Nonparametric Bayesian sparse factor models with application to gene expression modeling

David Knowles and Zoubin Ghahramani (2011)

Report by Miguel Biron Lattes

Abstract

The purpose of this report is to give a thorough assessment of the Nonparametric Sparse Factor Analysis (NSFA) model. We start by giving a brief description of Factor Analysis, followed by an introduction to the Indian Buffet Process, a key component of the NSFA. Then, we give the definition of the model, which we use to derive updates for a Gibbs sampler. We discuss discrepancies between our derivations and the ones given by the authors. Next, we evaluate our implementation in a simulation using a toy dataset, where we find no issues with the program. We then proceed to apply it to a sub-sample of the MNIST dataset. Finally, we offer a critical review of the paper based on our findings.

1 Introduction

Let $Y \in \mathbb{R}^{D \times N}$ be a matrix of D observations of N variables. Factor Analysis (FA) proposes a latent, lower dimensional structure for Y, expressed by the following equation:

$$Y = GX + E \tag{1}$$

where X is a matrix of so called "latent factors", of size $K \times N$, G is a matrix of "loadings" or "weights" of the observations onto the latent factors (of size $D \times K$), and E is a matrix that can be thought of as the error that is incurred when using GX to represent Y. The earliest prototype of this model can be traced back to the work of the English psychologist Charles Spearman on what he termed the "general intelligence factor" [1]. Soon this model was improved upon, and it became extensively applied in a wide range of fields, most notably – due to its origin – in psychology [2], but also in chemistry [3], marketing research [4], education [5], and others.

Historically, the literature on FA was dominated by the frequentist perspective. However, since the mid 1990's there has been a growing interest in Bayesian approaches to this problem (see [6] for a review). One of the main issues faced by Bayesian statisticians working in this model was to develop methods to select the appropriate number of latent factors K. In this context, the Non-parametric Sparse Factor Analysis (NSFA) model [7, 8] proposes a fully Bayesian non-parametric approach to FA, which the authors claim has two key properties: first, it handles an unbounded number of factors, through the use of the Indian Buffet Process (IBP) [9, 10] in the prior of G, and second, the latent factors in the active set tend to show high sparsity (because of the structure of the IBP). Interestingly, a couple of other approaches similar to the NSFA were proposed concurrently, most notably [11] and [12].

This report is organized as follows: Section 2 introduces the theory behind the IBP, which, as previously stated, is a crucial component of the NSFA. Section 3 presents the NSFA formally, and gives a detailed derivation of a Gibbs sampler for it. We highlight and discuss differences in our results with respect to the ones that the authors obtain. Section 4 to checks our implementation of the NSFA by applying it to a toy dataset. Section 5 shows an application of the model to a real dataset. Finally, in Section 6 we discuss our findings and propose improvements based on them.

2 The Indian Buffet Process

The Indian Buffet Process (IBP) is a stochastic process that defines a probability distribution over the space of binary matrices with infinite columns, having the key property that the matrices it produces tend to have high sparsity. Binary matrices are useful in the context of describing the association of observed objects with hidden factors. In this setting, the d-th object is associated with the k-th factor if in the binary matrix Z, we have $z_{dk} = 1$.

2.1 The finite case

In order to derive this process, we will start by considering the case of finite binary matrices Z of size $D \times K$ generated by sampling from the following model:

$$\begin{aligned} \pi_k &\overset{iid}{\sim} \text{Beta}\left(\frac{\alpha}{K}, 1\right) \\ z_{dk}|\pi_k &\overset{ind}{\sim} \text{Bernoulli}(\pi_k) \end{aligned} \tag{2}$$

where α is known. Under this simple setup, the joint density of $\pi \triangleq \{\pi_k\}_k$ and $Z \triangleq \{z_{dk}\}_{dk}$ becomes:

$$\begin{aligned} p(Z, \pi) &= \prod_{k=1}^{K} p(\pi_k) \prod_{d=1}^{D} p(z_{dk}|\pi_k) \\ &= \prod_{k=1}^{K} \frac{\Gamma(\alpha/K+1)}{\Gamma(\alpha/K)} \pi_k^{\alpha/K-1} \prod_{d=1}^{D} \pi_k^{z_{dk}} (1-\pi_k)^{1-z_{dk}} \\ &= \prod_{k=1}^{K} \frac{\Gamma(\alpha/K+1)}{\Gamma(\alpha/K)} \pi_k^{\alpha/K-1+\sum_{d=1}^{D} z_{dk}} (1-\pi_k)^{D-\sum_{d=1}^{D} z_{dk}} \end{aligned}$$

We quickly notice that, because of the conjugacy of the Beta and the Bernoulli distributions, we obtain the following conditional distribution for π_k:

$$\pi_k | \{z_{dk}\}_{d=1}^{D} \overset{ind}{\sim} \text{Beta}\left(\frac{\alpha}{K} + \sum_{d=1}^{D} z_{dk}, D - \sum_{d=1}^{D} z_{dk} + 1\right) \tag{3}$$

Using this fact, we can now easily marginalize out $\pi = \{\pi_k\}$ from the model, by noting that:

$$p(\pi|Z) = \frac{p(\pi, Z)}{p(Z)} \Leftrightarrow p(Z) = \frac{p(\pi, Z)}{p(\pi|Z)}$$

Therefore:

$$\begin{aligned}
p(Z) &= \frac{\prod_{k=1}^{K} \frac{\Gamma(\alpha/K+1)}{\Gamma(\alpha/K)} \pi_k^{\alpha/K-1+\sum_{d=1}^{D} z_{dk}} (1-\pi_k)^{D-\sum_{d=1}^{D} z_{dk}}}{\prod_{k=1}^{K} \frac{\Gamma(\alpha/K+D+1)}{\Gamma(\alpha/K+\sum_{d=1}^{D} z_{dk})\Gamma(D-\sum_{d=1}^{D} z_{dk}+1)} \pi_k^{\alpha/K-1+\sum_{d=1}^{D} z_{dk}} (1-\pi_k)^{D-\sum_{d=1}^{D} z_{dk}}} \\
&= \prod_{k=1}^{K} \frac{\Gamma(\alpha/K+1)\Gamma(\alpha/K+\sum_{d=1}^{D} z_{dk})\Gamma(D-\sum_{d=1}^{D} z_{dk}+1)}{\Gamma(\alpha/K)\Gamma(\alpha/K+D+1)} \\
&= \prod_{k=1}^{K} \frac{(\alpha/K)\Gamma(\alpha/K+m_k)\Gamma(D-m_k+1)}{\Gamma(\alpha/K+D+1)} \\
&= \left(\frac{\alpha}{K}\right)^K \prod_{k=1}^{K} \frac{\Gamma(\alpha/K+m_k)(D-m_k)!}{\Gamma(\alpha/K+D+1)}
\end{aligned}$$

Thus:

$$p(Z) = \left(\frac{\alpha}{K}\right)^K \prod_{k=1}^{K} \frac{\Gamma(\alpha/K+m_k)(D-m_k)!}{\Gamma(\alpha/K+D+1)} \tag{4}$$

where $m_k \triangleq \sum_{d=1}^{D} z_{dk}$ is the number of 1's in the k-th column. Since the probability distribution only depends on these quantities, and these are not affected by a relabeling of the $\{z_{dk}\}$, we see that $p(Z)$ is exchangeable.

2.2 Equivalence classes

From the generative process described in 2, it is clear that we can obtain many distinct matrices that produce equivalent allocations between objects and latent factors. Indeed, take any Z generated by 2, and then shuffle its columns. It is clear that this new matrix could also have been generated by this model, and also that the underlying association is preserved because factors have only been relabeled. Moreover, since the collection of column sums $\{m_k\}$ is preserved under this transformation, both matrices have the same probability. Therefore, we would like to "marginalize out" this irrelevant distinction between otherwise equivalent matrices.

