **20**, 1257–1264.

Salakhutdinov, R. and Mnih, A. (2008d). Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, volume 20.

Sato, I. and Nakagawa, H. (2010). Topic models with power-law using Pitman-Yor process. In *Knowledge Discovery and Data Mining*.

Schlegel, D. J., Finkbeiner, D. P., and Davis, M. (1998). Maps of dust infrared emission for use in estimation of reddening and cosmic microwave background radiation foregrounds. *The Astrophysical Journal*, **500**(2), 525–553.

Shan, H. and Banerjee, A. (2010). Generalized probabilistic matrix factorizations for collaborative filtering. In *Proceedings of the 2010 IEEE International Conference on Data Mining*, pages 1025–1030, Washington, DC, USA. IEEE Computer Society.

Singh, A. P. and Gordon, G. J. (2008). Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 650–658. ACM.

Spall, J. (2003). *Introduction to stochastic search and optimization: Estimation, simulation, and control*. John Wiley and Sons.

Teh, Y. W. and Jordan, M. I. (2010). Hierarchical Bayesian nonparametric models with applications. In N. Hjort, C. Holmes, P. Müller, and S. Walker, editors, *Bayesian Nonparametrics: Principles and Practice*. Cambridge University Press.

Thibaux, R. (2008). *Nonparametric Bayesian Models for Machine Learning*. Ph.D. thesis, EECS Department, University of California, Berkeley, CA. UCB/EECS-2008-130.

Tierney, L., Kass, R., and Kadane, J. (1989). Fully exponential Laplace approximations to expectations and variances of nonpositive functions. *Journal of American Statistical Association*, **84**(407).

Titsias, M. (2007). The infinite gamma-Poisson feature model. *Advances in Neural Information Processing Systems (NIPS)*, **19**.

Trevio, S., Sun, Y., Cooper, T. F., and Bassler, K. E. (2012). Robust detection of hierarchical communities from escherichia coli gene expression data. *PLoS Comput Biol*, **8**(2), e1002391.

Viamontes Esquivel, A. and Rosvall, M. (2011). Compression of flow can reveal overlapping-module organization in networks. *Phys. Rev. X*, **1**, 021025.

Wainwright, M. and Jordan, M. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, **1**(1–2), 1–305.

Wang, C. and Blei, D. (2011). Collaborative topic modeling for recommending scientific articles. In *Knowledge Discovery and Data Mining*.

Wang, C. and Blei, D. (2013). Variational inference in nonconjugate models. *Journal of Machine Learning Research*, **14**, 1005–1031.

Wang, Y. and Wong, G. (1987). Stochastic block models for directed graphs. *Journal of the American Statistical Association*, **82**(397), 8–19.

Weinberger, K., Dasgupta, A., Langford, J., Smola, A., and Attenberg, J. (2009). Feature hashing for large scale multitask learning. In *ICML*, pages 1113–1120.

Weir, B. S. and Cockerham, C. C. (1984). Estimating F-statistics for the analysis of population structure. *Evolution*, pages 1358–1370.

Wiggins, C. and Hofman, J. (2008). Bayesian approach to network modularity. *Physical Review Letters*, **100**(25).

Will, C. M. (2006). The confrontation between general relativity and experiment. *Living Rev. Relativity*, **9**.

Zhou, M. and Carin, L. (2013). Negative binomial process count and mixture modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **99**, 1.

Zhou, M., Hannah, L., Dunson, D. B., and Carin, L. (2012). Beta-negative binomial process and Poisson factor analysis. *Journal of Machine Learning Research - Proceedings Track*, **22**, 1462–1471.