Submitted to the Annals of Applied Statistics
arXiv: math.PR/154339

NONPARAMETRIC BAYESIAN SPARSE FACTOR MODELS WITH APPLICATION TO GENE EXPRESSION MODELLING

By David Knowles[*] and Zoubin Ghahramani[†]

University of Cambridge

A nonparametric Bayesian extension of Factor Analysis (FA) is proposed where observed data $\mathbf{Y}$ is modeled as a linear superposition, $\mathbf{G}$, of a potentially infinite number of hidden factors, $\mathbf{X}$. The Indian Buffet Process (IBP) is used as a prior on $\mathbf{G}$ to incorporate sparsity and to allow the number of latent features to be inferred. The model's utility for modeling gene expression data is investigated using randomly generated datasets based on a known sparse connectivity matrix for *E. Coli*, and on three biological datasets of increasing complexity.

1. Introduction.

Principal Components Analysis (PCA), Factor Analysis (FA) and Independent Components Analysis (ICA) are models which explain observed data, $\mathbf{y}_n \in \mathbb{R}^D$, in terms of a linear superposition of independent hidden factors, $\mathbf{x}_n \in \mathbb{R}^K$, so

$$\mathbf{y}_n = \mathbf{G}\mathbf{x}_n + \boldsymbol{\epsilon}_n \tag{1}$$

where $\mathbf{G}$ is the factor loading matrix and $\boldsymbol{\epsilon}_n$ is a noise vector, usually assumed to be Gaussian. These algorithms can be expressed in terms of performing inference in appropriate probabilistic models. The latent factors are usually considered as random variables, and the mixing matrix as a parameter to estimate. In both PCA and FA the latent factors are given a standard (zero mean, unit variance) normal prior. In PCA the noise is assumed to be isotropic, whereas in FA the noise covariance is only constrained to be diagonal. A standard approach in these models is to integrate out the latent factors and find the maximum likelihood estimate of the mixing matrix. In ICA the latent factors are assumed to be heavy-tailed, so it is not usually

[*]Supported by Microsoft Research through the Roger Needham Scholarship at Wolfson College, University of Cambridge.

[†]Supported by EPSRC Grant EP/F027400/1

Keywords and phrases: Nonparametric Bayes, Sparsity, Factor Analysis, Markov Chain Monte Carlo, Indian Buffet Process

AMS 2000 subject classifications: Primary 62H25; secondary 62F15

possible to integrate them out. In this paper we take a fully Bayesian approach, viewing not only the hidden factors but also the mixing coefficients as random variables whose posterior distribution given data we aim to infer.

Sparsity plays an important role in latent feature models, and is desirable for several reasons. It gives improved predictive performance, because factors irrelevant to a particular dimension are not included. Sparse models are more readily interpretable since a smaller number of factors are associated with observed dimensions. In many real world situations there is an intuitive reason why we expect sparsity: for example, in gene regulatory networks a transcription factor will only regulate genes with specific motifs. In our previous work (Knowles and Ghahramani, 2007) we investigated the use of sparsity the on latent factors $\mathbf{x}_n$, but this formulation is not appropriate in the case of modeling gene expression, where, as described above, a transcription factor will regulate only a small set of genes, corresponding to sparsity in the factor loadings, $\mathbf{G}$. Here we propose a novel approach to sparse latent factor modeling where we place sparse priors on the factor loading matrix, $\mathbf{G}$. The Bayesian Factor Regression Model of West *et al.* (2007) is closely related to our work in this way, although the hierarchical sparsity prior they use is somewhat different. An alternative "soft" approach to incorporating sparsity is to put a Gamma(a, b) (usually exponential, i.e. $a = 1$) prior on the precision of each element of $\mathbf{G}$ independently, resulting in the elements of $\mathbf{G}$ being marginally Student-t distributed a priori: see Fokoue (2004), Fevotte and Godsill (2006), and Archambeau and Bach (2009). A LASSO-based approach to generating a sparse factor loading has also been developed (Witten, Tibshirani and Hastie, 2009; Zou, Hastie and Tibshirani, 2004). We compare these sparsity schemes empirically in the context of gene expression modeling.

A problematic issue with this type of model is how to choose the latent dimensionality of the factor space, K. Model selection can be used to choose between different values of K, but generally requires significant manual input and still requires the range of K over which to search to be specified. Zhang *et al.* (2004) applied Reversible Jump MCMC to PCA, which has many of the advantages of our approach: a posterior distribution over the number of latent dimensions can be approximated, and the number of latent dimensions could potentially be unbounded. However, RJ MCMC is considerably more complex to implement for sparse Factor Analysis than our proposed framework.

We use the Indian Buffet Process (Griffiths and Ghahramani, 2006), which defines a distribution over infinite binary matrices, to provide sparsity and a framework for inferring the appropriate latent dimension of the dataset using a straightforward Gibbs sampling algorithm. The Indian Buffet Process (IBP) allows a potentially unbounded number of latent factors, so we do not have to specify a maximum number of latent dimensions a priori. We denote our model "NSFA" for "Non-parametric Sparse Factor Analysis". Our model is closely related to that of Rai and Daumé III (2008), and is a simultaneous development.

2. The Model.

We will define our model in terms of Equation 1. Let $\mathbf{Z}$ be a binary matrix whose (d, k)-th element represents whether observed dimension d includes any contribution from factor k. We then model the mixing matrix by

$$p(g_{dk}|Z_{dk}, \lambda_k) = Z_{dk}\mathcal{N}\left(g_{dk}; 0, \lambda_k^{-1}\right) + (1 - Z_{dk})\delta_0(g_{dk}) \tag{2}$$

where λ_k is the inverse variance (precision) of the kth factor and δ_0 is a delta function (pont-mass) at 0. Distributions of this type are sometimes known as "spike and slab" distributions. We allow a potentially infinite number of hidden sources, so that $\mathbf{Z}$ has infinitely many columns, although only a finite number will have non-zero entries. This construction allows us to use the IBP to provide sparsity and define a generative process for the number of latent factors.

We assume independent Gaussian noise, $\boldsymbol{\epsilon}_n$, with diagonal covariance matrix Ψ. We find that for many applications assuming isotropic noise is too restrictive, but this option is available for situations where there is strong prior belief that all observed dimensions should have the same noise variance. The latent factors, $\mathbf{x}_n$, are given Gaussian priors. Figure 1 shows the complete graphical model.

2.1. *Defining a distribution over infinite binary matrices.*

We now define our infinite model by taking the limit of a series of finite models.

Start with a finite model. We derive the distribution on $\mathbf{Z}$ by defining a finite K model and taking the limit as $K \to \infty$. We then show how the infinite case corresponds to a simple stochastic process.

