Latent Gaussian Models for Topic Modeling

Changwei Hu **Eunsu Ryu** **David Carlson** **Yingjian Wang** **Lawrence Carin**
Department of Electrical and Computer Engineering, Duke University, Durham NC 27708, USA

Abstract

A new approach is proposed for topic modeling, in which the latent matrix factorization employs Gaussian priors, rather than the Dirichlet-class priors widely used in such models. The use of a latent-Gaussian model permits simple and efficient approximate Bayesian posterior inference, via the Laplace approximation. On multiple datasets, the proposed approach is demonstrated to yield results as accurate as state-of-the-art approaches based on Dirichlet constructions, at a small fraction of the computation. The framework is general enough to jointly model text and binary data, here demonstrated to produce accurate and fast results for joint analysis of voting rolls and the associated legislative text. Further, it is demonstrated how the technique may be scaled up to massive data, with encouraging performance relative to alternative methods.

1 INTRODUCTION

We are interested in analyzing matrices of the form $\mathbf{C} \in \mathbb{Z}_+^{p\times n}$, where $\mathbb{Z}_+$ represents nonnegative integers (*i.e.*, counts). In many applications we have a count matrix $\mathbf{C} \in \mathbb{Z}_+^{p_1\times n}$ *and* an associated binary matrix $\mathbf{B} \in \{0,1\}^{p_2\times n}$, with an interest in analyzing them jointly. For example, one may analyze the binary votes of p_2 legislators on n pieces of legislation (Gerrish & Blei, 2011; Zhang & Carin, 2012), where the text of the legislation is in terms of p_1 words. In this setting one may wish to predict missing votes from $\mathbf{B}$ and, often more importantly, in predicting the votes of the p_2 legislators on a new piece of legislation, based upon counts of the p_1 words in the associated text of that legislation (predicting a new column of $\mathbf{B}$, based on observation of the associated column of $\mathbf{C}$).

Appearing in Proceedings of the 17^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2014, Reykjavik, Iceland. JMLR: W&CP volume 33.

In the context of topic modeling, the number of documents one may wish to analyze may be very large. There is consequently also increasing interest in developing topic-modeling techniques that scale up efficiently to a very large corpus of documents (Hoffman et al., 2010).

Virtually all models of documents, as well as *joint* analysis of text with auxiliary data (such as a binary matrix as above), are based on generalizations of topic models. Let $\mathbf{C} \in \mathbb{Z}_+^{p\times n}$ represent a corpus of n documents, with a p-word vocabulary. Topic models implicitly or explicitly perform a form of latent matrix factorization, expressed in general as

$$\mathbf{X} = \mathbf{D}\mathbf{S}^T \tag{1}$$

where $\mathbf{X}$ is a $p \times n$ matrix, and X_{ij} represents component (i, j) of this matrix. Typically it is assumed that $X_{ij} \geq 0$ for all (i, j).

In latent Dirichlet allocation (LDA) (Blei et al., 2003) and related topic models, it is assumed that $\sum_{i=1}^{p} X_{ij} = 1$ for all j. Here column j of $\mathbf{X}$ represents the probability with which each of the p words are manifested in document j. Letting $\boldsymbol{c}_j$ represent column j of $\mathbf{C}$, then LDA imposes $\boldsymbol{c}_j \sim \text{Mult}(|\boldsymbol{c}_j|, \boldsymbol{x}_j)$, where $\boldsymbol{x}_j$ is column j of $\mathbf{X}$, and $|\boldsymbol{c}_j| = \sum_{i=1}^{p} C_{ij}$ is the total number of words in document j.

In LDA, the factorization in (1) is defined by placing a Dirichlet prior on the K columns of $\mathbf{D}$, and a separate Dirichlet prior on the n rows of $\mathbf{S}$, for K latent topics. In recently developed topic models, like the nested Chinese restaurant process (nCRP)(Blei et al., 2010; Wang & Blei, 2009) and the focused topic model (FTM) (Williamson et al., 2009), the means by which the columns of $\mathbf{D}$ and rows of $\mathbf{S}$ are manifested are more complicated, but in each case the basic construct involves generalized usage of Dirichlet distributions.

In an alternative approach, the documents are modeled as $C_{ij} \sim \text{Pois}(X_{ij})$. Often a Dirichlet prior is again imposed on the columns of $\mathbf{D}$, and a gamma prior is typically placed on the components of $\mathbf{S}$ (M. Zhou & Carin, 2012). This Poisson-based model has been associated with state-of-the-art topic models, notably the FTM (Williamson et al., 2009). In

(M. Zhou & Carin, 2012) the authors make an extensive linkage between most topic models based on Dirichlet (mixture model) constructions and those based upon the Poisson setup.

In the above models, document j is characterized by $\boldsymbol{s}_j$, the transpose of which is the jth *row* of $\mathbf{S}$; $\boldsymbol{s}_j$ characterizes the strength with which each of the K topics is responsible for the words in document j. The probability of word usage for topic k is defined by column k of $\mathbf{D}$. When jointly modeling documents and a matrix like the binary matrix $\mathbf{B}$ discussed above (Gerrish & Blei, 2011; Zhang & Carin, 2012), $\boldsymbol{s}_j$ is typically also employed to model the binary entries in column j of $\mathbf{B}$. Therefore, the latent $\boldsymbol{s}_j$ is *shared* between the model of the text and the associated binary matrix, linking the two.

In the above-summarized models, which characterize the basic properties of virtually all topic models employed in the literature, the use of priors like the Dirichlet and gamma distributions complicate inference. While one may often derive MCMC (M. Zhou & Carin, 2012), VB (Beal, 2003) and EP (Minka, 2001) update equations with analytic update distributions, the computation time needed for such approaches can be significant in many applications.

In this paper we develop a new topic model based upon latent Gaussian priors. Approximate Bayesian posterior inference can be done accurately and efficiently via the Laplace approximation. We leverage ideas from the recently developed integrated nested Laplace approximation (INLA) (Rue et al., 2009) to make our method efficient to implement. INLA was originally developed for *linear* latent Gaussian models, and here we extend these ideas to latent matrix factorization, and for the joint analysis of text and auxiliary matrices, like associated binary data. The key thing the model exploits is that while the latent parameters with Gaussian priors may be very high-dimensional, the number of hyperparameters is typically small (here of dimension one or two). This property is leveraged in INLA, and it is utilized here in a new way in the context of topic modeling. It is demonstrated that the proposed simple and computationally efficient approach yields results as accurate as the most advanced topic models in the literature. We also demonstrate how the model may be scaled to consider data of massive dimension.