To this end, [10] propose the following strategy. First, let:

$$h_{d,k} \triangleq \sum_{j=1}^{d-1} 2^{d-1-j} z_{jk}$$

We call $h_{d,k}$ the "history" of the column k up to the d-th row. Note that $h_{d,k}$ is an integer corresponding to the decimal representation of the binary number formed by $(z_{1k}, ..., z_{d-1,k})$, taking the first row as the most significant place. Likewise, we can define the history for a whole column as:

$$h_k \triangleq \sum_{j=1}^{D} 2^{D-j} z_{jk}$$

which again is a decimal representation of the binary number formed by the k-th column, with the first row as the most significant bit. Note that there are $2^D - 1$ distinct numbers that can be specified with N bits. Thus, $h_k \in \{0, ..., 2^D - 1\}$. Also, h_k uniquely determines the k-th column of a binary matrix, because decimal representations are unique (this is not true for other summaries of columns, like m_k). Now, suppose we have sorted the columns of Z such that:

$$h_{a(1)} \geq h_{a(2)} \geq ... \geq h_{a(K)}$$

where $a(i)$ is a permutation of $\{1...K\}$. Let us define the function $lof(Z)$ such that:

$$lof(Z) \mapsto (z_{\cdot,a(1)}, z_{\cdot,a(2)}, ..., z_{\cdot,a(K)})$$

where $z_{\cdot,k}$ is the k-th column of Z. Therefore, $lof(Z)$ returns a matrix with the columns of Z arranged in descending order of their histories. In [10], these types of matrices are referred to as "left-ordered" (and hence the name of the function). We see that $lof(\cdot)$ maps many matrices to the same unique left-ordered representation of them. Therefore, a natural equivalence relation arises, with classes denoted by $[Z]$, such that another binary matrix Z^* belongs to this class if and only if:

$$lof(Z^*) = lof(Z)$$

Using $lof(\cdot)$, we will be able to resolve the non-uniqueness of the matrices produces by 2 that convey the same structural associations between objects and factors. Let K_h be the number of columns in the matrix Z with history h:

$$K_h \triangleq \sum_{k=1}^{K} \mathbb{I}(h_k = h)$$

Using this definition, we can write:

$$K = \sum_{h=0}^{2^D-1} K_h = K_0 + \sum_{h=1}^{2^D-1} K_h = K_0 + K_+$$

where K_0 is the number of columns with the 0 history; i.e, the columns with only 0 entries. Then, we can see that the number of binary matrices with the exact same counts of histories $\{K_h\}$ are:

$$\binom{K}{K_0, K_1, ..., K^{2^D-1}} = \frac{K!}{\prod_{h=0}^{2^D-1} K_h!}$$

This is the number of matrices that have the same left-ordered representation, and therefore it is the size of $[Z]$. Hence:

$$\mathbb{P}([Z]) = \sum_{Z^* \in [Z]} \mathbb{P}(Z^*) = \frac{K!}{\prod_{h=0}^{2^D-1} K_h!}\mathbb{P}(Z) = \frac{K!}{\prod_{h=0}^{2^D-1} K_h!}\left(\frac{\alpha}{K}\right)^K \prod_{k=1}^{K} \frac{\Gamma(\alpha/K + m_k)(D - m_k)!}{\Gamma(\alpha/K + D + 1)} \tag{5}$$

The second equality uses the fact that we mentioned earlier, which is that shuffling columns of Z preserves 4.

To simplify the expression above, note that for $m_k > 0$:

$$\begin{aligned}
\Gamma(\alpha/K + m_k) &= (\alpha/K + m_k - 1)\Gamma(\alpha/K + m_k - 1) \\
&= (\alpha/K + m_k - 1)(\alpha/K + m_k - 2)\Gamma(\alpha/K + m_k - 2) \\
&\vdots \\
&= \Gamma(\alpha/K)\prod_{l=1}^{m_k}(\alpha/K + m_k - l) \\
&= \Gamma(\alpha/K)\prod_{j=0}^{m_k-1}(\alpha/K + j) \\
&= (\alpha/K)\Gamma(\alpha/K)\prod_{j=1}^{m_k-1}(\alpha/K + j) \\
&= \Gamma(\alpha/K + 1)\prod_{j=1}^{m_k-1}(\alpha/K + j)
\end{aligned}$$

Using the same argument, we can see that:

$$\Gamma(\alpha/K + D + 1) = \Gamma(\alpha/K + 1)\prod_{j=1}^{D}(\alpha/K + j)$$

Therefore, if $m_k > 0$:

$$\begin{aligned}
\frac{\Gamma(\alpha/K + m_k)(D - m_k)!}{\Gamma(\alpha/K + D + 1)} &= \frac{(D - m_k)!\Gamma(\alpha/K + 1)\prod_{j=1}^{m_k-1}(\alpha/K + j)}{\Gamma(\alpha/K + 1)\prod_{j=1}^{D}(\alpha/K + j)} \\
&= \frac{(D - m_k)!\prod_{j=1}^{m_k-1}(\alpha/K + j)}{\prod_{j=1}^{D}(\alpha/K + j)}
\end{aligned}$$

On the other hand, if $m_k = 0$:

$$\frac{\Gamma(\alpha/K + m_k)(D - m_k)!}{\Gamma(\alpha/K + D + 1)} = \frac{\Gamma(\alpha/K)D!}{\Gamma(\alpha/K + 1)\prod_{j=1}^{D}(\alpha/K + j)} = (K/\alpha)\frac{D!}{\prod_{j=1}^{D}(\alpha/K + j)}$$

Putting these together:

$$\begin{aligned}
\prod_{k=1}^{K}\frac{\Gamma(\alpha/K + m_k)(D - m_k)!}{\Gamma(\alpha/K + D + 1)} &= \left[\prod_{k:m_k=0}\frac{K}{\alpha}\frac{D!}{\prod_{j=1}^{D}(\alpha/K + j)}\right]\left[\prod_{k:m_k>0}\frac{(D - m_k)!\prod_{j=1}^{m_k-1}(\alpha/K + j)}{\prod_{j=1}^{D}(\alpha/K + j)}\right] \\
&= \left[\frac{K}{\alpha}\frac{D!}{\prod_{j=1}^{D}(\alpha/K + j)}\right]^{K_0}\left[\frac{1}{\prod_{j=1}^{D}(\alpha/K + j)}\right]^{K_+}\left[\prod_{k:m_k>0}(D - m_k)!\prod_{j=1}^{m_k-1}(\alpha/K + j)\right] \\
&= \left[\frac{K}{\alpha}\right]^{K_0}\left[\frac{1}{\prod_{j=1}^{D}(\alpha/K + j)}\right]^{K}\frac{(D!)^K}{(D!)^{K_+}}\left[\prod_{k:m_k>0}(D - m_k)!\prod_{j=1}^{m_k-1}(\alpha/K + j)\right] \\
&= \left[\frac{K}{\alpha}\right]^{K_0}\left[\frac{1}{\prod_{j=1}^{D}(\alpha/K + j)}\right]^{K}(D!)^K\left[\prod_{k:m_k>0}\frac{(D - m_k)!\prod_{j=1}^{m_k-1}(\alpha/K + j)}{D!}\right]
\end{aligned}$$

Thus, going back and replacing this in the last line of 5, we get:

$$\begin{aligned}
\mathbb{P}([Z]) &= \frac{K!}{\prod_{h=0}^{2^D-1}K_h!}\left(\frac{\alpha}{K}\right)^K\left[\frac{K}{\alpha}\right]^{K_0}\left[\frac{1}{\prod_{j=1}^{D}(\alpha/K + j)}\right]^{K}(D!)^K\left[\prod_{k:m_k>0}\frac{(D - m_k)!\prod_{j=1}^{m_k-1}(\alpha/K + j)}{D!}\right] \\
&= \frac{\alpha^{K_+}}{\prod_{h=1}^{2^D-1}K_h!}\frac{K!}{K_0!K^{K_+}}\left[\frac{D!}{\prod_{j=1}^{D}(\alpha/K + j)}\right]^{K}\left[\prod_{k=1}^{K_+}\frac{(D - m_k)!\prod_{j=1}^{m_k-1}(\alpha/K + j)}{D!}\right]
\end{aligned} \tag{6}$$

Note that in the second line we have assumed that the columns with non-zero elements are the first K_+ columns, since we are concerned with the equivalence class of left-ordered matrices.

2.3 Taking the limit

We are now ready to derive the limit as $K \to \infty$ of the probability distribution for the equivalence classes of left-ordered binary matrices. Of course, since $K = K_0 + K_+$, and we are interested in obtaining sparse representations, it is clear that the limit we are interested in is $K_0 \to \infty$ as K_+ remains finite. From this, it follows that the first term in the last line of 6 is not affected by the limit. For the last term, we obtain:

$$\lim_{K\to\infty}\left[\prod_{k=1}^{K_+}\frac{(D - m_k)!\prod_{j=1}^{m_k-1}(\alpha/K + j)}{D!}\right] = \prod_{k=1}^{K_+}\frac{(D - m_k)!\prod_{j=1}^{m_k-1}j}{D!} = \prod_{k=1}^{K_+}\frac{(D - m_k)!(m_k - 1)!}{D!}$$

For the second term in 6, we note that:

$$\begin{aligned}
\frac{K!}{K_0! K^{K_+}} &= \frac{K_0!(K_0+1)(K_0+2)...(K-1)K}{K_0! K^{K_+}} \\
&= \frac{\overbrace{(K-(K_+-1))(K-(K_+-2))...(K-1)K}^{K_+ \text{ factors}}}{K^{K_+}} \\
&= \frac{K^{K_+} + \text{lower order terms on } K}{K^{K_+}}
\end{aligned}$$

Therefore:

$$\lim_{K\to\infty} \frac{K!}{K_0! K^{K_+}} = \lim_{K\to\infty} \frac{K^{K_+}}{K^{K_+}} = 1$$

Next, for the 3rd term in 6:

$$\begin{aligned}
\left[\frac{D!}{\prod_{j=1}^{D}(\alpha/K+j)}\right]^K = \left[\prod_{j=1}^{D}\frac{j}{(\alpha/K+j)}\right]^K = \left[\prod_{j=1}^{D}\frac{j}{j(1+(\alpha j)/K)}\right]^K = \prod_{j=1}^{D}\frac{1}{\left(1+\frac{\alpha j}{K}\right)^K} &\overset{K\to\infty}{=} \prod_{j=1}^{D} e^{-\alpha j} \\
&= e^{-\alpha H_D}
\end{aligned}$$

where $H_D = \sum_{j=1}^{D} 1/j$. Finally, we see that the limit of $\mathbb{P}([Z])$ becomes:

$$\lim_{K\to\infty} \mathbb{P}([Z]) = \frac{\alpha^{K_+}}{\prod_{h=1}^{2^D-1} K_h!} e^{-\alpha H_D} \prod_{k=1}^{K_+} \frac{(D-m_k)!(m_k-1)!}{D!} \tag{7}$$

One important property of this limiting distribution is that the rows are exchangeable under this measure. Indeed, suppose we permute the rows of a given matrix Z. The number of active columns is trivially conserved. Also, since the column counts m_k are not affected, the terms inside the product on the right remain constant. Finally, fix a history h in the original matrix, and consider its count K_h. By permuting rows, K_h could arbitrarily change. However, all the columns with history h will be equally affected by the permutation, so that history h' created by permuting h will have the same count K_h. This will happen for every history. Hence, $\prod_h K_h!$ is not affected by the permutation, and thus none of the terms are affected by it.