We have D dimensions and K hidden sources. Recall that z_{dk} of matrix $\mathbf{Z}$ tells us whether hidden source k contributes to dimension d. We assume that the probability of a source k contributing to any dimension is π_k, and that the rows are generated independently. We find

$$P(\mathbf{Z}|\boldsymbol{\pi}) = \prod_{k=1}^{K}\prod_{d=1}^{D} P(z_{dk}|\pi_k) = \prod_{k=1}^{K} \pi_k^{m_k}(1-\pi_k)^{D-m_k} \tag{3}$$

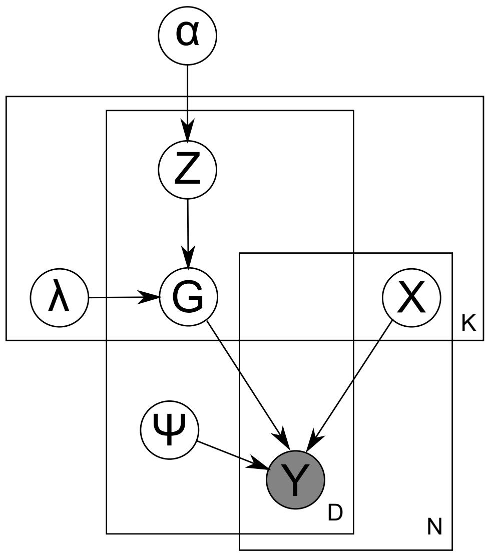


FIG 1. *Graphical model*

where $m_k = \sum_{d=1}^{D} z_{dk}$ is the number of dimensions to which source k contributes. The inner term of the product is a binomial distribution, so we choose the conjugate Beta(r, s) distribution for π_k. For now we take $r = \frac{\alpha}{K}$ and $s = 1$, where α is the strength parameter of the IBP. The model is defined by

$$\pi_k | \alpha \sim \text{Beta}\left(\frac{\alpha}{K}, 1\right) \tag{4}$$

$$z_{dk} | \pi_k \sim \text{Bernoulli}(\pi_k) \tag{5}$$

Due to the conjugacy between the binomial and beta distributions we are able to integrate out π to find

$$P(\mathbf{Z}) = \prod_{k=1}^{K} \frac{\frac{\alpha}{K}\Gamma(m_k + \frac{\alpha}{K})\Gamma(D - m_k + 1)}{\Gamma(D + 1 + \frac{\alpha}{K})} \tag{6}$$

where $\Gamma(.)$ is the Gamma function.

Take the infinite limit. Griffiths and Ghahramani (2006) define a scheme to order the non-zero rows of $\mathbf{Z}$ which allows us to take the limit $K \to \infty$ and find

$$P(\mathbf{Z}) = \frac{\alpha^{K_+}}{\prod_{h>0} K_h!} \exp\left(-\alpha H_D\right) \prod_{k=1}^{K_+} \frac{(D - m_k)!(m_k - 1)!}{N!} \tag{7}$$

where K_+ is the number of active features (i.e. non-zero columns of $\mathbf{Z}$), $H_D = \sum_{j=1}^{D} \frac{1}{j}$ is the D-th harmonic number, and K_h is the number of rows whose entries correspond to the binary number h.

Go to an Indian Buffet. This distribution corresponds to a simple stochastic process, the Indian Buffet Process. Consider a buffet with a seemingly infinite number of dishes (hidden sources) arranged in a line. The first customer (observed dimension) starts at the left and samples Poisson(α) dishes. The ith customer moves from left to right sampling dishes with probability $\frac{m_k}{i}$ where m_k is the number of customers to have previously sampled dish k. Having reached the end of the previously sampled dishes, he tries Poisson($\frac{\alpha}{i}$) new dishes. Figure 2 shows two draws from the IBP for two different values of α.

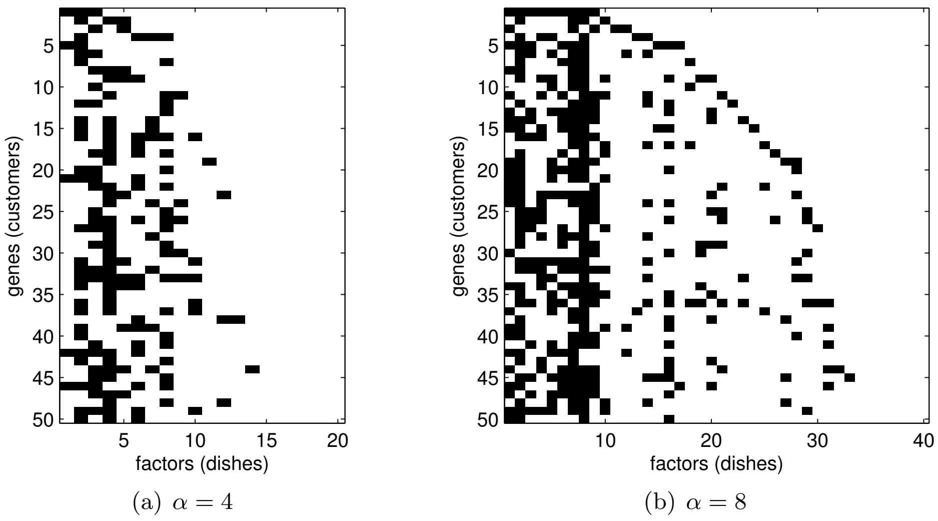


FIG 2. *Draws from the one parameter IBP for two different values of α.*

If we apply the same ordering scheme to the matrix generated by this process as for the finite model, we recover the correct exchangeable distribution. Since the distribution is exchangeable with respect to the customers we find by considering the last customer that

$$P(z_{kt} = 1 | \mathbf{z}_{-kn}) = \frac{m_{k,-t}}{D} \tag{8}$$

where $m_{k,-t} = \sum_{s \neq t} z_{ks}$, which is used in sampling $\mathbf{Z}$. By exchangeability and considering the first customer, the number of active sources for each dimension follows a Poisson(α) distribution, and the expected number of entries in $\mathbf{Z}$ is $D\alpha$. We also see that the number of active features, $K_+ = \sum_{d=1}^{D} \text{Poisson}(\frac{\alpha}{d}) = \text{Poisson}(\alpha H_D)$.

3. Related work.

The Bayesian Factor Regression Model (BFRM) of West *et al.* (2007) is closely related to the finite version of our model. The key difference is the use of a hierarchical sparsity prior. Each element of $\mathbf{G}$ has prior of the form

$$g_{dk} \sim (1 - \pi_{dk})\delta_0(g_{dk}) + \pi_{dk}\mathcal{N}\left(g_{dk}; 0, \lambda_k^{-1}\right)$$

The finite IBP model is equivalent to setting $\pi_{dk} = \pi_k \sim \text{Beta}(\alpha/K, 1)$ and then integrating out π_k. In BFRM a hierarchical prior is used:

$$\pi_{dk} \sim (1 - \rho_k)\delta_0(\pi_{dk}) + \rho_k \text{Beta}(\pi_{dk}; am, a(1-m))$$

where $\rho_k \sim \text{Beta}(sr, s(1-r))$. Non-zero elements of π_{dk} are given a diffuse prior favoring larger probabilities ($a = 10, m = 0.75$ are suggested in West *et al.* (2007)), and ρ_k is given a prior which strongly favors small values, corresponding to a sparse solution (e.g. $s = D, r = \frac{5}{D}$).