When computing the maximum *a posterior* (MAP) estimate of the parameters within the Laplace approximation, we are solving an optimization problem. This is attractive, in that it allows us to leverage many methods from the optimization literature (Jenatton et al., 2010; Bittorf et al., 2012; Lee & Seung, 2000; Boyd et al., 2010; Lin, 2007) when computing the MAP, thereby effectively integrating ideas from the Bayesian and optimization communities. Additionally, this approach provides very simple and efficient parallization of the computations, yielding further acceleration.

2 LATENT GAUSSIAN MODEL

2.1 Gaussian Priors

Let $\boldsymbol{d}_{1i} \in \mathbb{R}^K$ represent a column vector, the transpose of which corresponds to row i in $\mathbf{D}_1$; we are assuming a K-factor model (equivalent to K topics). We use a subscript 1 on $\mathbf{D}$ from (1), in anticipation of also modeling an associated binary matrix, with its own related matrix $\mathbf{D}_2$. Element (i,j) of $\mathbf{X}_1 = \mathbf{D}_1\mathbf{S}^T$ in (1) is $X_{1,ij} = \boldsymbol{d}_{1i}^T\boldsymbol{s}_j$.

We have considered two means of modeling count data, the first based on a probit link. The ordinal probit model, which we denote it as latent Gaussian ordinal probit model (LG-OP), is expressed as

$$C_{ij} = h \text{ if } c_h \leq g_{ij} < c_{h+1}, \quad g_{ij} \sim \mathcal{N}(X_{1,ij}, 1) \quad (2)$$

where $c_0, c_1, \ldots$ are "cut points," with $c_0 = -\infty$, $c_1 = 0$, $c_2 = 1$, $c_3 = 2$, etc. The model in (2) is widely applied for binary data, with $h = 0$ or $h = 1$, and $c_0 = -\infty$, $c_1 = 0$ and $c_2 = \infty$; here we extend it to analysis of count data from a document corpus, with h a nonnegative integer.

To our knowledge, a model like (2) has not been considered previously for topic modeling. However, related models have been considered in which the binning reflected in (2) is avoided, and $\mathbf{C}$ is simply treated as a nonnegative *real* matrix. Dictionary learning and related methods (Jenatton et al., 2010; Bittorf et al., 2012; Lee & Seung, 2000; Boyd et al., 2010; Lin, 2007) have been applied for the analysis of such matrices.

So motivated, another means of modeling the count data, which is anologous to the ordinal probit model, employs $C_{ij} \sim \mathcal{N}(X_{1,ij}, 1)$, with the requirement that the elements of $\mathbf{D}_1$ and $\mathbf{S}$ are nonnegative. This model imposes that $X_{1,ij}$ typically has unit variance from the observed count C_{ij}, analogous to the ordinal probit model.

In the experiments of Section 5, results are primarily (but not exclusively) presented based the second method of modeling counts, which we term LG-NMF due to its close connection to nonnegative matrix factorization. We therefore consider that method when presenting priors for $\{\boldsymbol{d}_{1i}\}$ and $\{\boldsymbol{s}_j\}$. Specifically, we impose

$$\boldsymbol{d}_{1i} \sim \mathcal{N}_+(0, \frac{1}{K}\mathbf{I}_K), \quad \boldsymbol{s}_j \sim \mathcal{N}_+(0, \alpha_1^{-1}\mathbf{I}_K) \quad (3)$$

where $\mathbf{I}_K$ represents the $K \times K$ identity matrix, $\alpha_1 \in \mathbb{R}_+$, and $\mathcal{N}_+(\cdot,\cdot)$ is the truncated normal distribution (restricted to be positive). We impose a weak gamma prior on α_1 (*i.e.*, $\text{Ga}(10^{-6}, 10^{-6})$). For LG-OP we simply use Gaussian priors, without the positivity constraint.

The above priors are similar to those in the work of (Salakhutdinov & Mnih, 2008; Silva & Carin, 2012), except there topic modeling was not considered. In Jenatton et al. (2010), a related model was considered for topic modeling, but an optimization solution was considered. As discussed below, the proposed Bayesian solution leverages optimization approaches like that considered in Jenatton et al. (2010).

2.2 Joint Analysis of Text and Binary Matrix

If we have an accompanying binary matrix $\mathbf{B} \in \{0,1\}^{p_2 \times n}$, we assume an associated latent matrix

$$\mathbf{X}_2 = \mathbf{D}_2 \mathbf{S}^T \tag{4}$$

where $\mathbf{D}_2 \in \mathbb{R}^{p_2 \times K}$ and $\mathbf{S} \in \mathbb{R}^{n \times K}$ is the *same* matrix as used in (1) for representation of the text data $\mathbf{C}$. Therefore, $\mathbf{S}$ is *shared* between the model of the text and binary matrix. With $\boldsymbol{d}_{2i} \in \mathbb{R}^K$ representing the transpose of row i of $\mathbf{D}_2$, we impose the prior

$$\boldsymbol{d}_{2i} \sim \mathcal{N}(0, \alpha_2^{-1} \mathbf{I}_K) \tag{5}$$

The link between $\mathbf{X}_2$ and $\mathbf{B}$ is the probit link function as in (2), modified for two observations, $h = 0$ or $h = 1$. We again place a weak gamma hyperprior on α_2.

2.3 Model Characteristics

The above model is defined by many vectors $\{\boldsymbol{d}_{1i}\}$, $\{\boldsymbol{d}_{2i}\}$ and $\{\boldsymbol{s}_j\}$, but only two hyperparameters $\boldsymbol{\theta} = (\alpha_1, \alpha_2)^T$. When we only model text, or only model a binary matrix, there is only one hyperparameter, $\theta = \alpha_1$ *or* $\theta = \alpha_2$. The fact that each of the many vectors that we model has a Gaussian prior, and we only have a very small set of hyperparameters, significantly simplifies inference, yielding a procedure closely connected to the recently developed INLA algorithm (Rue et al., 2009).