Note that Equation 7 differs slightly from Equation 7 in [8], as they have $N!$ instead of $D!$ in the denominator inside the product. This is probably a typo, since N (the number of columns of the data Y) is not associated in any sense with Z.

2.4 Construction as a stochastic process

According to [9], the IBP can be derived starting from a simple stochastic process: assume there are N customers in line at an Indian buffet, which has (countably) infinitely many dishes available. The first client will draw a number from a Poisson(α), and then try the dishes in order until he reaches said number. In this analogy, trying a dish means setting $z_{dk} = 1$. The d-th client ($d > 1$) will first choose to grab food from the dishes that have already been tried by other customers. He will do so independently with $z_{dk} \sim \text{Bernoulli}(m_{dk}/d)$, where m_{dk} is the number of clients before i that have tried the k-th dish. Finally, he will try a number of new dishes, by drawing from a $\text{Poisson}(\alpha/d)$. This process continues until all clients have selected their dishes.

The authors in [10] argue that, if one corrects the non-exchangeability of the above process (the order in which the clients come affects the result), then one is left with a process that generates assignment matrices with probability distribution equal to 7, which we already have seen is exchangeable.

Once it is established that the IBP can be modified to be exchangeable, it is reasonable to ask what the mixing distribution is under which the process becomes independent, since this is guaranteed to exist by de Finetti's theorem [13]. The answer was given by [14], where it is shown that the mixing distribution corresponds to the Beta Process. They also show that sampling from the exchangeable IBP is equivalent to sampling from a Beta-Bernoulli process in some measurable space $(\Omega, \mathcal{B})$, where the base measure of the Beta process is such that $B_0(\Omega) = \alpha$, and the concentration parameter is $c = 1$.

2.5 A Gibbs sampler for the IBP

As a final step, we derive the full conditional distribution for an element z_{dk} of a matrix Z. This will allow us to construct a Markov Chain with stationary distribution equal to 4, so that we can obtain samples from the IBP prior starting from any binary matrix. We will leverage this Gibbs updates for Z when constructing a sampler for the factor model.

We will build first a Gibbs sampler for the finite case, and then proceed to take the limit as $K \to \infty$.

2.5.1 The finite case

Let $z_{-dk} \triangleq Z \setminus \{z_{dk}\}$. By Bayes' theorem:

$$\mathbb{P}(z_{dk}|z_{-dk}) = \frac{\mathbb{P}(z_{dk}, z_{-dk})}{\mathbb{P}(z_{-dk})} = \frac{\mathbb{P}(Z)}{\mathbb{P}(z_{-dk})}$$

The numerator of the RHS is Equation 4. We need only compute the denominator:

$$
\begin{aligned}
\mathbb{P}(z_{-dk}) &= \sum_{z_{dk}=0}^{1} \mathbb{P}(Z) \\
&= \sum_{z_{dk}=0}^{1} \prod_{l=1}^{K} \frac{(\alpha/K)\Gamma(\alpha/K + m_l)(D - m_l)!}{\Gamma(\alpha/K + D + 1)} \\
&= \prod_{l \neq k} \frac{(\alpha/K)\Gamma(\alpha/K + m_l)(D - m_l)!}{\Gamma(\alpha/K + D + 1)} \sum_{z_{dk}=0}^{1} \frac{(\alpha/K)\Gamma(\alpha/K + m_k)(D - m_k)!}{\Gamma(\alpha/K + D + 1)}
\end{aligned}
$$

Therefore:

$$
\begin{aligned}
\mathbb{P}(z_{dk}|z_{-dk}) &= \frac{\prod_{l=1}^{K} \frac{(\alpha/K)\Gamma(\alpha/K+m_l)(D-m_l)!}{\Gamma(\alpha/K+D+1)}}{\prod_{l \neq k} \frac{(\alpha/K)\Gamma(\alpha/K+m_l)(D-m_l)!}{\Gamma(\alpha/K+D+1)} \sum_{z_{dk}=0}^{1} \frac{(\alpha/K)\Gamma(\alpha/K+m_k)(D-m_k)!}{\Gamma(\alpha/K+D+1)}} \\
&= \frac{\Gamma(\alpha/K + m_{-dk} + z_{dk})(D - m_{-dk} - z_{dk})!}{\sum_{z_{dk}=0}^{1} \Gamma(\alpha/K + m_{-dk} + z_{dk})(D - m_{-dk} - z_{dk})!}
\end{aligned}
$$

where $m_{-dk} \triangleq m_k - z_{dk}$. From here, we can now obtain:

$$
\begin{aligned}
\mathbb{P}(z_{dk} = 1|z_{-dk}) &= \frac{\Gamma(\alpha/K + m_{-dk} + 1)(D - m_{-dk} - 1)!}{\Gamma(\alpha/K + m_{-dk})(D - m_{-dk})! + \Gamma(\alpha/K + m_{-dk} + 1)(D - m_{-dk} - 1)!} \\
&= \frac{(\alpha/K + m_{-dk})\Gamma(\alpha/K + m_{-dk})(D - m_{-dk} - 1)!}{\Gamma(\alpha/K + m_{-dk})(D - m_{-dk} - 1)!(D - m_{-dk}) + (\alpha/K + m_{-dk})\Gamma(\alpha/K + m_{-dk})(D - m_{-dk}} \\
&= \frac{(\alpha/K + m_{-dk})}{(D - m_{-dk}) + (\alpha/K + m_{-dk})} \\
&= \frac{\alpha/K + m_{-dk}}{\alpha/K + D}
\end{aligned}
$$

Thus, in the finite case, we have the update rule:

$$
\mathbb{P}(z_{dk} = 1|z_{-dk}) = \frac{\alpha/K + m_{-dk}}{\alpha/K + D} \tag{8}
$$

Using the above expression, it is easy to derive a Gibbs sampler for the finite case. Starting with a random binary matrix, we apply the update rule sequentially by rows, until we reach the end of the matrix, from where we then proceed to start a new iteration. The Markov chain produced by this process will have as a stationary distribution the expression in Equation 4.

We could slightly alter the proposed Gibbs sampler so as to gain more insight into the infinite case. Indeed, for each object d, we could deal separately with the columns where $m_k > 0$ and $m_k = 0$; that is, we distinguish between the columns that are empty (K_0 in total) and the ones that are not (K_+ in total). Note that $m_k = 0$ implies $m_{-dk} = 0$ for all d, and that the expression in Equation 8 applies regardless of the value of m_{-dk}. For each row d, the modified Gibbs sampler proceeds in the following way: for columns with $m_k > 0$, use the update rule in Equation 8. However, for empty columns, applying that update rule independently is equivalent to drawing a number κ_d of new columns from the K_0 available, with probability distribution:

$$\kappa_d | z_{-dk} \sim \text{Binomial}\left(K_0, \frac{\alpha/K}{\alpha/K + D}\right) \tag{9}$$

Then, we could sample without replacement κ_d empty columns of Z and set $z_{dk} = 1$ for them. Each of these configurations has probability $\binom{K_0}{\kappa_d}^{-1} = \frac{\kappa_d!(K_0-\kappa_d)!}{K_0}$.

2.5.2 A Gibbs sampler for the infinite case

Using the second Gibbs sampler, obtaining an extension for the infinite case is straightforward. Indeed, the update rule for active columns becomes:

$$\lim_{K\to\infty} \mathbb{P}(z_{dk} = 1 | z_{-dk}) = \lim_{K\to\infty} \frac{\alpha/K + m_{-dk}}{\alpha/K + D} = \frac{m_{-dk}}{D} \tag{10}$$

Note that this expression indicates that for active columns which are only associated with the d-th row (i.e, $m_{-dk} = 0$), the column is eliminated with probability one.