Note that on integrating out π_{dk}, the prior on g_{dk} is

$$g_{dk} \sim (1 - m\rho_k)\delta_0(g_{dk}) + m\rho_k\mathcal{N}\left(g_{dk}; 0, \lambda_k^{-1}\right)$$

This hierarchical sparsity prior is motivated by improved interpretability in terms of less uncertainty in the posterior as to whether an element of G is non-zero. However, this comes at a cost of significantly increased computation and reduced predictive performance, suggesting that the uncertainty removed from the posterior was actually important.

The LASSO-based Sparse PCA (SPCA) method of Zou, Hastie and Tibshirani (2004) and Witten, Tibshirani and Hastie (2009) has similar aims to our work in terms of providing a sparse variant of PCA to aid interpretation of the results. However, since SPCA is not formulated as a generative model it is not necessarily clear how to choose the regularization parameters or dimensionality without resorting to cross-validation. In our experimental comparison to SPCA we adjust the regularization constants such that each component explains roughly the same proportion of the total variance as the corresponding standard (non-sparse) principal component.

4. Inference.

Given the observed data $\mathbf{Y}$, we wish to infer the hidden sources $\mathbf{X}$, which sources are active $\mathbf{Z}$, the mixing matrix $\mathbf{G}$, and all hyperparameters. We use Gibbs sampling, but with Metropolis-Hastings (MH) steps for sampling new features. We draw samples from the marginal distribution of the model parameters given the data by successively sampling the conditional distributions of each parameter in turn, given all other parameters.

Since we assume independent Gaussian noise, the likelihood function is

$$P(\mathbf{Y}|\mathbf{G},\mathbf{X},\boldsymbol{\psi}) = \prod_{t=1}^{N} \frac{1}{(2\pi)^{\frac{D}{2}}|\boldsymbol{\psi}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\mathbf{y}_n - \mathbf{G}\mathbf{x}_n)^T \boldsymbol{\psi}^{-1}(\mathbf{y}_n - \mathbf{G}\mathbf{x}_n)\right) \tag{9}$$

where $\boldsymbol{\psi}$ is a diagonal noise covariance matrix.

Notation. We use $-$ to denote the "rest" of the model, i.e. the values of all variables not explicitly conditioned upon in the current state of the Markov chain. The r-th row and c-th column of matrix A are denoted $A_{r:}$ and $A_{:c}$ respectively.

Mixture coefficients. We first derive a Gibbs sampling step for an individual element of the IBP matrix, Z_{dk}, determining whether factor k is active for dimension d. Recall that λ_k is the precision (inverse covariance) of the factor loadings for the k-th factor. The ratio of likelihoods can be calculated using Equation 9. Integrating out the (d,k)-th element of the factor loading matrix, g_{dk} (whose prior is given by Equation 2) we obtain

$$\frac{P(\mathbf{Y}|Z_{dk}=1,-)}{P(\mathbf{Y}|Z_{dk}=0,-)} = \frac{\int P(\mathbf{Y}|g_{dk},-)\mathcal{N}\left(g_{dk};0,\lambda_k^{-1}\right) dg_{dk}}{P(\mathbf{Y}|g_{dk}=0,-)} \tag{10}$$

$$= \sqrt{\frac{\lambda_k}{\lambda}} \exp\left(\frac{1}{2}\lambda\mu^2\right) \tag{11}$$

where we have defined $\lambda = \psi_d^{-1} X_{k:}^T X_{k:} + \lambda_k$ and $\mu = \frac{\psi_d^{-1}}{\lambda} X_{k:}^T \hat{E}_{d:}$ with the matrix of residuals $\hat{\mathbf{E}} = \mathbf{Y} - \mathbf{G}\mathbf{X}$ evaluated with $g_{dk} = 0$. The dominant calculation is that for μ since the calculation for λ can be cached. This operation is $O(N)$ and must be calculated $D \times K$ times, so sampling the IBP matrix, $\mathbf{Z}$ and factor loading matrix, $\mathbf{G}$ is order $O(NDK)$.

From the exchangeability of the IBP we can imagine that dimension d was the last to be observed, so that the ratio of the priors is

$$\frac{P(Z_{dk}=1|-)}{P(Z_{dk}=0|-)} = \frac{m_{-d,k}}{N-1-m_{-d,k}} \tag{12}$$

where $m_{-d,k}$ is the number of dimensions for which factor k is active, excluding the current dimension d. Multiplying Equations 11 and 12 gives the expression for the ratio of posterior probabilities for Z_{dk} being 1 or 0, which is used for sampling. If Z_{dk} is set to 1, we sample $g_{dk}|- \sim \mathcal{N}\left(\mu, \lambda^{-1}\right)$ with μ, λ defined as for Equation 11.

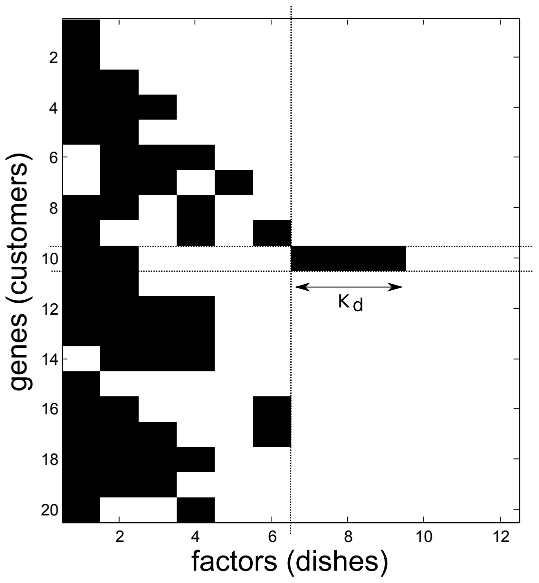


FIG 3. *A diagram to illustrate the definition of κ_d, for $d = 10$.*

Adding new features. $\mathbf{Z}$ is a matrix with infinitely many columns, but the non-zero columns contribute to the likelihood and are held in memory. However, the zero columns still need to be taken into account since the number of active factors can change. Let κ_d be the number of columns of $\mathbf{Z}$ which contain 1 only in row d, i.e. the number of features which are active only for dimension d. Note that due to the form of the prior for elements of $\mathbf{Z}$ given in Equation 12, $\kappa_d = 0$ for all d after a sampling sweep of $\mathbf{Z}$. Figure 3 illustrates κ_d for a sample $\mathbf{Z}$ matrix.

New features are proposed by sampling κ_d with a MH step. It is possible to integrate out either the new elements of the mixing matrix, $\mathbf{g}$ (a $1 \times \kappa_d$ vector), or the new rows of the latent feature matrix, $\mathbf{X}'$ (a $\kappa_d \times N$ matrix), but not both. Since the latter generally has higher dimension, we choose to integrate out $\mathbf{X}'$ and include $\mathbf{g}^T$ as part of the proposal. Thus the proposal is $\xi = \{\kappa_d, \mathbf{g}\}$, and we propose a move $\xi \rightarrow \xi^*$ with probability $J(\xi^*|\xi)$. In this case $\xi = \emptyset$ since as noted above $\kappa_d = 0$ initially. The simplest proposal, following Meeds *et al.* (2006), would be to use the prior on ξ^*, i.e.

$$J(\xi) = P(\kappa_d|\alpha) \cdot p(\mathbf{g}|\kappa_d, \lambda_k) = \text{Poisson}\,(\kappa_d; \gamma) \cdot N(\mathbf{g}; 0, \lambda_k^{-1})$$

where $\gamma = \frac{\alpha}{D-1}$.