3 APPROXIMATE INFERENCE

3.1 MAP Estimate

We consider the joint analysis of a count matrix $\mathbf{C}$ and an associated binary matrix $\mathbf{B}$; the method simplifies when only $\mathbf{C}$ or $\mathbf{B}$ is analyzed. The negative log of the posterior is

$$\begin{aligned} &-\log p(\{\boldsymbol{d}_{1i}\}, \{\boldsymbol{d}_{2i}\}, \{\boldsymbol{s}_j\}, \alpha_1, \alpha_2 | \mathbf{C}, \mathbf{B}) \\ &= \textstyle\sum_{i=1}^{p_1} \sum_{j=1}^{n} (C_{ij} - \boldsymbol{d}_{1i}^T \boldsymbol{s}_j)^2 \\ &- \textstyle\sum_{i=1}^{p_2} \sum_{j=1}^{n} \log \Psi(B_{ij}, \boldsymbol{d}_{2i}^T \boldsymbol{s}_j) \\ &+ \tfrac{\alpha_1}{2} \textstyle\sum_{j=1}^{n} \|\boldsymbol{s}_j\|_2^2 + \tfrac{K}{2} \sum_{i=1}^{p_1} \|\boldsymbol{d}_{1i}\|_2^2 \\ &+ \tfrac{\alpha_2}{2} \textstyle\sum_{i=1}^{p_2} \|\boldsymbol{d}_{2i}\|_2^2 + f(\alpha_1, \alpha_2) + \text{const} \end{aligned} \tag{6}$$

where $\Psi(B_{ij}, \boldsymbol{d}_{2i}^T \boldsymbol{s}_j) = [F(\boldsymbol{d}_{2i}^T \boldsymbol{s}_j)]^{B_{ij}} [1 - F(\boldsymbol{d}_{2i}^T \boldsymbol{s}_j)]^{1 - B_{ij}}$, with $F(x) = \int_{-\infty}^{x} \mathcal{N}(\beta; 0, 1) d\beta$.

For fixed values of the hyperparameters $\boldsymbol{\theta} = (\alpha_1, \alpha_2)^T$, one may estimate *point* values of $\{\boldsymbol{d}_{1i}\}$, $\{\boldsymbol{d}_{2i}\}$, and $\{\boldsymbol{s}_j\}$, and in fact this problem is very similar to that considered in the optimization literature Jenatton et al. (2010). Specifically, we have implemented the following iterative procedure. With two of the sets of vectors $\{\boldsymbol{d}_{1i}\}$, $\{\boldsymbol{d}_{2i}\}$ and $\{\boldsymbol{s}_j\}$ held fixed, we perform a gradient optimization for the third set of vectors. We sequentially cycle through the three sets of vectors, optimizing one set of vectors while the other two are fixed. While there are not guarantees of a globally optimal solution, this iterative approach is assured to converge, and is widely employed (Jenatton et al., 2010; Chan & Wong, 2000).

Gradient descent is used to update a given set of parameters, and many methods exist in the literature for this (see Jenatton et al. (2010) and the references therein). For brevity we omit details here, other than to note that the gradients for the representation in (6) may be expressed analytically. It is important to emphasize that the proposed Bayesian inference method has the salutary property of being able to leverage many of the optimization methods that have been developed in the literature Recht (2011); Candès & Recht (2012); Abernethy et al. (2009); Boyd et al. (2010), for estimation of the MAP with fixed $\boldsymbol{\theta}$. This provides computational speed, and will also allow the model to scale to large problems, as discussed in Section 4. Additionally, for each setting of hyperparameters $\boldsymbol{\theta}$, these optimization problems may be performed in parallel.

3.2 Approximate Hyperparameter Posterior

Let $\boldsymbol{x}$ be a vector that represents a concatenation of all vectors $\{\boldsymbol{d}_{1i}\}$, $\{\boldsymbol{d}_{2i}\}$ and $\{\boldsymbol{s}_j\}$ on which Gaussian priors are employed, and again the small vector $\boldsymbol{\theta}$ represents the hyperparameters. Then the posterior distribution on the hyperparameters is

$$p(\boldsymbol{\theta}|\mathbf{C}, \mathbf{B}) = \frac{p(\boldsymbol{x}, \boldsymbol{\theta}|\mathbf{C}, \mathbf{B})}{p(\boldsymbol{x}|\boldsymbol{\theta}, \mathbf{C}, \mathbf{B})} \propto \frac{p(\boldsymbol{x}, \boldsymbol{\theta}, \mathbf{C}, \mathbf{B})}{p(\boldsymbol{x}|\boldsymbol{\theta}, \mathbf{C}, \mathbf{B})} \tag{7}$$

with proportionality constant $p(\mathbf{C}, \mathbf{B})$.

We have an explicit expression for $p(\boldsymbol{x}, \boldsymbol{\theta}, \mathbf{C}, \mathbf{B})$, and for any fixed setting of the hyperparameters $\boldsymbol{\theta}$, we may construct an approximation for $p(\boldsymbol{x}|\boldsymbol{\theta}, \mathbf{C}, \mathbf{B})$. Specifically, since we have a Gaussian prior on $\boldsymbol{x}$, it is reasonable to approximate $p(\boldsymbol{x}|\boldsymbol{\theta}, \mathbf{C}, \mathbf{B})$ as Gaussian, $p(\boldsymbol{x}|\boldsymbol{\theta}, \mathbf{C}, \mathbf{B}) \approx p_G(\boldsymbol{x}|\boldsymbol{\theta}, \mathbf{C}, \mathbf{B})$, yielding the Laplace approximation

$$\tilde{p}(\boldsymbol{\theta}|\mathbf{C}, \mathbf{B}) \propto \frac{p(\boldsymbol{x}, \boldsymbol{\theta}, \mathbf{C}, \mathbf{B})}{p_G(\boldsymbol{x}|\boldsymbol{\theta}, \mathbf{C}, \mathbf{B})}_{|\boldsymbol{x}=\boldsymbol{x}^*(\boldsymbol{\theta})} \tag{8}$$

where $\boldsymbol{x}^*(\boldsymbol{\theta})$ represents the approximate MAP solution for hyperparameters $\boldsymbol{\theta}$, computed as discussed in Section 3.1. We use the symbol $\tilde{p}(\boldsymbol{\theta}|\mathbf{C},\mathbf{B})$ to emphasize that this is an approximation; note that while $p(\boldsymbol{x},\boldsymbol{\theta},\mathbf{C},\mathbf{B})$ is approximated as Gaussian, $\tilde{p}(\boldsymbol{\theta}|\mathbf{C},\mathbf{B})$ is generally *not* Gaussian.