For inactive columns, we see that by taking the limit of the expression in 9 (and with a slight abuse of notation), we obtain:

$$\kappa_d | z_{-dk} \overset{\mathcal{D}}{\to} \lim_{K\to\infty} \text{Binomial}\left(K_0, \frac{\alpha/K}{\alpha/K + D}\right) = \text{Poisson}\left(\lim_{K\to\infty} \frac{\alpha(K_0/K)}{\alpha/K + D}\right) = \text{Poisson}\left(\frac{\alpha}{D}\right) \tag{11}$$

Note that this distribution is the same as the one the last client uses to select new plates after sampling the foods already selected by others. This is a consequence of the exchangeability of the IBP prior. The above tells us that, in the infinite case, after we finish updating the components of the d-th row associated with active columns, we proceed to initialize κ_d of the leftmost inactive columns (since we are concerned only with the lof equivalence class) by setting $z_{dk} = 1$ for them while leaving everything else equal to 0.

3 Nonparametric Sparse Factor Analysis

By leveraging the IBP prior, the authors in [8] propose a model for Factor Analysis (Equation 1), called the Nonparametric Sparse Factor Analysis (NSFA) model, with two interesting properties: an unbounded number of latent factors, and a sparsity inducing prior for the mixing matrix. The structure of the model is:

$$\begin{aligned}
\alpha &\sim \text{Gamma}(e, f) \\
Z|\alpha &\sim \text{IBP}(\alpha) \\
d &\sim \text{Gamma}(c_0, d_0) \\
\lambda_k|d &\overset{iid}{\sim} \text{Gamma}(c, d) \\
g_{dk}|Z_{dk}, \lambda_k &\overset{ind}{\sim} z_{dk}\mathcal{N}(g_{dk}; 0, \lambda_k^{-1}) + (1 - z_{dk})\delta_0(g_{dk}) \\
x_{kn} &\overset{iid}{\sim} \mathcal{N}(0, 1) \\
b &\sim \text{Gamma}(a_0, b_0) \\
\psi_d|b &\overset{iid}{\sim} \text{Inv-Gamma}(a, b) \\
y_n|G, x_n, \Psi &\overset{ind}{\sim} \mathcal{N}(Gx_n, \Psi)
\end{aligned} \tag{12}$$

where $\Psi \triangleq \text{diag}(\psi_1, ..., \psi_D)$. Note that the prior on g_{dk} uses a type of spike-and-slab prior [15]. In more recent works [11, 16], this prior has been made independent from Z, by making the following modifications:

$$g_{dk}|\lambda_k \overset{ind}{\sim} \mathcal{N}(0, \lambda_k^{-1}) \qquad\qquad y_n|Z, G, x_n, \Psi \overset{ind}{\sim} \mathcal{N}((G \odot Z)x_n, \Psi)$$

where $\odot$ is the Hadamard product. This formulation is equivalent to the one proposed by the authors, but allows the prior on g_{dk} to have a density, which makes the model more tractable.

3.1 Gibbs sampler for the NSFA

3.1.1 Update for the mixture coefficients

Since the prior on g_{dk} does not have a density (because of the spike at 0), we need to integrate this variable out in order to be able to update z_{dk}, and then use this to sample g_{dk}. Since z_{dk} is a binary variable, it suffices to find the ratio:

$$\frac{\mathbb{P}(z_{dk} = 1|Y, -)}{\mathbb{P}(z_{dk} = 0|Y, -)}$$

because the constraint of the numerator and denominator summing to 1 gives the complete distribution. Note that:

$$\frac{\mathbb{P}(z_{dk} = 1|Y, -)}{\mathbb{P}(z_{dk} = 0|Y, -)} = \frac{p(Y|z_{dk} = 1, -)\mathbb{P}(z_{dk} = 1|z_{-dk})}{p(Y|z_{dk} = 0, -)\mathbb{P}(z_{dk} = 0|z_{-dk})} \tag{13}$$

The ratio on the right can be obtained from 10, so we only need to calculate the first ratio on the RHS. We start with the denominator:

$$\begin{aligned} p(Y|z_{dk}=0,-) &= p(Y|g_{dk}=0, g_{-dk}, X, \Psi) \\ &= \prod_{n=1}^{N} (2\pi)^{-D/2} |\Psi|^{-1/2} \exp\left(-\frac{1}{2}(y_n - G^* x_n)^T \Psi^{-1} (y_n - G^* x_n)\right) \\ &= (2\pi)^{-DN/2} |\Psi|^{-N/2} \prod_{n=1}^{N} \exp\left(-\frac{1}{2} \hat{E}_{:n}^T \Psi^{-1} \hat{E}_{:n}\right) \end{aligned} \tag{14}$$

where G^* is equal to G but has a 0 in the (d,k) position, and $\hat{E} \triangleq Y - G^* X$. We move on now to the numerator of the first ratio in the RHS of 13:

$$\begin{aligned} p(Y|z_{dk}=1,-) &= \int_{g_{dk}} p(Y|g_{dk}, g_{-dk}, X, \Psi) p(g_{dk}|\lambda_k) dg_{dk} \\ &= \int_{g_{dk}} p(g_{dk}|\lambda_k) \prod_{n=1}^{N} (2\pi)^{-D/2} |\Psi|^{-1/2} \exp\left(-\frac{1}{2}(y_n - Gx_n)^T \Psi^{-1} (y_n - Gx_n)\right) dg_{dk} \end{aligned}$$

In order to make sense of the integral above, it will be useful to decompose the terms inside the $\exp(\cdot)$:

$$(y_n - Gx_n)^T \Psi^{-1} (y_n - Gx_n) = y_n^T \Psi^{-1} y_n - 2 y_n^T \Psi^{-1} G x_n + (Gx_n)^T \Psi^{-1} G x_n$$

Note that:

$$y_n^T \Psi^{-1} G x_n = \frac{y_{dn} g_{dk} x_{kn}}{\psi_d} + y_n^T \Psi^{-1} G^* x_n$$

Also:

$$\begin{aligned}
(Gx_n)^T\Psi^{-1}Gx_n &= \sum_{l=1}^{K_+}\sum_{u=1}^{K_+} x_{ln}x_{un}\sum_{v=1}^{D}\frac{g_{vk}g_{vl}}{\psi_v}\\
&= \sum_{l=1}^{K_+}\sum_{u=1}^{K_+} x_{ln}x_{un}\left[\frac{g_{dk}g_{dl}}{\psi_d}+\sum_{v\neq d}\frac{g_{vk}g_{vl}}{\psi_v}\right]\\
&= \sum_{l=1}^{K_+}\sum_{u=1}^{K_+} x_{ln}x_{un}\frac{g_{dk}g_{dl}}{\psi_d}+\sum_{l=1}^{K_+}\sum_{u=1}^{K_+} x_{ln}x_{un}\sum_{v\neq d}\frac{g_{vk}g_{vl}}{\psi_v}\\
&= \sum_{l=1}^{K_+}\left[x_{ln}x_{kn}\frac{g_{dk}g_{dl}}{\psi_d}+\sum_{u\neq k}x_{ln}x_{un}\frac{g_{dk}g_{dl}}{\psi_d}\right]+\sum_{l=1}^{K_+}\sum_{u=1}^{K_+} x_{ln}x_{un}\sum_{v\neq d}\frac{g_{vk}g_{vl}}{\psi_v}\\
&= \frac{g_{dk}^2x_{kn}^2}{\psi_d}+\frac{g_{dk}x_{kn}}{\psi_d}(G^*x_n)_d+\sum_{l\neq k}\left[x_{ln}x_{kn}\frac{g_{dk}g_{dl}}{\psi_d}+\sum_{u\neq k}x_{ln}x_{un}\frac{g_{dk}g_{dl}}{\psi_d}\right]\\
&+\sum_{l=1}^{K_+}\sum_{u=1}^{K_+} x_{ln}x_{un}\sum_{v\neq d}\frac{g_{vk}g_{vl}}{\psi_v}\\
&= \frac{g_{dk}^2x_{kn}^2}{\psi_d}+\frac{g_{dk}x_{kn}}{\psi_d}(G^*x_n)_d+\frac{g_{dk}x_{kn}}{\psi_d}\sum_{l\neq k}x_{ln}g_{dl}+\sum_{l\neq k}\sum_{u\neq k}x_{ln}x_{un}\frac{g_{dk}g_{dl}}{\psi_d}\\
&+\sum_{l=1}^{K_+}\sum_{u=1}^{K_+} x_{ln}x_{un}\sum_{v\neq d}\frac{g_{vk}g_{vl}}{\psi_v}\\
&= \frac{g_{dk}^2x_{kn}^2}{\psi_d}+2\frac{g_{dk}x_{kn}}{\psi_d}(G^*x_n)_d+(G^*x_n)^T\Psi^{-1}G^*x_n
\end{aligned}$$

Therefore:

$$\begin{aligned}
(y_n-Gx_n)^T\Psi^{-1}(y_n-Gx_n) &= \frac{x_{kn}^2g_{dk}^2}{\psi_d}-2\frac{x_{kn}g_{dk}}{\psi_d}\left[y_{dn}-(G^*x_n)_d\right]+\hat{E}_{:n}^T\Psi^{-1}\hat{E}_{:n}\\
&= \frac{x_{kn}^2g_{dk}^2}{\psi_d}-2\frac{x_{kn}g_{dk}}{\psi_d}\hat{E}_{dn}+\hat{E}_{:n}^T\Psi^{-1}\hat{E}_{:n}
\end{aligned} \tag{15}$$