Unfortunately, the rate constant of the Poisson prior tends to be so small that new features are very rarely proposed, resulting in slow mixing. To remedy this we modify the proposal distribution for κ_d and introduce two tunable parameters, π and λ.

$$J(\kappa_d) = (1-\pi)\text{Poisson}\,(\kappa_d; \lambda\gamma) + \pi\mathbf{1}(\kappa_d = 1) \tag{13}$$

Thus the Poisson rate is multiplied by a factor λ, and a spike at $\kappa_d = 1$ is added with mass π. The proposal is accepted with probability $\min(1, a_{\xi\to\xi^*})$ where

$$a_{\xi\to\xi^*} = \frac{P(\xi^*|-,Y)J(\xi|\xi^*)}{P(\xi|-,Y)J(\xi^*|\xi)} = \frac{P(Y|\xi^*,-)P(\kappa_d|\alpha)p(\mathbf{g}|\kappa_d,\lambda_k)}{P(Y|-)J(\kappa_d)p(\mathbf{g}|\kappa_d,\lambda_k)} = a_l \cdot a_p \tag{14}$$

where

$$a_l = \frac{P(Y|\xi^*,-)}{P(Y|-)} \tag{15}$$

$$a_p = \frac{P(\kappa_d|\alpha)}{J(\kappa_d)} = \frac{\text{Poisson}\,(\kappa_d;\gamma)}{\text{Poisson}\,(\kappa_d;\lambda\gamma)} \tag{16}$$

Note that we take $J(\xi|\xi^*) = 1$ since $\xi = \emptyset$. To calculate the likelihood ratio, a_l, we need the collapsed likelihood under the new proposal:

$$P(Y_{d:}|\xi^*,-) = \prod_{n=1}^N \int P(Y_{dn}|\xi^*,\mathbf{x}'_n,-)P(\mathbf{x}'_n)d\mathbf{x}' \tag{17}$$

$$= \prod_{n=1}^N (2\pi\psi_d^{-1})^{-\frac{1}{2}}(2\pi)^{\frac{\kappa_d}{2}}|\mathbf{M}|^{-\frac{1}{2}}\exp\left(\frac{1}{2}(\mathbf{m}_n^T\mathbf{M}\mathbf{m}_n - \psi_d^{-1}\hat{E}_{dn}^2)\right) \tag{18}$$

where we have defined $\mathbf{M} = \psi_d^{-1}\mathbf{g}\mathbf{g}^T + I_{\kappa_d}$ and $\mathbf{m}_n = \mathbf{M}^{-1}\psi_d^{-1}\mathbf{g}\hat{E}_{dn}$ with the matrix of residuals $\hat{\mathbf{E}} = \mathbf{Y} - \mathbf{G}\mathbf{X}$. The likelihood under the current sample is:

$$P(Y_{d:}|\xi,-) = \prod_{n=1}^N (2\pi\psi_d^{-1})^{-\frac{1}{2}}\exp\left(-\frac{1}{2}\psi_d^{-1}\hat{E}_{dn}^2\right) \tag{19}$$

Substituting these likelihood terms into the expression for the ratio of likelihood terms, a_l, gives

$$a_l = (2\pi)^{\frac{N\kappa_d}{2}}|\mathbf{M}|^{-\frac{N}{2}}\exp\left(\frac{1}{2}\sum_n \mathbf{m}_n^T\mathbf{M}\mathbf{m}_n\right) \tag{20}$$

We found that appropriate scheduling of the sampler improved mixing, particularly with respect to adding new features. The final scheme we settled on is described in Algorithm 1.

IBP parameters. We can choose to sample the IBP strength parameter α, with conjugate Gamma(e, f) prior (note that we use the inverse scale parameterization of the Gamma distribution). The conditional prior of Equation 7, acts as the likelihood term and the posterior update is as follows:

$$P(\alpha|\mathbf{Z}) \propto P(\mathbf{Z}|\alpha)P(\alpha) = \text{Gamma}\left(\alpha; K_+ + e, f + H_D\right) \tag{21}$$

where K_+ is the number of active sources and $H_D = \sum_{j=1}^{D} \frac{1}{j}$ is the D-th harmonic number.

The remaining sampling steps are standard, but are included here for completeness.

Latent variables. Sampling the columns $\mathbf{x}_n$ of the latent variable matrix $\mathbf{X}$ for each $t \in [1, \dots, N]$ we have

$$P(\mathbf{x}_n|-) \propto P(\mathbf{y}_n|\mathbf{x}_n, -)P(\mathbf{x}_n) = \mathcal{N}\left(\mathbf{x}_n; \boldsymbol{\mu}_n, \boldsymbol{\Lambda}\right) \tag{22}$$

where we have defined $\boldsymbol{\Lambda} = \mathbf{G}^T \boldsymbol{\psi}^{-1}\mathbf{G} + I$ and $\boldsymbol{\mu}_n = \boldsymbol{\Lambda}^{-1}\mathbf{G}^T\boldsymbol{\psi}^{-1}\mathbf{y}_n$. Note that since $\boldsymbol{\Lambda}$ does not depend on n we only need to compute and invert it once per iteration. Calculating $\boldsymbol{\Lambda}$ is order $O(K^2 D)$, and inverting it is $O(K^3)$. Calculating $\boldsymbol{\mu}_t$ is order $O(KD)$ and must be calculated for all N $\mathbf{x}_t$'s, a total of $O(NKD)$. Thus sampling $\mathbf{X}$ is order $O(K^2 + K^3 + NKD)$.

Factor precision. If the mixture coefficient prior precisions λ_k are constrained to be equal, we have $\lambda_k = \lambda \sim$ Gamma(c, d). The posterior update is then given by $\lambda|G \sim \text{Gamma}(c + \frac{\sum_k m_k}{2}, d + \sum_{d,k} G_{dk}^2)$.

However, if the variances are allowed to be different for each column of $\mathbf{G}$, we set $\lambda_k \sim$ Gamma(c, d), and the posterior update is given by $\lambda_k|G \sim \text{Gamma}(c + \frac{m_k}{2}, d + \sum_d G_{dk}^2)$. In this case we may also wish to share power across factors, in which case we also sample d. Putting a Gamma prior on d such that $d \sim \text{Gamma}(c_0, d_0)$, the posterior update is $d|\lambda_k \sim \text{Gamma}(c_0 + cK, d_0 + \sum_{k=1}^{K} \lambda_k)$.

Noise variance. The additive Gaussian noise can be constrained to be isotropic, in which case the inverse variance is given a Gamma prior: $\psi_d^{-1} = \psi^{-1} \sim \text{Gamma}(a, b)$ which gives the posterior update $\psi^{-1}|- \sim \text{Gamma}(a + \frac{ND}{2}, b + \sum_{d,n} \hat{E}_{dn}^2)$.