We represent the Gaussian approximation as $p_G(\boldsymbol{x}|\boldsymbol{\theta},\mathbf{C},\mathbf{B}) = \mathcal{N}(\boldsymbol{x}^*(\boldsymbol{\theta}),\boldsymbol{\Sigma}(\boldsymbol{\theta},\boldsymbol{x}^*(\boldsymbol{\theta})))$,

$$\boldsymbol{\Sigma}(\boldsymbol{\theta},\boldsymbol{d}_{1i}^*) = K\mathbf{I}_K + \mathbf{S}^T\mathbf{S} \quad (9)$$

$$\boldsymbol{\Sigma}(\boldsymbol{\theta},\boldsymbol{s}_j^*) = \alpha_1\mathbf{I}_K + \mathbf{D}_1^T\mathbf{D}_1 - \frac{\partial \ell_{\mathbf{B}}}{\partial \boldsymbol{s}_j \boldsymbol{s}_j^T} \quad (10)$$

$$\boldsymbol{\Sigma}(\boldsymbol{\theta},\boldsymbol{d}_{2i}^*) = \alpha_2\mathbf{I}_K - \frac{\partial \ell_{\mathbf{B}}}{\partial \boldsymbol{d}_{2i} \boldsymbol{d}_{2i}^T} \quad (11)$$

where

$$\ell_{\mathbf{B}} = \sum_{i=1}^{p_2}\sum_{j=1}^{n}[B_{ij}\log F(\boldsymbol{d}_{2i}^T\boldsymbol{s}_j) + (1-B_{ij})\log(1-F(\boldsymbol{d}_{2i}^T\boldsymbol{s}_j))] \quad (12)$$

All derivatives in (10) and (11) may be computed analytically.

Note that in the Gaussian approximation, the mean $\boldsymbol{x}^*(\boldsymbol{\theta})$ estimates all $\boldsymbol{x}^*(\boldsymbol{\theta})$ parameters jointly, while the covariance is factorized between the different associated vector parameters, for simplicity.

3.3 Making Predictions

Assume that $\{\boldsymbol{\theta}_m\}_{m=1,M}$ represent a discrete set of hyperparameters at which $\tilde{p}(\boldsymbol{\theta}_m|\mathbf{C},\mathbf{B})$ is computed; the normalization constant is readily inferred by imposing $\sum_{m=1}^{M}\Delta_m\tilde{p}(\boldsymbol{\theta}_m|\mathbf{C},\mathbf{B}) = 1$, where Δ_m represents the gridding rate of hyperparameter space. Assume for example that we wish to infer missing entry (i,j) in $\mathbf{B}$, $B_{ij} \in \{0,1\}$, then

$$p(B_{ij}|\mathcal{D}) \approx \textstyle\sum_m \Delta_m\tilde{p}(\boldsymbol{\theta}_m|\mathcal{D})p_m(B_{ij}|\mathcal{D}) \quad (13)$$
$$p_m(B_{ij}|\mathcal{D}) = \textstyle\int d\boldsymbol{d}_{2i}\int d\boldsymbol{s}_j p(\boldsymbol{d}_{2i},\boldsymbol{s}_j|\mathcal{D},\boldsymbol{\theta}_m)p(B_{ij}|\boldsymbol{d}_{2i},\boldsymbol{s}_j)$$

where $p(B_{ij}|\boldsymbol{d}_{2i},\boldsymbol{s}_j)$ is defined by the binary probit link function(Albert & Chib, 1993), and $\mathcal{D}$ represents the observed portion of $\mathbf{C}$ and/or $\mathbf{B}$.

When computing metrics like preplexity on held-out documents, a MAP estimate of $\mathbf{D}_1$ is learned for each $\boldsymbol{\theta}_m$ based upon the training data (characteristic of the K topics), and after inferring a posterior on $\mathbf{S}$ for the test data, we may a computation like in (13), but the nonnegative topic model is used, rather than the binary one.

What remains for computation of (13) is an approximation for $p(\boldsymbol{d}_{2i},\boldsymbol{s}_j|\mathcal{D},\boldsymbol{\theta}_m)$. Several options immediately suggest themselves. Recall that in constituting the approximate posterior for $\boldsymbol{\theta}$, the employed Laplace approximation used a Gaussian approximation for expressions like $p(\boldsymbol{d}_{2i},\boldsymbol{s}_j|\mathcal{D},\boldsymbol{\theta}_m)$; that same Gaussian approximation may be used here. Alternatively, a more accurate approximation may be constituted by making another Laplace approximation, now on $p(\boldsymbol{d}_{2i},\boldsymbol{s}_j|\mathcal{D},\boldsymbol{\theta}_m)$. If we were to do this, we would be making the Laplace approximation twice: once for approximating $p(\boldsymbol{\theta}|\mathcal{D})$, and another for $p(\boldsymbol{d}_{1i},\boldsymbol{d}_{2i},\boldsymbol{s}_j|\mathcal{D},\boldsymbol{\theta})$. This latter approach corresponds to the integrated *nested* Laplace approximation (INLA) developed by Rue et al. (2009). Note however that to date INLA has only been applied to *linear* models, not the *nonlinear* model proposed here, which involves products of vectors in the sets $\{\boldsymbol{d}_{1i}\}$, $\{\boldsymbol{d}_{2i}\}$, and $\{\boldsymbol{s}_j\}$.

The simplest approximation in (13) is to let $p_m(B_{ij}|\mathcal{D}) \approx p(B_{ij}|\boldsymbol{d}_{2i}^*(\boldsymbol{\theta}_m),\boldsymbol{s}_j^*(\boldsymbol{\theta}_m))$, which just corresponds to evaluating the probit link for parameters $(\boldsymbol{d}_{2i}^*(\boldsymbol{\theta}_m),\boldsymbol{s}_j(\boldsymbol{\theta}_m))$ corresponding to the MAP solution at $\boldsymbol{\theta}_m$. This is by far the simplest approach, and the excellent quality of our results mitigates against using either of the two more-complicated approximations discussed above. The utilized method reduces to a weighted average of a finite set of models, with each model defined by discrete $\boldsymbol{x}(\boldsymbol{\theta}_m)$, and the weights defined by $\Delta_m\tilde{p}(\boldsymbol{\theta}_m|\mathcal{D})$.

3.4 Gridding Hyperparameter Space

The proposed approach differs from INLA in two ways: (i) the matrix factorization is a nonlinear function of the vectors $\boldsymbol{x}$ on which Gaussian priors are imposed (INLA assumes a linear model); and (ii) we only employ one layer of Laplace approximation, on $p(\boldsymbol{\theta}|\mathcal{D})$, while INLA also imposes a Laplace approximation to $p(\boldsymbol{x}|\mathcal{D},\boldsymbol{\theta})$. However, the proposed model borrows a key component of INLA, on how to select the set $\{\boldsymbol{\theta}_m\}$. The reader is referred to Rue et al. (2009) for further details, and here we summarize the approach.