Putting all of the above together, we obtain:

$$p(Y|z_{dk}=1,-)=p(Y|z_{dk}=0,-)\int_{g_{dk}}p(g_{dk}|\lambda_k)\prod_{n=1}^{N}\exp\left(-\frac{1}{2}\left\{\frac{x_{kn}^2g_{dk}^2}{\psi_d}+-2\frac{x_{kn}g_{dk}}{\psi_d}\hat{E}_{dn}\right\}\right)dg_{dk}$$

Therefore:

$$
\begin{aligned}
\frac{p(Y|z_{dk}=1,-)}{p(Y|z_{dk}=0,-)} &= (2\pi\lambda_k^{-1})^{-1/2}\int_{g_{dk}}\prod_{n=1}^{N}\exp\left(-\frac{1}{2}\left\{\frac{\lambda_k}{N}g_{dk}^2+\frac{x_{kn}^2 g_{dk}^2}{\psi_d}+-2\frac{x_{kn}g_{dk}}{\psi_d}\hat{E}_{dn}\right\}\right)dg_{dk}\\
&= (2\pi\lambda_k^{-1})^{-1/2}\int_{g_{dk}}\prod_{n=1}^{N}\exp\left(-\frac{1}{2}\left\{g_{dk}^2\left[\frac{x_{kn}^2}{\psi_d}+\frac{\lambda_k}{N}\right]+-2\frac{x_{kn}g_{dk}}{\psi_d}\hat{E}_{dn}\right\}\right)dg_{dk}\\
&= (2\pi\lambda_k^{-1})^{-1/2}\int_{g_{dk}}\exp\left(-\frac{1}{2}\left\{g_{dk}^2\sum_{n=1}^{N}\left[\frac{x_{kn}^2}{\psi_d}+\frac{\lambda_k}{N}\right]+-2g_{dk}\left(\frac{1}{\psi_d}\sum_{n=1}^{N}x_{kn}\hat{E}_{dn}\right)\right\}\right)dg_{dk}\\
&= (2\pi\lambda_k^{-1})^{-1/2}\int_{g_{dk}}\exp\left(-\frac{1}{2}\left\{g_{dk}^2\lambda+-2g_{dk}\lambda\mu\right\}\right)dg_{dk}
\end{aligned}
$$

with:

$$
\lambda \triangleq \sum_{n=1}^{N}\left[\frac{x_{kn}^2}{\psi_d}+\frac{\lambda_k}{N}\right]=\frac{1}{\psi_d}X_{k:}^T X_{k:}+\lambda_k \qquad \mu \triangleq \frac{1}{\lambda\psi_d}\sum_{n=1}^{N}x_{kn}\hat{E}_{dn}=\frac{1}{\lambda\psi_d}X_{k:}^T\hat{E}_{d:} \tag{16}
$$

Finally, we can obtain a simple form for the ratios of the likelihoods conditioned on z_{dk}:

$$
\begin{aligned}
\frac{p(Y|z_{dk}=1,-)}{p(Y|z_{dk}=0,-)} &= (2\pi\lambda_k^{-1})^{-1/2}\int_{g_{dk}}\exp\left(-\frac{1}{2}\left\{g_{dk}^2\lambda+-2g_{dk}\lambda\mu\right\}\right)dg_{dk}\\
&= (2\pi\lambda_k^{-1})^{-1/2}\exp\left(\frac{1}{2}\lambda\mu^2\right)(2\pi\lambda^{-1})^{1/2} \qquad (17)\\
&= \sqrt{\frac{\lambda_k}{\lambda}}\exp\left(\frac{1}{2}\lambda\mu^2\right)
\end{aligned}
$$

Here we used the normalizing constant of a normal random variable with known mean μ and precision λ. Now, we can go back to Equation 13 to obtain:

$$
\frac{\mathbb{P}(z_{dk}=1|Y,-)}{\mathbb{P}(z_{dk}=0|Y,-)}=\sqrt{\frac{\lambda_k}{\lambda}}\exp\left(\frac{1}{2}\lambda\mu^2\right)\frac{m_{-dk}}{D-m_{-dk}}
$$

We note that the last ratio on the RHS of this expression (ratio for the priors) is slightly different from Equation (12) in [8], as they show a $N-1$ term instead of the corresponding number of rows of Z, which is D[1]. We attribute this to a typo, as there are a couple of others throughout the paper, because D can be arbitrarily different from N (they are dictated by the data Y). Moreover, a paper cited by the authors [17], shows that our derivation is correct (use $\beta=1$ in Eq. 3). Thus, we use our result instead for experiments. From this, it's straightforward to get the Gibbs update for z_{dk}:

$$
\mathbb{P}(z_{dk}=1|Y,-)=\frac{\sqrt{\frac{\lambda_k}{\lambda}}\exp\left(\frac{1}{2}\lambda\mu^2\right)\frac{m_{-dk}}{D-m_{-dk}}}{1+\sqrt{\frac{\lambda_k}{\lambda}}\exp\left(\frac{1}{2}\lambda\mu^2\right)\frac{m_{-dk}}{D-m_{-dk}}} \tag{18}
$$

[1]Note that in [8], Equation (8) does have the right expression on the RHS. The LHS has a minor typo, though (using z_{-kn} instead of z_{-kt}).

Having sampled z_{dk}, we can move on to obtaining an update for g_{dk}. Indeed, if $z_{dk} = 0$, we simply set $g_{dk} = 0$. Otherwise:

$$\begin{aligned}
p(g_{dk}|z_{dk} = 1, Y, -) &= \frac{p(Y|-)p(g_{dk}|z_{dk} = 1, -)}{p(Y|z_{dk} = 1, -)} \\
&= \frac{(2\pi\lambda_k^{-1})^{-1/2} e^{-\frac{\lambda_k g_{dk}^2}{2}} \prod_{n=1}^{N} (2\pi)^{-D/2} |\Psi|^{-1/2} \exp\left(-\frac{1}{2}(y_n - Gx_n)^T \Psi^{-1} (y_n - Gx_n)\right)}{\sqrt{\frac{\lambda_k}{\lambda}} \exp\left(\frac{1}{2}\lambda\mu^2\right) \prod_{n=1}^{N} (2\pi)^{-D/2} |\Psi|^{-1/2} \exp\left(-\frac{1}{2}\hat{E}_{:n}^T \Psi^{-1} \hat{E}_{:n}\right)} \\
&= (2\pi\lambda^{-1})^{-1/2} \exp\left(-\frac{1}{2}\lambda\mu^2 - \frac{1}{2}\lambda g_{dk}^2 + \lambda\mu g_{dk}\right) \\
&= (2\pi\lambda^{-1})^{-1/2} \exp\left(-\frac{\lambda}{2}(g_{dk} - \mu)^2\right)
\end{aligned}$$

Therefore, we conclude that the Gibbs update for the mixture components is:

$$g_{dk}|Y, - \sim z_{dk}\mathcal{N}(g_{dk}; \mu, \lambda^{-1}) + (1 - z_{dk})\delta_0(g_{dk}) \tag{19}$$

3.1.2 Adding new features

Following the same structure presented in Section 2.5.2, once we have updated the components of the matrix Z of a given object d associated with the active columns (including dropping the ones which were only used at row d), we proceed to sample κ_d new columns for said object and set $z_{dk} = 1$ for those positions. Note that for rows distinct to d, these new columns will be left unchanged at 0.

However, we now need to account for both the latent features X, and the mixture matrix G, which will also grow as the active components of Z expand. In other words, we need to consider a joint update for (κ_d, X', g'^T), where X' is a matrix of size $\kappa_d \times N$ composed of the new latent features to be added to X, and g'^T is a vector of size $\kappa_d \times 1$ containing the new elements of G at row d. Note that we need not worry about sampling new elements of G at rows different of d, because, as stated in the preceding paragraph, Z is zero in those places, and so the corresponding elements in G are also 0 (recall its prior).

Now, the authors of the paper argue that while it is not possible to integrate out both X' and g'^T from this update, it is feasible to do that for either of them. They state that, because the number of elements in X' is far more than in g'^T, we should marginalize the former. Therefore, we consider the random variable $\xi \triangleq (\kappa_d, g'^T)$.

Since the full conditional for ξ does not have a closed form, the authors include a Metropolis-Hastings step within the Gibbs sampler [18]. As a proposal, their first attempt is to sample from the priors of κ_d and g'^T:

$$J(\xi) = p(\kappa_d|\alpha)p(g'^T|\kappa_d, \lambda_k) = \text{Poisson}(\kappa_d; \gamma)\mathcal{N}(g'^T; 0_{\kappa_d}, \lambda_k^{-1} I_{\kappa_d})$$

with $\gamma \triangleq \frac{\alpha}{D}$. Note that this equation again differs from the one presented in [8], as the authors use $\gamma = \alpha/(D-1)$ instead. We believe that this is likely a typo, as the paper from which the authors draw the idea for the MH step has the same expression as us (use $\beta = 1$ in Eq. 6 of [17]). Hence, we will use our derivation in the implementation.