However, if the noise is only assumed to be independent (which we have found to be more appropriate for gene expression data), then each dimension has a separate noise variance, whose inverse is given a Gamma prior: $\psi_d^{-1} \sim \text{Gamma}(a, b)$ which gives the posterior update $\psi_d^{-1}|- \sim \text{Gamma}(a + \frac{N}{2}, b + \sum_n E_{dn}^2)$ where the matrix of residuals $\hat{\mathbf{E}} = \mathbf{Y} - \mathbf{G}\mathbf{X}$. We can share power between dimensions by giving the hyperparameter b a hyperprior

Gamma(a_0, b_0) resulting in the Gibbs update $b|- \sim \text{Gamma}(a_0 + aD, b_0 + \sum_{d=1}^{D} \psi_d^{-1})$. This hierarchical prior results in soft coupling between the noise variances in each dimension, so we will refer to this variant as *sc*.

Algorithm 1 One iteration of the NSFA sampler

```
for $d = 1$ to $D$ do
  for $k = 1$ to $K$ do
    Sample $Z_{dk}$
  end for
  Sample $\kappa_d$
end for
for $n = 1$ to $N$ do
  Sample $X_{:n}$
end for
Sample $\alpha, \phi, \lambda_g$
```

5. Results.

We compare the following models:

- FA - Bayesian Factor Analysis, see for example Kaufman and Press (1973) or Rowe and Press (1998)
- AFA - Factor Analysis with ARD prior to determine active sources
- FOK - The sparse Factor Analysis method of Fokoue (2004), Fevotte and Godsill (2006) and Archambeau and Bach (2009)
- SPCA - The Sparse PCA method of Zou, Hastie and Tibshirani (2004)
- BFRM - Bayesian Factor Regression Model of West *et al.* (2007).
- SFA - Sparse Factor Analysis, using the finite IBP
- NSFA - The proposed model: Nonparametric Sparse Factor Analysis

Note that all of these models can be learned using the software package we provide simply by using appropriate settings.

5.1. *Synthetic data.*

Since generating a connectivity matrix $\mathbf{Z}$ from the IBP itself would clearly bias towards our model, we instead use the $D = 100$ gene by $K = 16$ factor *E. Coli* connectivity matrix derived in Kao *et al.* (2004) from RegulonDB and current literature. We ignore whether the connection is believed to be up or down regulation, resulting in a binary matrix $\mathbf{Z}$. We generate random datasets with $N = 100$ samples by drawing the non-zero elements of $\mathbf{G}$ (corresponding to the elements of $\mathbf{Z}$ which are non-zero), and all elements of $\mathbf{X}$, from a zero mean unit variance Gaussian, calculating $\mathbf{Y} = \mathbf{GX} + \mathbf{E}$, where $\mathbf{E}$ is Gaussian white noise with variance set to give a signal to noise ratio of 10.

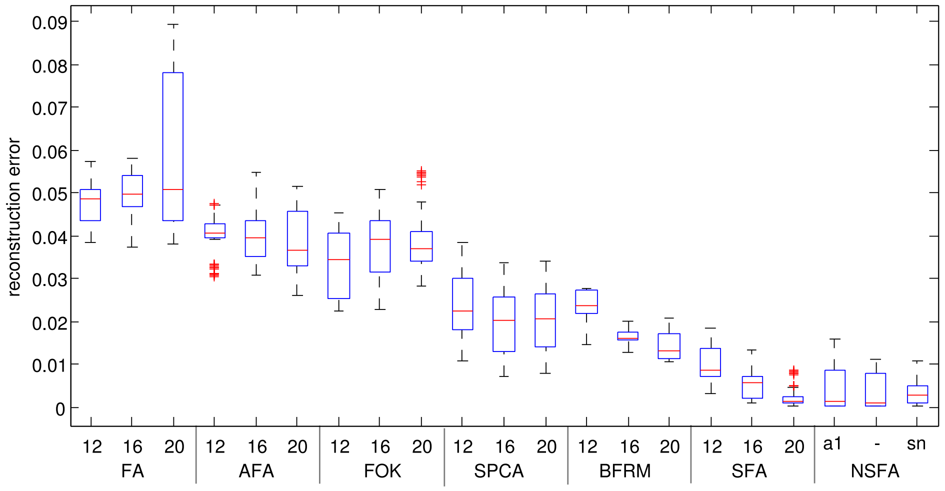


FIG 4. *Boxplot of reconstruction errors for simulated data derived from the* E. Coli *connectivity matrix of Kao et al. (2004). Ten datasets were generated and the reconstruction error calculated for the last ten samples from each algorithm. Numbers refer to the number of latent factors used, K.* a1 *denotes fixing $\alpha = 1$.* sn *denotes sharing power between noise dimensions.*

Here we will define the reconstruction error, E_r as

$$E_r(G, \hat{G}) = \frac{1}{DK} \sum_{k=1}^{K} \min_{\hat{k} \in \{1,..,\hat{K}\}} \sum_{d=1}^{D} (G_{dk} - G_{d\hat{k}})^2$$

where $\hat{G}, \hat{K}$ are the inferred quantities. Although we minimize over permutations, we do not minimize over rotations since, as noted in Fokoue (2004), the sparsity of the prior stops the solution being rotation invariant. We average this error over the last ten samples of the MCMC run. This error function does not penalize inferring extra spurious factors, so we will investigate this possibility separately. The precision and recall of active elements of the $\mathbf{Z}$ achieved by each algorithm (after thresholding for the non-sparse algorithms) are presented in the supplementary material (see Supplement A), but omitted here since the results are consistent with the reconstruction error.

The reconstruction error for each method with different numbers of latent features is shown in Figure 4. Ten random datasets were used and for the sampling methods (all but SPCA) the results were averaged over the last ten samples out of 1000. Unsurprisingly, plain Factor Analysis (FA) performs the worst, with increasing overfitting as the number of factors is increased. For $\hat{K} = 20$ the variance is also very high, since the four spurious features fit noise. Using an ARD prior on the features (AFA) improves the performance, and overfitting no longer occurs. The reconstruction error is actually less for $\hat{K} = 20$, but this is an artifact due to the reconstruction error not penalizing additional spurious features in the inferred $\mathbf{G}$. The Sparse PCA (SPCA) of Zou, Hastie and Tibshirani (2004) shows improved reconstruction compared to the non-sparse methods (FA and AFA) but does not perform as well as the Bayesian sparse models. Sparse factor analysis (SFA), the finite version of the full infinite model, performs very well. The Bayesian Factor Regression Model (BFRM) performs significantly better than the ARD factor analysis (AFA), but not as well as our sparse model (SFA). It is interesting that for BFRM the reconstruction error decreases significantly with increasing $\hat{K}$, suggesting that the default priors may actually encourage too much sparsity for this dataset. Fokoue's method (FOK) only performs marginally better than AFA, suggesting that this "soft" sparsity scheme is not as effective at finding the underlying sparsity in the data. Overfitting is also seen, with the error increasing with $\hat{K}$. This could potentially be resolved by placing an appropriate per factor ARD-like prior over the scale parameters of the Gamma distributions controlling the precision of elements of $\mathbf{G}$. Finally, the Non-parametric Sparse Factor Analysis (NSFA) proposed here and in Rai and Daumé III (2008) performs very well. With fixed $\alpha = 1$ (a1) or inferring α we see very similar performance. Using the soft coupling (sc) variant which shares power between dimensions when fitting the noise variances seems to reduce the variance of the sampler, which is reasonable in this example since the noise was in fact isotropic.