A gradient-descent algorithm is employed to infer the parameter $\boldsymbol{\theta}$ that maximizes (8). Once that parameter $\boldsymbol{\theta}^*$ is determined approximated, we use finite differences to approximate the Hessian, $\mathbf{H}$ (recall that, for our problem, $\boldsymbol{\theta}$ is two dimensional for joint analysis of text and a binary matrix, and it is only one dimensional when only considering text; so $\mathbf{H}$ is either a scalar precision, or a 2×2 precision matrix. Let $\boldsymbol{\Omega} = \mathbf{H}^{-1}$ approximate the covariance matrix of $\boldsymbol{\theta}$. We perform an SVD of $\boldsymbol{\Omega}$, and then grid up in a 2D or 1D grid, where in the 2D case the grid is Cartesian in directions defined by the principal vectors. The gridding size Δ_m is made small relative to the variance in the principal dimensions.

Note that once the grid of hyperparameters $\{\boldsymbol{\theta}_m\}_{m=1,M}$ is so defined, the means by which the associated $\{\boldsymbol{x}(\boldsymbol{\theta}_m)\}$ are computed reduces to an optimization problem when each $\boldsymbol{\theta}_m$ is fixed; the optimization problem is defined in (6). As discussed

in the next section, the fact that for each $\boldsymbol{\theta}_m$ we need solve an optimization problem allows the leveraging of a vast optimization literature, within a Bayesian solution. It is also important to note that solutions for $\boldsymbol{x}(\boldsymbol{\theta}_m)$ may be computed independently for each $\boldsymbol{\theta}_m$, on separate processors in parallel, manifesting significant computational speedup. By contrast, approximate Bayesian inference techniques like variational Bayesian (VB) and expectation propagation (EP) do not admit such simple parallelization. One may run parallel MCMC chains (Suchard et al., 2012). However, that is less relevant for the work considered here: for computational convenience and practical use, in machine learning one typically runs a relatively small number of MCMC samples, which must be done serially (discussed further when presenting results in Section 5).

4 SCALING UP

The large-scale datasets that we have access to are text-only (no binary matrix $\mathbf{B}$), and therefore we specialize this discussion to analysis of a count matrix $\mathbf{C}$. Similar techniques may be applied for joint analysis of large-scale $\mathbf{C}$ *and* $\mathbf{B}$.

For each discrete setting of hyperparameters $\boldsymbol{\theta}_m$, we wish to infer

$$\boldsymbol{x}^*(\boldsymbol{\theta}_m) = \text{argmax}_{\boldsymbol{x}(\boldsymbol{\theta}_m)} - \log p(\boldsymbol{x}, \boldsymbol{\theta}_m | \mathbf{C}) \qquad (14)$$

where $\boldsymbol{x}$ is a concatenation of $\{\boldsymbol{d}_{1i}\}$ and $\{\boldsymbol{s}_j\}$, on which we have imposed the Gaussian priors. This optimization is performed in parallel for each of the set of discrete $\{\boldsymbol{\theta}_m\}$.

We perform stochastic gradient descent (SGD) (Bottou, 1998; Mairal et al., 2009) for the learning of $\boldsymbol{x}(\boldsymbol{\theta})$, for each $\boldsymbol{\theta}$. The data batches and the sequence with which they are analyzed is the same for all $\{\boldsymbol{\theta}_m\}$. For each epoch $t \in \{1, ..., T\}$, we randomly permute the data set $\mathbf{C}$ and group the shuffled data into L mini-batches of size B (batches are defined by selecting a subset of documents, which corresponds to columns of $\mathbf{C}$).

Let $\mathbf{C}_b$ be a batch of documents, and let $\ell = -\log p(\{\boldsymbol{d}_{1i}\}, \{\boldsymbol{s}_j\}, \alpha_1 | \mathbf{C}_b)$. For each batch $\mathbf{C}_b$, we iteratively update $\mathbf{D}$ and $\mathbf{S}$ by $D_{1,ik}^{t+1} = D_{1,ik}^t - \eta \frac{\partial \ell}{\partial D_{1,ik}}$, and $S_{jk}^{t+1} = S_{jk}^t - \eta \frac{\partial \ell}{\partial S_{jk}}$, where η is the step size. We set $\eta = 0.003$ in our experiments. The gradient with respect to $D_{1,ik}$ and S_{jk} can be computed by

$$\frac{\partial \ell}{\partial D_{1,ik}} = \sum_{j=1}^{n} (\boldsymbol{d}_{1i}^T \boldsymbol{s}_j - C_{b,ij}) S_{jk} + K \boldsymbol{e}_{(p_1)}^T \boldsymbol{d}_{:k} \quad (15)$$

$$\frac{\partial \ell}{\partial S_{jk}} = \sum_{i=1}^{p_1} (\boldsymbol{d}_{1i}^T \boldsymbol{s}_j - C_{b,ij}) D_{1,ik} + \alpha_1 \boldsymbol{e}_{(n)}^T \boldsymbol{s}_{:k} \quad (16)$$

where $\boldsymbol{e}_{(p_1)}$ and $\boldsymbol{e}_{(n)}$ are $p_1 \times 1$ and $n \times 1$ column vectors with all entries being 1.

5 RESULTS

We examine the proposed latent Gaussian model with Laplace approximation on several data sets, with comparisons to some of the most recently developed topic-modeling alternatives. In some comparisons the LG model will be implemented with MCMC inference, and those results are denoted LG-MCMC (by comparing LG-NMF and LG-MCMC, the model is the same and the only thing that is different is the inference method; for long MCMC chains, LG-MCMC may be used as a reference for comparison). All computations were run on a computer with Intel i5 2.6 GHz processor with 4 GB RAM. For all models except one, the software was written in Matlab. The online LDA-VB model (Hoffman et al., 2010) was written in Python.

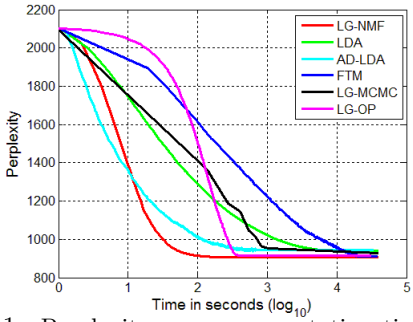

Figure 1: Perplexity versus computation time for LDA, AD-LDA, FTM, LG-NMF, LG-MCMC, and LG-OP, for the PsyRev dataset. For LDA, FTM and LG-MCMC, Gibbs sampling is used for inference.