More importantly, this equation reveals something problematic with the proposal: in order to draw g'^T (a vector of length κ_d), we actually need to first sample κ_d precision values λ_k. Otherwise, it is unclear what variance will be used for sampling g'^T. Nevertheless, this can be easily fixed by expanding the proposal to $\xi = (\kappa_d, \lambda', g'^T)$, where λ' is a vector of length κ_d with the required precisions. If we also use the prior as the proposal, we obtain:

$$J(\xi) = \text{Poisson}(\kappa_d; \gamma) \prod_{k=1}^{\kappa_d} \text{Gamma}(\lambda_k; c, d) \mathcal{N}(g'_k; 0, (\lambda'_k)^{-1})$$

Now, in spite of the above, the authors find that simply using the prior for κ_d results in proposals that are too often equal to zero. Because of this, they modify it to allow for greater flexibility:

$$J(\kappa_d) = (1 - \pi)\text{Poisson}(\kappa_d; \lambda\gamma) + \pi\mathbb{I}(\kappa_d = 1) \tag{20}$$

where $\pi \in [0, 1]$ and $\lambda > 0$ are additional parameters (the latter should not be confused with the one in Equation 16). In other words, the new proposal becomes a mixture between a modified prior and a point mass at 1. Using this in the proposal for ξ, we get an acceptance probability equal to $\min\{1, a_{\xi\to\xi^*}\}$, with:

$$a_{\xi\to\xi^*} \triangleq \frac{p(\xi^*|Y, -)J(\xi|\xi^*)}{p(\xi|Y, -)J(\xi^*|\xi)} = \frac{p(Y|\xi^*, -)p(\xi^*|-)J(\xi|\xi^*)}{p(Y|\xi, -)p(\xi|-)J(\xi^*|\xi)} \tag{21}$$

Here the authors make a questionable remark, which is that, in the MH sense, the "current state" is $\xi = \emptyset$, which leads them to conclude that $p(\xi|-) = J(\xi|\xi^*) = 1$. The problem is that this contradicts one of the consequences of Kolmogorov's axioms; namely, that $\mathbb{P}(\emptyset) = 0$ in any probability space. In fact, dividing by the expression $p(\xi|Y, -)$ in the acceptance ratio wouldn't make sense in the first place.

One way to fix this issue is to make the subtle correction that the proposal $\xi = (\kappa_d, \lambda', g^T)$ if $\kappa_d > 0$, and $\xi = (\kappa_d = 0)$ otherwise. This would be compatible with the intuition that the current state is actually $\kappa_d = 0$, meaning that we don't add columns and nothing changes if the proposal gets rejected. Using this convention in 21 leads to:

$$\begin{aligned} a_{\xi\to\xi^*} &= \frac{p(Y|\xi^*, -)p(\xi^*|-)J(\xi|\xi^*)}{p(Y|\xi, -)p(\xi|-)J(\xi^*|\xi)} \\ &= \frac{p(Y|\xi^*, -)p(\xi^*|-)J(\kappa_d = 0)}{p(Y|-)\mathbb{P}(\kappa_d = 0|\alpha)J(\xi^*|\xi)} \\ &= \frac{p(Y|\xi^*, -)p(\kappa_d|\alpha)p(\lambda'|d)p(g^T|\kappa_d, \lambda')J(\kappa_d = 0)}{p(Y|-)\mathbb{P}(\kappa_d = 0|\alpha)J(\kappa_d)p(\lambda'|d)p(g^T|\kappa_d, \lambda')} \\ &= \underbrace{\frac{p(Y|\xi^*, -)}{p(Y|-)}}_{a_l} \underbrace{\frac{p(\kappa_d|\alpha)}{J(\kappa_d)}}_{a_p} \underbrace{\frac{J(\kappa_d = 0)}{\mathbb{P}(\kappa_d = 0|\alpha)}}_{a_c} \end{aligned} \tag{22}$$

Note that, since we chose to use the prior as the proposal for λ', it is not present in $a_{\xi\to\xi^*}$. However, compare the above with Eq. 14 in [8]:

$$a_{\xi\to\xi^*} = \underbrace{\frac{p(Y|\xi^*,-)}{p(Y|-)}}_{a_l}\underbrace{\frac{p(\kappa_d|\alpha)}{J(\kappa_d)}}_{a_p}$$

This equation is problematic, since in the case when the proposal ξ^* is actually $\kappa_d = 0$ (i.e, not modifying anything), we get $a_{\xi\to\xi} \neq 1$, which contradicts a direct consequence of the standard MH step (to see this, simply replace ξ^* by ξ in Equation 21). This occurs because, although the likelihoods cancel, we have:

$$a_p(\kappa_d = 0) = \frac{\mathbb{P}(\kappa_d = 0|\alpha)}{J(\kappa_d = 0)} = \frac{e^{\gamma(\lambda-1)}}{1-\pi}$$

which can be either lower or greater than 1, depending on (λ, π). However, this situation is corrected in our derivation in 22, because:

$$a_{\xi\to\xi} = \frac{p(Y|-)}{p(Y|-)}\frac{\mathbb{P}(\kappa_d = 0|\alpha)}{J(\kappa_d = 0)}\frac{J(\kappa_d = 0)}{\mathbb{P}(\kappa_d = 0|\alpha)} = 1$$

This result holds for any proposal $J(\kappa_d)$ such that $J(\kappa_d = 0) > 0$. Sadly, there is no hint we can use in [17] to check if our derivation is correct, since they use the prior of κ_d for the proposal, so that everything cancels out. However, they do not mention that $\xi = \emptyset$. In any case, we believe that there is substantial evidence to show that Equation 22 is correct, so that we will use this in our implementation.

The only element that remains to be computed in Equation 22 is the ratio of the likelihoods. First, note that:

$$\frac{p(Y|\xi^*,-)}{p(Y|-)} = \prod_{d'=1}^{D}\frac{p(Y_{d':}|\xi^*,-)}{p(Y_{d':}|-)} = \frac{p(Y_{d:}|\xi^*,-)}{p(Y_{d:}|-)}\prod_{d'\neq d}\underbrace{\frac{p(Y_{d':}|\xi^*,-)}{p(Y_{d':}|-)}}_{1} = \frac{p(Y_{d:}|\xi^*,-)}{p(Y_{d:}|-)}$$

The last line uses the fact that ξ^* is a proposal that only concerns the object d. Next, note that:

$$p(Y_{d:}|\xi^*,-) = \prod_{n=1}^{N} p(Y_{dn}|\xi^*,-) = \prod_{n=1}^{N}\int_{x'_n} p(Y_{dn}|x'_n,\xi^*,-)p(x'_n)dx'_n$$

where x'_n is the n-th column of X', and therefore a vector of length κ_d. We will derive the integral inside the product for an arbitrary n. First, recall from the definition of the model in Equation 12 that:

$$Y_{dn}|G, x_n, \psi_d \sim \mathcal{N}((Gx_n)_d, \psi_d) \qquad\qquad x'_n \sim \mathcal{N}(0_{\kappa_d}, I_{\kappa_d})$$

Using $G' = (G^T, g'^T)^T$ and $x_n^* = (x_n, x_n')$, we get that:

$$y_{dn} - (G'x_n^*)_d = y_{dn} - ((G^T, g'^T)^T (x_n, x_n'))_d = y_{dn} - (Gx_n)_d - (g'^T)^T x_n' = \hat{E}_{dn} - (g'^T)^T x_n'$$

where $\hat{E} = Y - GX$ (not to be confused with the one used in Section 3.1.1). Therefore:

$$\begin{aligned}
p(Y_{dn}|x_n', \xi^*, -)p(x_n') &= (2\pi\psi_d)^{-1/2} \exp\left(-\frac{1}{2\psi_d}\left[\hat{E}_{dn} - (g'^T)^T x_n'\right]^2\right)(2\pi)^{-\kappa_d/2} \exp\left(-\frac{1}{2}{x'}_n^T x_n'\right) \\
&= (2\pi\psi_d)^{-1/2}(2\pi)^{-\kappa_d/2} \exp\left(-\frac{\hat{E}_{dn}^2}{2\psi_d} - \frac{1}{2\psi_d}\underbrace{((g'^T)^T x_n')^2}_{{x'}_n^T g g'^T x_n'} + \frac{\hat{E}_{dn}(g'^T)^T x_n'}{\psi_d} - \frac{1}{2}{x'}_n^T x_n'\right) \\
&= (2\pi\psi_d)^{-1/2}(2\pi)^{-\kappa_d/2} \exp\left(-\frac{1}{2}{x'}_n^T \underbrace{\left[\frac{1}{\psi_d} g' g'^T + I_{\kappa_d}\right]}_{M} x_n' - \frac{\hat{E}_{dn}^2}{2\psi_d} + \frac{\hat{E}_{dn}(g'^T)^T x_n'}{\psi_d}\right) \\
&= (2\pi\psi_d)^{-1/2}(2\pi)^{-\kappa_d/2} \exp\left(-\frac{1}{2}{x'}_n^T M x_n' - \frac{\hat{E}_{dn}^2}{2\psi_d} + \underbrace{\frac{\hat{E}_{dn}}{\psi_d}(g'^T)^T M^{-1}}_{m_n^T} M x_n'\right) \\
&= (2\pi\psi_d)^{-1/2}(2\pi)^{-\kappa_d/2} \exp\left(-\frac{\hat{E}_{dn}^2}{2\psi_d} + \frac{1}{2} m_n^T M m_n\right) \exp\left(-\frac{1}{2}({x'}_n^T - m_n)^T M ({x'}_n^T - m_n)\right)
\end{aligned}$$

where:

$$M \triangleq \frac{1}{\psi_d} g' g'^T + I_{\kappa_d} \qquad\qquad m_n \triangleq \frac{\hat{E}_{dn}}{\psi_d} M^{-1} g'^T \tag{23}$$