Since the reconstruction error does not penalize spurious factors it is important to check that NSFA is not scoring well simply by inferring many additional factors. Histograms for the number of latent features inferred for the nonparametric sparse model are shown in Figure 5. This represents an approximate posterior over K. For fixed $\alpha = 1$ the distribution is centered around the true value of $K = 16$, with minimal bias ($\mathbb{E}K = 16.1$). The variance is significant (standard deviation of 1.46), but is reasonable considering the noise level (SNR=10) and that in some of the random datasets, elements of $\mathbf{Z}$ which are 1 could be masked by very small corresponding values of $\mathbf{G}$. This hypothesis is supported by the results of a similar experiment where $\mathbf{G}$ was set equal to $\mathbf{Z}$. In this case, the sampler always converged to at least 16 features, but would also sometimes infer spurious features from noise (results not shown). When inferring α some bias and skew are noticeable. The mean of the posterior is now at 18.3 with standard deviation 2.0, suggesting

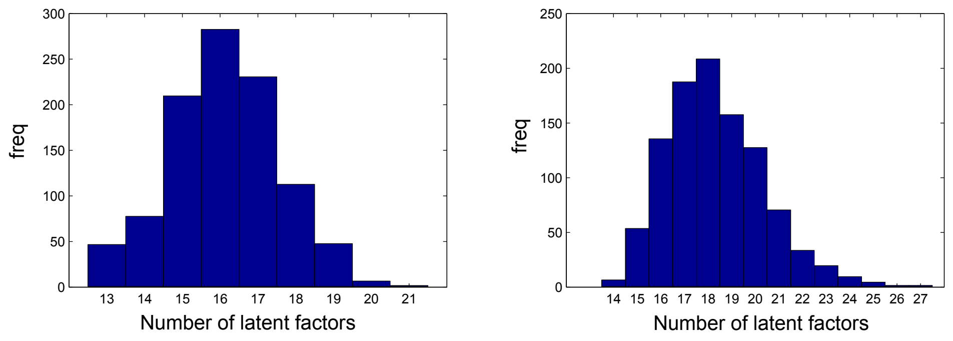
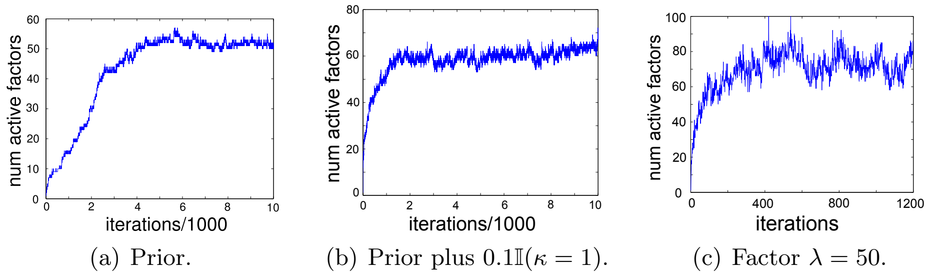


FIG 5. *Histograms of the number of latent features inferred by the nonparametric sparse FA sampler for the last 100 samples out of 1000.* Left: *With* $\alpha = 1$. Right: *Inferring* α.

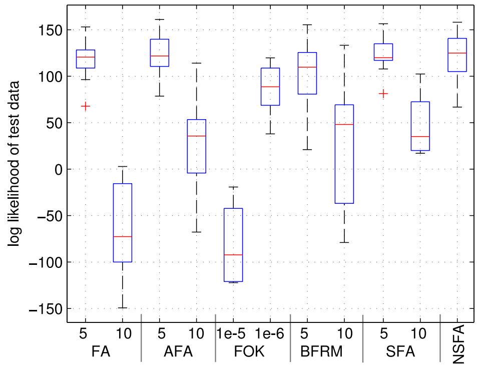
there is little to gain from sampling α in this data.

5.2. *Convergence.* NSFA can suffer from slow convergence if the number of new features is drawn from the prior. Figure 6 shows how the different proposals for κ_d effect how quickly the sampler reaches a sensible number of features. If we use the prior as the proposal distribution, mixing is very slow, taking around 5000 iterations to converge, as shown in Figure 6(a). If a mass of 0.1 is added at $\kappa_d = 1$ (see Equation 13), then the sampler reaches the equilibrium number of features in around 1500 iterations, as shown in Figure 6(b)). However, if we try to add features even faster, for example by setting the factor $\lambda = 50$ in Equation 13, then the sampling noise is greatly increased, as shown in Figure 6(c), and the computational cost also increases significantly because so many spurious features are proposed only to be rejected.

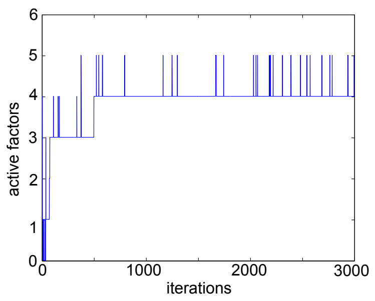


(a) Prior. (b) Prior plus $0.1\mathbb{I}(\kappa = 1)$. (c) Factor $\lambda = 50$.

FIG 6. *The effect of different proposal distributions for the number of new features.*



(a) Log likelihood of test data under each model based on the last 100 MCMC samples. The boxplots show variation across 10 different random splits of the data into training and test sets.



(b) Number of active latent features during a typical MCMC run of the NSFA model.

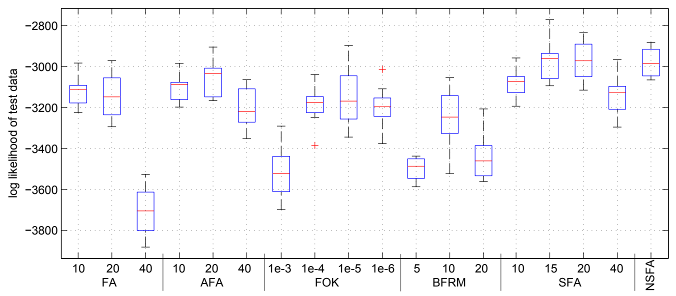
Fig 7. *Results on* E. Coli *time-series dataset from Kao et al. (2004) ($N = 24, D = 100$, 3000 MCMC iterations).*

5.3. *Biological data:* E. Coli *time-series dataset.* To assess the performance of each algorithm on the biological data where no ground truth is available, we calculated the test set log likelihood under the posterior. Ten percent of entries from $\mathbf{Y}$ were removed at random, ten times, to give ten datasets for inference. We do not use mean square error as a measure of predictive performance because of the large variation in the signal to noise ratio across gene expression level probes.