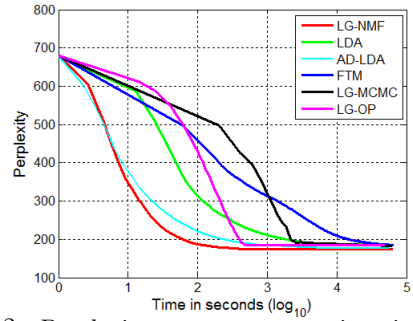

Figure 2: Perplexity versus computation time for LDA, AD-LDA, FTM, LG-NMF, LG-MCMC, and LG-OP, for the NIPS dataset. For LDA, FTM and LG-MCMC, Gibbs sampling is used for inference.

Table 1: Perplexity of different methods, on PsyRev and NIPS datasets.

DATA	LDA	AD-LDA	FTM	LG-NMF	LG-MCMC	LG-OP
PsyRev	940	946	911	902	919	915
NIPS	187	177	185	175	183	185

5.1 Topic Modeling on Small Corpora

We first consider the topic-modeling component of the LG-NMF model alone, on two widely studied document corpora: PsyRev abstracts and NIPS articles. For comparison, we consider MCMC versions of LDA (Blei et al., 2003), approximate distributed LDA (AD-LDA)(Newman et al., 2009) and the focused topic model (FTM) (Williamson et al., 2009), the latter one of the most advanced topic models in the literature. While here we consider LDA with MCMC inference, for the large-scale application below we compare to a VB LDA implementation. LDA and FTM are different models than that associated with the proposed latent-Gaussian (LG) model. Thus, in comparing with LDA and FTM, the model is different from LG-NMF, *and* the inference method is different. We therefore make a comparison to LG-MCMC, as in this case the LG-NMF and LG-MCMC *models* are the same, and the differences are only associated with the manner of inference. In addition, to give an understanding of how LG-OP performs, experimental results with respect to small corpora obtained by LG-OP are presented as well. In practice we prefer LG-NMF over LG-OP, because we have observed the former consistently converges faster (to similar models).

In Figures 1 and 2 we present the perplexity of each method, as a function of computation time, for the PsyRev and NIPS data, respectively. For the LG-NMF and LG-OP methods, time is defined by number of iterations spent iteratively updating the factor loadings and factor scores (iteratively computing the MAP solution), while for the other methods time corresponds to number of iterations of the MCMC sampler. For LG-NMF and LG-OP, the grid on α_1 was of dimension 10, and each grid point was analyzed in parallel. The rank in the LG models is set as $K = 50$, and in LDA, AD-LDA and FTM the number of topics is set to 50. For AD-LDA, the documents were assigned to 10 CPU cores.

Each of the models considered in Figures 1 and 2 were initialized exactly the same, to impose a fair comparison. Specifically, LDA is initialized at random, and then run one MCMC iteration, from which we define the initial $\mathbf{D}_1$ and $\mathbf{S}$ in the latent Gaussian models. The FTM model also uses the same initialization. The relative results in Figures 1 and 2 were robust to numerous types of initializations (including random), and the fast convergence of LG-NMF to a good result was observed repeatedly.

Let $\mathbf{Y}$ be the word count matrix for the testing documents. The perplexity is defined as

$$\text{Perplexity}\left(\mathbf{Y}\right) = \exp\left[-\frac{\sum_{d=1}^{N_t est}\log p\left(\boldsymbol{y}_d\right)}{\sum_{d=1}^{N_t est} n_{test,d}}\right] \quad (17)$$

where $n_{test,d}$ is the number of words in the dth testing document, and $\boldsymbol{y}_d$ denotes the vector of word counts for the dth document. For the perplexity calculation of our model, let $\mathbf{G} = [\mathbf{D}_1, \mathbf{S}]$. The numerator in equation 17 can then be written as

$$p(\mathbf{Y}|\mathbf{C}) = \int d\alpha_1 \int d\mathbf{G} p(\mathbf{Y}|\alpha_1, \mathbf{G}) p(\mathbf{G}|\alpha_1, \mathbf{C}) p(\alpha_1|\mathbf{C}) \quad (18)$$

For the LDA and FTM models, perplexity is computed on these models as reported in the literature (Williamson et al., 2009).

The PsyRev data contains a vocabulary of 9244 words and 1281 documents, while the NIPS data contains 13649 unique words and 1740 documents. We choose 80% of the data at random as the training set, and use the remaining 20% as the testing set.

As indicated in Figures 1 and 2, the proposed LG models asymptotically yield perplexity results very similar to LDA and FTM (often slightly better than these models; see Table 1), despite the fact that the basic model construction of the LG models is very different than the Dirichlet-distribution-based LDA and FTM. We have consistently observed that both the LG-NMF and LG-OP methods converge to a good solution much faster than LDA and FTM, and that LG-NMF converges more quickly than LG-MCMC. AD-LDA achieves similar perplexity compared with LDA, but with much less time. However, LG-NMF and LG-OP still converge faster than AD-LDA.

For each of the corpora considered here, we infer a set of topics. In the context of the LG model, the topics are characterized by the columns of $\mathbf{D}_1$, in which column k defines the strength with which each of the words contribute to topic k. As an example of the types of topics the model learns, in Table 2 we summarize a subset of topics inferred by LG-NMF for the NIPS data, where for each topic we show a subset of words that are most likely to occur. In all of our experiments, the form of the topics learned by LG-NMF was very similar to that learned by more-conventional topic models, like LDA and FTM.

5.2 Joint Analysis of Text & Binary Matrices

We compare the LG-NMF and FTM-BMF (Zhang & Carin, 2012) for predicting votes on legislation, with the text $\mathbf{C}$ and votes $\mathbf{B}$ modeled jointly (this problem was also considered in (Gerrish & Blei, 2011)). FTM-BMF is an advanced model, incorporating some of the latest methods in Bayesian analysis. The FTM analysis of the text (Williamson et al., 2009) yields state-of-the-art results, and the binary matrix factorization (BMF)(Meeds et al., 2006), based on the Indian buffet process (Griffiths & Ghahramani, 2005), is a sophisticated method of jointly modeling text and votes. FTM-BMF yields accurate results, but comes at the

Table 2: Topics of NIPS dataset, obtained by proposed LG-NMF method.