Also, M is invertible because of Lemma 1 (see Appendix A), since $\frac{1}{\psi_d} g' g'^T$ is a real, symmetric matrix. Using the results from above, we have:

$$\begin{aligned}
&\int_{x_n'} p(Y_{dn}|x_n', \xi^*, -)p(x_n') dx_n' = \\
&\quad (2\pi\psi_d)^{-1/2}(2\pi)^{-\kappa_d/2} \exp\left(-\frac{\hat{E}_{dn}^2}{2\psi_d} + \frac{1}{2} m_n^T M m_n\right) \int_{x_n'} \exp\left(-\frac{1}{2}({x'}_n^T - m_n)^T M ({x'}_n^T - m_n)\right) dx_n' \\
&\quad = (2\pi\psi_d)^{-1/2}(2\pi)^{-\kappa_d/2} \exp\left(-\frac{\hat{E}_{dn}^2}{2\psi_d} + \frac{1}{2} m_n^T M m_n\right)(2\pi)^{\kappa_d/2} |M^{-1}|^{1/2} \\
&\quad = (2\pi\psi_d)^{-1/2} |M|^{-1/2} \exp\left(-\frac{\hat{E}_{dn}^2}{2\psi_d} + \frac{1}{2} m_n^T M m_n\right)
\end{aligned}$$

And from here we obtain:

$$\begin{aligned}p(Y_{d:}|\xi^*,-) &= \prod_{n=1}^{N}(2\pi\psi_d)^{-1/2}|M|^{-1/2}\exp\left(-\frac{\hat{E}_{dn}^2}{2\psi_d}+\frac{1}{2}m_n^T M m_n\right)\\ &= (2\pi\psi_d)^{-N/2}|M|^{-N/2}\exp\left(-\frac{1}{2\psi_d}\hat{E}_{d:}^T\hat{E}_{d:}+\frac{1}{2}\sum_{n=1}^{N}m_n^T M m_n\right)\end{aligned}$$

The expression for $p(Y_{d:}|-)$ can be directly obtained from the model definition in Equation 12:

$$\begin{aligned}p(Y_{d:}|-) &= \prod_{n=1}^{N}p(Y_{dn}|-)\\ &= \prod_{n=1}^{N}(2\pi\psi_d)^{-1/2}\exp\left(\frac{(y_{dn}-(Gx_n)_{dn})^2}{2\psi_d}\right)\\ &= (2\pi\psi_d)^{-N/2}\exp\left(\sum_{n=1}^{N}\frac{\hat{E}_{dn}^2}{2\psi_d}\right)\\ &= (2\pi\psi_d)^{-N/2}\exp\left(-\frac{1}{2\psi_d}\hat{E}_{d:}^T\hat{E}_{d:}\right)\end{aligned}$$

which means that the likelihood ratio becomes:

$$a_l = |M|^{-N/2}\exp\left(\frac{1}{2}\sum_{n=1}^{N}m_n^T M m_n\right) \tag{24}$$

This equation differs from Equation 20 in [8] by the factor $(2\pi)^{N\kappa_d/2}$, which comes their expression for $p(Y_{d:}|\xi^*,-)$. This could be simply a typo, or an error if the authors didn't consider the term $(2\pi)^{-\kappa_d/2}$ in the prior for x_n', which cancels with the one arising from the integral of the Gaussian kernel. Again, we can't be sure that theirs is a typo or not, but all the steps we have shown here seem to be accurate derivations. Hence, we will stick with our version for the implementation. Hence, the acceptance ratio becomes:

$$\begin{aligned}a_{\xi\to\xi^*} &= a_l a_p a_c\\ &= |M|^{-N/2}\exp\left(\frac{1}{2}\sum_{n=1}^{N}m_n^T M m_n\right)\frac{\text{Poisson}(\kappa_d;\gamma)}{(1-\pi)\text{Poisson}(\kappa_d;\lambda\gamma)+\pi\mathbb{I}(\kappa_d=1)}(1-\pi)e^{-\gamma(\lambda-1)}\end{aligned} \tag{25}$$

Assuming that the proposal is accepted, we need to additionally sample the matrix X'.

$$\begin{aligned}
p(X'|\xi^*, Y, -) &= \frac{p(Y|X', \xi^*, -)p(X')}{p(Y|\xi^*, -)} \\
&= \frac{p(Y_d|X', \xi^*, -)p(X')}{p(Y_d|\xi^*, -)} \\
&= \frac{\prod_{n=1}^N (2\pi\psi_d)^{-1/2}(2\pi)^{-\kappa_d/2} \exp\left(-\frac{\hat{E}_{dn}^2}{2\psi_d} + \frac{1}{2} m_n^T M m_n\right) \exp\left(-\frac{1}{2}({x'}_n^T - m_n)^T M({x'}_n^T - m_n)\right)}{(2\pi\psi_d)^{-N/2}|M|^{-N/2} \exp\left(-\frac{1}{2\psi_d}\hat{E}_{d:}^T\hat{E}_{d:} + \frac{1}{2}\sum_{n=1}^N m_n^T M m_n\right)} \\
&= \frac{(2\pi\psi_d)^{-N/2} \exp\left(-\frac{1}{2\psi_d}\hat{E}_{d:}^T\hat{E}_{d:} + \frac{1}{2}\sum_{n=1}^N m_n^T M m_n\right) \prod_{n=1}^N (2\pi)^{-\kappa_d/2} \exp\left(-\frac{1}{2}({x'}_n^T - m_n)^T M(x'\right.}{(2\pi\psi_d)^{-N/2}|M|^{-N/2} \exp\left(-\frac{1}{2\psi_d}\hat{E}_{d:}^T\hat{E}_{d:} + \frac{1}{2}\sum_{n=1}^N m_n^T M m_n\right)} \\
&= \prod_{n=1}^N (2\pi)^{-\kappa_d/2}|M|^{1/2} \exp\left(-\frac{1}{2}({x'}_n^T - m_n)^T M({x'}_n^T - m_n)\right)
\end{aligned}$$

Hence:

$$x'_n|\xi^*, Y, - \overset{ind}{\sim} \mathcal{N}(m_n, M^{-1}) \tag{26}$$

It is important to note that this step is entirely missing from the original paper. Moreover, the fact that we recovered exactly the functional form for $\mathcal{N}(m_n, M^{-1})$, suggests that our expression for $p(Y_{d:}|\xi^*, -)$ is the correct one, and therefore that our acceptance ratio (i.e, without the term $(2\pi)^{N\kappa_d/2}$) is correct.

3.1.3 Update for the parameter of the IBP

Since we have put a prior on α, its parameters can be updated using:

$$\begin{aligned}
p(\alpha|Z) &\propto p(Z|\alpha)p(\alpha) \\
&\propto p([Z]|\alpha)p(\alpha) \\
&\propto \alpha^{e-1}e^{-f\alpha}\frac{\alpha^{K_+}}{\prod_{h=1}^{2^D-1} K_h!}e^{-\alpha H_D}\prod_{k=1}^{K_+}\frac{(D-m_k)!(m_k-1)!}{D!} \\
&\propto \alpha^{e+K_+-1}e^{-\alpha(f+H_D)}
\end{aligned}$$

Hence:

$$\alpha|Z \sim \text{Gamma}(e + K_+, f + H_D) \tag{27}$$

3.1.4 Updates for the latent variables

The update for the n-th component of X is obtained from:

$$\begin{aligned}
p(x_n|Y,-) &\propto p(y_n|x_n,-)p(x_n) \\
&\propto \exp\left(-\frac{1}{2}\left[(y_n - Gx_n)^T \Psi^{-1}(y_n - Gx_n) + x_n^T x_n\right]\right) \\
&\propto \exp\left(-\frac{1}{2}\left[-2y_n^T \Psi^{-1} G x_n + x_n^T G^T \Psi^{-1} G x_n + x_n^T x_n\right]\right) \\
&= \exp\left(-\frac{1}{2}\left[-2y_n^T \Psi^{-1} G x_n + x_n^T \underbrace{\left\{G^T \Psi^{-1} G + I_{K_+}\right\}}_{\Lambda} x_n\right]\right) \\
&= \exp\left(-\frac{1}{2}\left[x_n^T \Lambda x_n - 2\underbrace{y_n^T \Psi^{-1} G \Lambda^{-1}}_{\mu_n^T} \Lambda x_n\right]\right)
\end{aligned}$$

Hence:

$$x_n|Y,- \sim \mathcal{N}(\mu_n, \Lambda^{-1}) \tag{28}$$

with:

$$\Lambda \triangleq G^T \Psi^{-1} G + I_{K_+} \qquad\qquad \mu_n \triangleq \Lambda^{-1} G^T \Psi^{-1} y_n \tag{29}$$

By Lemma 1 (see Appendix A), we know that Λ is non-singular because $G^T \Psi^{-1} G$ is a symmetric real matrix.