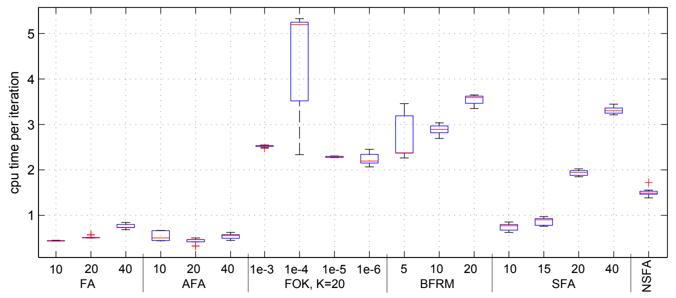
The test log likelihood achieved by the various algorithms on the *E. Coli* dataset from Kao *et al.* (2004), including 100 genes at 24 time-points, is shown in Figure 7(a). On this simple dataset incorporating sparsity doesn't improve predictive performance. Overfitting the number of latent factors does damage performance, although using the ARD or sparse prior alleviates the problem. Based on predictive performance of the finite models, five is a sensible number of features for this dataset: the NSFA model infers a median number of 4 features, with some probability of there being 5, as shown in Figure 7(b).

5.4. *Breast cancer dataset.* We assess these algorithms in terms of *predictive performance* on the breast cancer dataset of West *et al.* (2007), including 226 genes across 251 individuals. We find that all the finite models are sensitive to the choice of the number of factors, K. The samplers were found to have converged after around 1000 samples according to standard multiple chain convergence measures, so 3000 MCMC iterations were used for all models. The predictive log likelihood was calculated using the final 100 MCMC samples. Figure 8(a) shows test set log likelihoods for 10 random divisions of the data into training and test sets. Factor analysis (FA) shows significant overfitting as the number of latent features is increased from 20 to 40. Using the ARD prior prevents this overfitting (AFA), giving improved performance when using 20 features and only slightly reduced performance when 40 features are used. The sparse finite model (SFA) shows an advantage over AFA in terms of predictive performance as long as underfitting does not occur: performance is comparable when using only 10 features. However, the performance of SFA is sensitive to the choice of the number of factors, K. The performance of the sparse nonparametric model (NSFA) is comparable to the sparse finite model when an appropriate number of features is chosen, but avoids the time consuming model selection process. Fokoue's method (FOK) was run with $K = 20$ and various settings of the hyperparameter d which controls the overall sparsity of the solution. The model's predictive performance depends strongly on the setting of this parameter, with results approaching the performance of the sparse models (SFA and NSFA) for $d = 10^{-4}$. The performance of BFRM on this dataset is noticeably worse than the other sparse models.

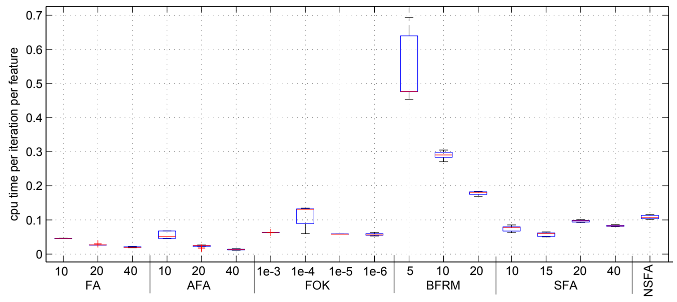
We now consider the *computation cost* of the algorithms. As described in Section 4, sampling $\mathbf{Z}$ and $\mathbf{G}$ takes order $O(NKD)$ operations per iteration, and sampling $\mathbf{X}$ takes $O(K^2 + K^3 + ND)$. However, for the moderate values encountered for datasets 1 and 2 the main computational cost is sampling the non-zero elements of $\mathbf{G}$, which takes $O((1-s)DK)$ where s is the sparsity of the model. Figure 8(c) shows the mean CPU time per iteration divided by the number of features at that iteration. Naturally, straight FA is the fastest, taking only around $0.025s$ per iteration per feature. The value increases slightly with increasing K, suggesting that here the $O(K^2D+K^3)$ calculation and inversion of $\boldsymbol{\lambda}$, the precision of the conditional on $\mathbf{X}$, must be contributing. The computational cost of adding the ARD prior is negligible (AFA). The CPU time per iteration is just over double for the sparse finite model (SFA), but the cost actually decreases with increasing K, because the sparsity of the solution increases to avoid overfitting. There are fewer non-zero elements of $\mathbf{G}$ to sample per feature, so the CPU time *per feature* decreases. The CPU time per iteration per feature for the non-parametric sparse model (NSFA) is somewhat higher than for the finite model because of the cost of the feature birth and death process. However, Figure 8(b) shows the absolute CPU time per iteration, where we see that the nonparametric model is only marginally more expensive than the finite model of



(a) Predictive performance: log likelihood of test (the 10% missing) data under each model based on the last 100 MCMC samples. Higher values indicate better performance. The boxplots show variation across 10 different random splits of the data into training and test sets.



(b) CPU time (in seconds) per iteration, averaged across the 3000 iteration run.



(c) CPU time (in seconds) per iteration divided by the number of features at that iteration, averaged across all iterations.

FIG 8. *Results on breast cancer dataset ($N = 251, D = 226$, 3000 MCMC iterations).*

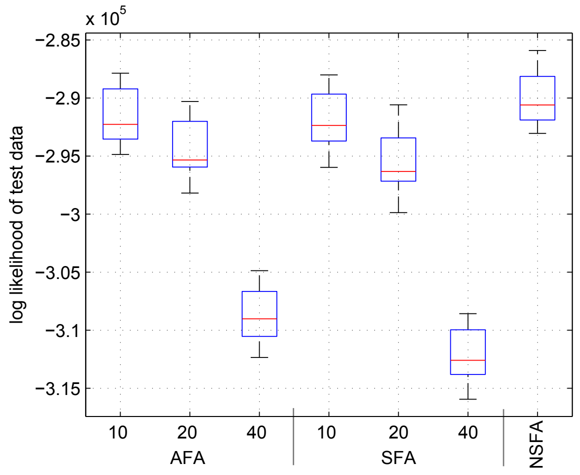


FIG 9. *Test set log likelihoods on Prostate cancer dataset from Yu et al. (2004), including 12557 genes across 171 individuals (1000 MCMC iterations).*

appropriate size $\hat{K} = 15$ and cheaper than choosing an unnecessarily large finite model (SFA with $K = 20, 40$). Fokoue's method (FOK) has comparable computational performance to the sparse finite model, but interestingly has increased cost for the optimal setting of $d = 10^{-4}$. The parameter space for FOK is continuous, making search easier but requiring a normal random variable for every element of $\mathbf{G}$. BFRM pays a considerable computational cost for both the hierarchical sparsity prior and the DP prior on $\mathbf{X}$. SPCA was not run on this dataset but results on the synthetic data in Section 5.1 suggest it is somewhat faster than the sampling methods, but not hugely so. The computational cost of SPCA is $ND^2 + mO(D^2K + DK^2 + D^3)$ in the $N > D$ case (where m is the number of iterations to convergence) and $ND^2 + mO(D^2K + DK^2)$ in the $D > N$ case taking the limit $\lambda \to \infty$. In either case an individual iteration of SPCA is more expensive than one sampling iteration of NSFA (since $K < D$) but fewer iterations will generally be required to reach convergence of SPCA than are required to ensure mixing of NSFA.