TOPIC 1	TOPIC 2	TOPIC 3
eeg	error	stimulus
ica	test	response
components	rate	cortex
component	machine	cortical
independent	validation	stimuli
analysis	errors	neurons
pca	cross	responses
TOPIC 4	**TOPIC 5**	**TOPIC 6**
circuit	neurons	tree
current	neuron	node
voltage	connections	nodes
design	firing	trees
winner	spiking	decision
circuits	excitatory	routing
gate	lateral	leaf

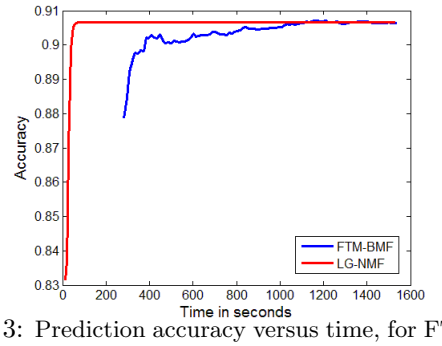

Figure 3: Prediction accuracy versus time, for FTM-BMF and LG-NMF methods, considering the legislative voting data and associated legislative text.

cost of computational complexity, as MCMC inference is required (Zhang & Carin, 2012), and the model is fairly complicated. We here follow the experimental settings considered in (Zhang & Carin, 2012).

We consider legislation (text) and roll-call data (binary matrix) for US House of Representatives (House) session 110; the data are available from `thomas.loc.gov`. Entry $B_{ij} = 1$ in $\mathbf{B}$ denotes that legislator i voted "Yea" or "Yes" to legislation j, and $B_{ij} = 0$ denotes the corresponding response is "Nay" or "No". The bills are partitioned into 6 folds, and 5 folds are used as training data, and the remaining fold is the test data. The rank for both LG-NMF and FTM-BMF is $K = 30$. For LG-NMF, we use 20 grid points two discretize $\boldsymbol{\theta} = (\alpha_1, \alpha_2)$. In this test, as in (Zhang & Carin, 2012), after learning the model based on the training data, the topic model is employed on held-out legislation, and based on the topic distribution of that legislation, the votes of *all* legislators are predicted. LG-NMF and FTM-BMF achieve virtually identical performance on this test: 90.63% prediction accuracy for FTM-BMF, and 90.65% for LG-NMF (these are asymptotic results, after the model has converged, as discussed next).

In Figure 3 we show the accuracy as a function of

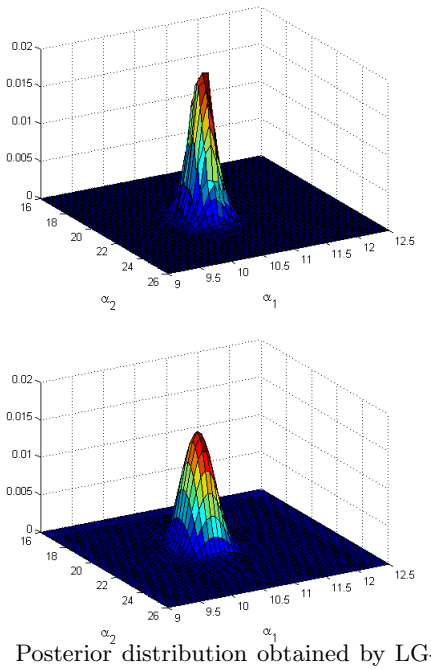

Figure 4: Posterior distribution obtained by LG-MCMC (top) and LG-NMF (bottom).

time consumed on the training data. For LG-NMF this corresponds to number of iterations in updating the factor loadings and scores, while for FTM-BMF this corresponds to number of MCMC iterations. In both cases we start from a random initialiation. Results for LG-NMF were run independently in parallel, and shown is the computation time per grid point of $\boldsymbol{\theta}$. One clearly sees the advantage of LG-NMF from a CPU perspective, with essentially no loss in accuracy. In Figure 3, the curve for FTM-BMF starts after about 300 seconds; this is because FTM-BMF first uses FTM on the documents alone, for model initialization, which takes some time.

In the above examples, as well as the large-scale example considered in the next subsection, it is observed that LG-NMF yields quantitative performance as good as the best Bayesian models in the literature, at a fraction of the computational cost. The heart of the model involves estimation of $p(\boldsymbol{\theta}|\mathbf{C}, \mathbf{B})$, and therefore it is of interest to examine the accuracy of the LG-NMF approximation of this posterior. For this comparison, we run the LG-NMF and LG-MCMC for the joint analysis of the legislation and votes. The model is the same in each case, only the inference method is different. In Figure 4 we show the LG-MCMC approximation of $p(\alpha, \alpha_2|\mathbf{C}, \mathbf{B})$, as well as an approximation based on LG-NMF; note that the scales of the figures are exactly the same. The approximated posteriors are in good agreement (again considering data from the 110th Congress). To obtain benchmark LG-MCMC results, we ran the algorithm for about 50 hours, run-

ning 30,000 burn-in iterations, and 40,000 iterations after burn-in period. The posterior distribution is obtained by samples collected every 4 iterations after the burn-in period. The accuracy of this posterior based on LG-NMF is to what we attribute the accuracy of this model.

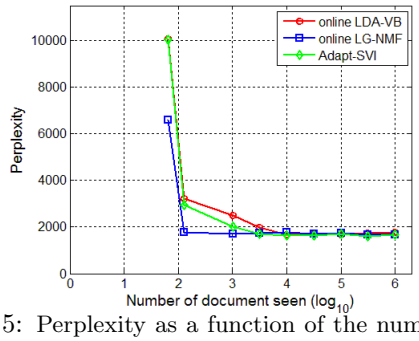

Figure 5: Perplexity as a function of the number of documents seen, for online LG-NMF, online LDA-VB and Adapt-SVI, analyzing 1 million documents Wikipedia documents.

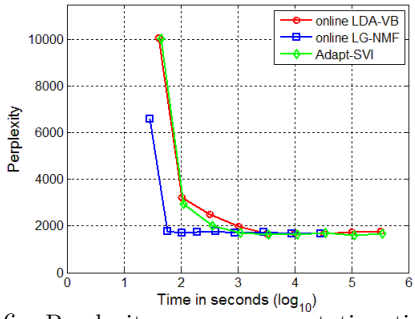

Figure 6: Perplexity versus computation time, for online LG-NMF, online LDA-VB and Adapt-SVI, analyzing 1 million documents from the Wikipedia dataset.

Table 3: Perplexity obtained by online LDA-VB, online LG-NMF and Adapt-SVI for Wikipedia dataset.

METHODS	LDA-VB	ADAPT-SVI	LG-NMF
Perplexity	1761	1666	1683

5.3 Large-Scale Dataset

We now consider LG-NMF for a large-scale corpora, in particular the Wikipedia data considered by the online LDA-VB method (Hoffman et al., 2010) and by stochastic variational inference with an adaptive learning rate (Adapt-SVI) (Ranganath et al., 2013). Borrowing nomenclature from (Hoffman et al., 2010), we refer to the stochatic gradient implementation of the model as "online LG-NMF," although the data are not processed in a streaming/online manner, but rather we randomly select batches of data from the corpus. We compare the performance of online LDA-VB (Hoffman et al., 2010) and our online LG-NMF. For the former, we use the publicly available code from `http://www.cs.princeton.edu/~blei/topicmodeling.html`, which is written in Python, while our online LG-NMF is written in Matlab. Therefore, the time comparisons are not exactly fair, and if anything the Matlab code is at a disadvantage.

The Wikipedia corpora contains 3.3 million articles, and we randomly select 1 million documents in our experiment. The number of topics for both models is $K = 200$. The batch size is 64, and we use 640 documents as the held-out testing data. The time consumed for online LDA-VB is about 90 hours. For online LG-NMF, the parallel computation time consumption is about 8 hours, with 8 grid points for parameter α_1 run in parallel; we set the number of iterations at 25 for the stochastic gradient descent method. When run serially on one computer, the LG-NMF requires about 64 hours for the entire corpus.

The per-word perplexity versus the number of documents analyzed (connected to number of batches processed) and associated CPU time are shown in Figures 5 and 6. Adapt-SVI is more computationally effective than online LDA-VB and requires fewer documents to converge to a stable perplexity. Table 3 indicates that Adapt-SVI achieves lower perplexity than online LDA-VB. LG-NMF achieves similar perplexity as Adapt-SVI, but with much less computation time.

In all experiments, LG-NMF inferred topics that were similar in quality to those generated by conventional topic models, like LDA and FTM.

6 CONCLUSIONS

Borrowing ideas from the recently developed INLA method (Rue et al., 2009), we have developed a new means of performing topic modeling, based on latent Gaussian priors on the factor loadings and scores. We have jointly analyzed text and associated binary data, and have developed a stochastic-gradient-descent implementation that scales to massive data. The use of Gaussian priors for matrix factorization is widely employed (Salakhutdinov & Mnih, 2008; Silva & Carin, 2012), but to the authors' knowledge its use in topic modeling has been limited, and leveraging ideas from INLA has not been considered previously. Encouraging performance on a diverse set of applications was observed relative to some of the most advanced methods in the literature. Optimization plays a key role here in estimating the Bayesian posterior, offering the potential in future work to further integrate ideas from the optimization and Bayesian communities.

Acknowledgements

The research reported here was supported in part by ARO, DARPA, DOE, NGA, and ONR.

References

Abernethy, J., Bach, F., Evgeniou, T., and Vert, J.-P. A new approach to collaborative filtering: operator estimation with spectral regularization. *J. Machine Learning Research*, 2009.

Albert, J. H. and Chib, S. Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88(422):669–679, 1993.

Beal, M. *Variational Algorithms for Approximate Bayesian Inference*. PhD thesis, Gatsby Computational Neuroscience Unit, University College London, 2003.

Bittorf, V., Recht, B., Re, C., and Tropp, J.A. Factoring nonnegative matrices with linear programs. In *NIPS*, 2012.

Blei, D., Ng, A., and Jordan, M. Latent Dirichlet allocation. *JMLR*, 2003.

Blei, D., Griffiths, T., and Jordan, M. The chinese restaurant process and bayesian nonparametric inference of topic hierarchies. *Journal of the ACM*, 2010.

Bottou, L. *Online Algorithms and Stochastic Approximations*. Cambridge University Press, 1998.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 2010.

Candès, E. and Recht, B. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, pp. 111–119, 2012.

Chan, T.F. and Wong, C.K. Convergence of the alternating minimization algorithm for blind deconvolution. *Linear Algebra and its App.*, 2000.

Gerrish, S. and Blei, D.M. Predicting legislative roll calls from text. In *ICML*, 2011.

Griffiths, T.L. and Ghahramani, Z. Infinite latent feature models and the indian buffet process. In *NIPS*, pp. 475–482, 2005.

Hoffman, M.D., Blei, D.M., and Bach, F. Online learning for latent dirichlet allocation. In *NIPS*, 2010.

Jenatton, R., Mairal, J., Obozinski, G., and Bach, F. Proximal methods for sparse hierarchical dictionary learning. In *ICML*, 2010.

Lee, D.D. and Seung, H.S. Algorithms for nonnegative matrix factorization. In *NIPS*, 2000.

Lin, C. J. Projected gradient methods for non-negative matrix factorization. *Neural Computation*, 2007.

M. Zhou, L. Hannah, D. Dunson and Carin, L. Beta-negative binomial process and Poisson factor analysis. In *AISTATS*, 2012.

Mairal, J., Bach, F., Ponce, J., and Sapiro, G. Online dictionary learning for sparse coding. In *Proc. ICML*, 2009.

Meeds, E., Ghahramani, Z., Neal, R., and Roweis, S. Modeling dyadic data with binary latent factors. In *Neural Information Processing Systems*, 2006.

Minka, T. Expectation propagation for approximate bayesian inference. In *17th Conference in Uncertainty in Artificial Intelligence*, 2001.

Newman, D., Asuncion, A., Smyth, P., and Welling, M. Distributed algorithms for topic models. *J. Machine Learning Research*, 10(8):1801–1828, 2009.

Ranganath, R., Wang, C, Blei, D., and Xing, E. An adaptive learning rate for stochastic variational inference. In *ICML*, 2013.

Recht, B. A simpler approach to matrix completion. *J. Machine Learning Research*, 2011.

Rue, H., Martino, S., and Chopin, N. Approximate Bayesian inference for latent Gaussian models using integrated nested Laplace approximations. *J. Royal Stat. Soc., Series B*, 2009.

Salakhutdinov, R. and Mnih, A. Bayesian probabilistic matrix factorization with MCMC. In *NIPS*, 2008.

Silva, J. and Carin, L. Active learning for online bayesian matrix factorization. In *ACM SIGKDD*, 2012.

Suchard, M.A., Wang, Q., Chan, C., Frelinger, J., Cron, M., and West, M. Understanding GPU programming for statistical computation: Studies in massively parallel massive mixtures. *J. Comp. Graph. Statistics*, 2012.

Wang, C. and Blei, D. Variational infernece for the nested chinese restaurant process. In *NIPS*, 2009.

Williamson, S., Wang, C., Heller, K., and Blei, D. Focused topic models. *NIPS Workshop on Applications of Topic Models*, 2009.

Zhang, X. and Carin, L. Joint modeling of a matrix with associated text via latent binary features. In *NIPS*, 2012.