5.5. *Prostate cancer dataset.* Figure 9 shows the predictive performance of AFA, FOK and NSFA on the prostate cancer dataset of Yu *et al.* (2004), for ten random splits into training and test data. The boxplots show variation from ten random splits into training and test data. The large number of genes ($D = 12557$ across $N = 171$ individuals) in this dataset makes inference slower, but the problem is manageable since the computational complexity is linear in the number of genes. Despite the large number of genes, the appropriate number of latent factors, in terms of maximizing predictive performance, is still small, around 10 (NSFA infers a median of 12 factors). This may seem small relative to the number of genes, but it should be noted that the genes included in the breast cancer and *E. Coli* datasets are those capturing the most variability. Surprisingly, SFA actually performed slightly worse on this dataset than AFA. Both are highly sensitive to the number of latent factors chosen. NSFA however gives better predictive log likelihoods than either finite model for any fixed number of latent factors K. Running 1000 iterations of NSFA on this dataset takes under 8 hours. BFRM and FOK were impractically slow to run on this dataset.

6. Discussion.

We have seen that in both the *E. Coli* and breast cancer datasets that sparsity can improve predictive performance, as well as providing a more easily interpretable solution. Using the IBP to provide sparsity is straightforward, and allows the number of latent factors to be inferred within a well defined theoretical framework. This has several advantages over manually choosing the number of latent factors. Choosing too few latent factors damages predictive performance, as seen for the breast cancer dataset. Although choosing too many latent factors can be compensated for by using appropriate ARD-like priors, we find this is typically more computationally expensive than the birth and death process of the IBP. Manual model selection is an alternative but is time consuming. Finally we show that running NSFA on full gene expression datasets with 10000+ genes is feasible so long as the number of latent factors remains relatively small. An interesting direction for this research is how to incorporate prior knowledge, for example if certain transcription factors are known to regulate specific genes. Incorporating this knowledge could both improve the performance of the model and improve interpretability by associating latent variables with specific transcription factors. Another possibility is incorporating correlations in the Indian Buffet Process, which has been proposed for simpler models (Courville, Eck and Bengio, 2009; Doshi-Velez and Ghahramani, 2009). This would be appropriate in a gene expression setting where multiple transcription factors might be expected to share sets of regulated genes due to common motifs. Unfortunately, performing MCMC in all but the simplest of these models suffers from slow mixing.

Acknowledgements. We would like to thank the anonymous reviewers for helpful comments.

SUPPLEMENTARY MATERIAL

Supplement A: Graphs of precision and recall for the synthetic data experiment.
(doi: ??????; .pdf). The precision and recall of active elements of the **Z** matrix achieved by each algorithm (after thresholding for the non-sparse algorithms) on the synethic data experiment, described in Section 5.1. The results are consistent with the reconstruction error.

References.

Archambeau, C. and Bach, F. (2009). Sparse Probabilistic Projections. In *Proceedings of the Conference on Neural Information Processing Systems (NIPS)* (D. Koller, D. Schuurmans, Y. Bengio and L. Bottou, eds.) 73-80. MIT Press, Vancouver, Canada.

Courville, A. C., Eck, D. and Bengio, Y. (2009). An Infinite Factor Model Hierarchy Via a Noisy-Or Mechanism. In *Advances in Neural Information Processing Systems 21*. The MIT Press, Cambridge, MA, USA.

Doshi-Velez, F. and Ghahramani, Z. (2009). Correlated Nonparametric Latent Feature Models In *Conference on Uncertainty in Artificial Intelligence*.

Fevotte, C. and Godsill, S. J. (2006). A Bayesian Approach for Blind Separation of Sparse Sources. *Audio, Speech, and Language Processing, IEEE Transactions on* **14** 2174-2188.

Fokoue, E. (2004). Stochastic determination of the intrinsic structure in Bayesian factor analysis. Technical Report No. 17, Statistical and Applied Mathematical Sciences Institute.

Griffiths, T. L. and Ghahramani, Z. (2006). Infinite Latent Feature Models and the Indian Buffet Process. In *Advances in Neural Information Processing Systems 18*. The MIT Press, Cambridge, MA, USA.

Kao, K. C., Yang, Y.-L., Boscolo, R., Sabatti, C., Roychowdhury, V. and Liao, J. C. (2004). Transcriptome-based determination of multiple transcription regulator activities in Escherichia coli by using network component analysis. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)* **101** 641–646.

Kaufman, G. M. and Press, S. J. (1973). Bayesian factor analysis. Technical Report No. 662-73, Sloan School of Management, University of Chicago.

Knowles, D. and Ghahramani, Z. (2007). Infinite Sparse Factor Analysis and Infinite Independent Components Analysis. In *7th International Conference on Independent Component Analysis and Signal Separation* 381-388.

Meeds, E., Ghahramani, Z., Neal, R. and Roweis, S. (2006). Modeling Dyadic Data with Binary Latent Factors. In *Neural Information Processing Systems* **19**.

Rai, P. and Daumé III, H. (2008). The Infinite Hierarchical Factor Regression Model. In *Neural Information Processing Systems*.

Rowe, D. B. and Press, S. J. (1998). Gibbs Sampling and Hill Climbing in Bayesian Factor Analysis. Technical Report No. 255, Department of Statistics, University of California Riverside.

West, M., Chang, J., Lucas, J., Nevins, J. R., Wang, Q. and Carvalho, C. (2007). High-Dimensional Sparse Factor Modelling: Applications in Gene Expression Genomics Technical Report, ISDS, Duke University.

Witten, D. M., Tibshirani, R. and Hastie, T. (2009). A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics* **10** 515–534.

Yu, Y. P., Landsittel, D., Jing, L., Nelson, J., Ren, B., Liu, L., McDonald, C., Thomas, R., Dhir, R., Finkelstein, S., Michalopoulos, G., Becich, M. and Luo, J.-H. (2004). Gene expression alterations in prostate cancer predicting tumor aggression and preceding development of malignancy. *Journal of Clinical Oncology* **22** 2790–2799.

Zhang, Z., Chan, K. L., Kwok, J. T. and yan Yeung, D. (2004). Bayesian Inference on Principal Component Analysis using Reversible Jump Markov Chain Monte Carlo. In *Proceedings of the 19th National Conference on Artificial Intelligence, San Jose, California, USA* 372-377. AAAI Press.

Zou, H., Hastie, T. and Tibshirani, R. (2004). Sparse Principal Component Analysis. *Journal of Computational and Graphical Statistics* **15** 2006.

Cambridge University Engineering Department
Trumpington Street
Cambridge
CB2 1PZ
UK
E-mail: dak33@cam.ac.uk
zoubin@eng.cam.ac.uk