3.1.5 Update for the factors' precisions

From the definition of the model in Equation 12, we know that the Markov blanket of λ_k includes $(G_{:k}, Z_{:k})$. Hence:

$$\begin{aligned}
p(\lambda_k|G_{:k}, Z_{:k}) &\propto p(G_{:k}|\lambda_k, Z_{:k})p(\lambda_k|d) \\
&\propto \left[\prod_{d:z_{dk}=1} (2\pi\lambda_k^{-1})^{-1/2} e^{-\frac{1}{2}\lambda_k g_{dk}^2}\right] \lambda_k^{c-1} e^{-\lambda_k d} \\
&\propto \lambda_k^{m_k/2+c-1} e^{-\lambda_k (d+(1/2)\sum_{d=1}^{D} g_{dk}^2)}
\end{aligned}$$

Thus:

$$\lambda_k|G_{:k}, Z_{:k} \sim \text{Gamma}\left(c + \frac{1}{2}m_k, d + \frac{1}{2}\sum_{d=1}^{D} g_{dk}^2\right) \tag{30}$$

The update for the common parameter d is obtained from:

$$\begin{aligned}
p(d|\{\lambda_k\} &\propto p(d)\prod_{k=1}^{K_+} p(\lambda_k|d) \\
&\propto d^{c_0-1}e^{-dd_0}\prod_{k=1}^{K_+} d^c e^{-\lambda_k d} \\
&= d^{c_0+cK_+-1}e^{-d(d_0+\sum_{k=1}^{K_+}\lambda_k)}
\end{aligned}$$

Therefore:

$$d|\{\lambda_k\} \sim \text{Gamma}\left(c_0 + cK_+, d_0 + \sum_{k=1}^{K_+}\lambda_k\right) \tag{31}$$

3.1.6 Update for the noise variances

From the model definition in Equation 12, we have that:

$$\begin{aligned}
p(\Psi|Y,-) &\propto p(Y|\Psi,-)p(\Psi) \\
&= p(\Psi)\prod_{n=1}^{N}(2\pi)^{-D/2}|\Psi|^{-1/2}\exp\left(-\frac{1}{2}\hat{E}_{:n}^T\Psi^{-1}\hat{E}_{:n}\right) \\
&\propto p(\Psi)\prod_{n=1}^{N}\prod_{d=1}^{D}\psi_d^{-1/2}e^{-\frac{\hat{E}_{dn}^2}{2\psi_d}} \\
&\propto \prod_{d=1}^{D}\psi^{-a-1}e^{-b/\psi_d}\psi_d^{-N/2}e^{-\frac{1}{2\psi_d}\sum_{n=1}^{N}\hat{E}_{dn}^2} \\
&\propto \prod_{d=1}^{D}\psi^{-a-N/2-1}e^{-\frac{1}{\psi_d}\left(b+\frac{1}{2}\sum_{n=1}^{N}\hat{E}_{dn}^2\right)}
\end{aligned}$$

Thus:

$$\psi_d|Y,- \sim \text{Inv-Gamma}\left(a+\frac{N}{2}, b+\frac{1}{2}\sum_{n=1}^{N}\hat{E}_{dn}^2\right) \tag{32}$$

Finally, for the common parameter b:

$$\begin{aligned}
p(b|\Psi) &\propto p(b)p(\Psi|b) \\
&\propto b^{a_0-1}e^{-bb_0}\prod_{d=1}^{D}b^a e^{-b/\psi_d} \\
&= b^{a_0+aD-1}e^{-b\left(b_0+\sum_{d=1}^{D}1/\psi_d\right)}
\end{aligned}$$

Hence:

$$b|\Psi \sim \text{Gamma}\left(a_0 + aD, b_0 + \sum_{d=1}^{D} \frac{1}{\psi_d}\right) \tag{33}$$

3.2 Summary of the sampling procedure

Algorithm 1: NSFA Gibbs sampler

```
input : Data $Y$, hyper-parameters $\mathcal{H} = \{c, e, f, a, a_0, b_0, c_0, d_0, \lambda_{MH}, \pi_{MH}\}$, #
        iterations $S$
$\mathcal{C} \triangleq \{\alpha, Z, d, \lambda, G, X, b, \Psi\} \leftarrow$ init_chain($\mathcal{H}$)
$\mathbb{C} \leftarrow \mathcal{C}$
for $s \leftarrow 1$ to $S$ do
    $K \leftarrow$ ncol($Z$)
    for $d \leftarrow 1$ to $D$ do
        for $k \leftarrow 1$ to $K$ do
            $Z_{dk}, G_{dk} \leftarrow$ update_z_g($Y, \mathcal{C}, \mathcal{H}$)
        $\kappa_d \leftarrow$ kappa_proposal($\mathcal{C}, \mathcal{H}$)
        if $\kappa_d > 0$ then
            $g', \lambda' \leftarrow$ g_lambda_proposal($\mathcal{C}, \mathcal{H}$)
            $a_{\xi \to \xi^*} \leftarrow$compute_acc_ratio($Y, \mathcal{C}, \mathcal{H}$)
            if $U(0,1) < \min\{1, a_{\xi \to \xi^*}\}$ then
                $Z' \leftarrow 0_{D \times \kappa_d}, Z'_{d:} \leftarrow 1^T_{\kappa_d}, Z \leftarrow (Z, Z')$
                $G \leftarrow (G, g'), \lambda \leftarrow (\lambda, \lambda')$
                $X' \leftarrow$ get_X_new($Y, \mathcal{C}, \mathcal{H}$)
                $X \leftarrow (X^T, X'^T)^T$
    for $n \leftarrow 1$ to $N$ do
        $X_{:n} \leftarrow$ update_X_n($Y, \mathcal{C}, \mathcal{H}$)
    $\{\alpha, \lambda, G, \Psi, b, d\} \leftarrow$ update_rest($Y, \mathcal{C}, \mathcal{H}$)
    $\mathcal{C} \leftarrow \{\alpha, Z, d, \lambda, G, X, b, \Psi\}$
    $\mathbb{C} \leftarrow (\mathbb{C}, \mathcal{C})$
output: $\mathbb{C}$
```

As a means to synthesize the construction of the Gibbs sampler given in the present section, and putting particularly more detail in the stage where new features are added, we describe the Gibbs simple in pseudo-code in Algorithm 1. Note that we maintained the schedule suggested by the authors.

3.3 Out-of-sample evaluation

We would like to evaluate the performance using the log-likelihood computed not only in the training set Y, but also on a held-out test set Y_{new} of size $D_{\text{new}} \times N$. To do this, suppose we have run the chain in a training set, and for simplicity assume we keep only one point. Thus, we have one sample from (hopefully) the posterior distribution of $\{\alpha, Z, d, \lambda, G, X, b, \Psi\}$. We can use the parameters that do not depend on the objects, i.e $\{\alpha, d, \lambda, X, b\}$, to sample the ones that do. We will call these new parameters $\{Z_{\text{new}}, G_{\text{new}}, \Psi_{\text{new}}\}$.

To get Z_{new}, recall that, given the known parameters, we actually know the number of hidden features K, which is equal to the number of rows in X. Thus, we can sample Z using the finite version of the IBP. To do this, we first need to draw the vector π_{new} from its posterior distribution $p(\pi_{\text{new}}|Z)$ using Equation 3. Then, we simply sample Z_{new} from the predictive posterior:

$$p(Z_{\text{new}}|Z) = \int_{\pi_{\text{new}}} p(Z_{\text{new}}|\pi_{\text{new}})p(\pi_{\text{new}}|Z)d\pi$$

In practice, this amounts to drawing Z_{new} using in the second line of Equation 2. Having sampled, Z_{new}, we can now generate G_{new} using Z_{new} and λ in the prior for G. Finally, we can get Ψ_{new} also from its prior by using the given value of b.

Using the new set of parameters $\{\alpha, d, \lambda, X, b, Z_{\text{new}}, G_{\text{new}}, \Psi_{\text{new}}\}$ we can proceed to compute any test quantity. In particular, we can compute the log-likelihood for Y_{new}. In the case when we have more than one sample from the posterior, we simply repeat the above process for each of them, and then take the average of the log-likelihoods computed.

4 Simulation

We implemented Algorithm 1 in R. As a way to check its validity, we applied it to a toy dataset, sampled from a simpler version of the NSFA model: instead of putting priors on $(\alpha, \{\lambda_k\}, \{\psi_d\})$, we set them to fixed values. We will later refer to this model as the "Reduced NSFA". Specifically, we set $\alpha = 5$, $\lambda_k = 0.1, \forall k$, and $\psi_d = 0.2, \forall d$. The choices of λ_k and ψ_d result in a situation of high signal-to-noise ratio (SNR), which should make it easy for the model to find the latent structure. We chose $D = 100$ and $N = 150$. With the choices of D and α, we have $\mathbb{E}[K|\alpha] = \alpha H_D \approx 26.9$ (see Appendix B for a derivation of this formula). Finally, we split the rows of the Y matrix generated, by using 2/3 of it for training and 1/3 for testing.

The hyperparameters we used were copied from the Github repository that authors created for their implementation[2]. They used a value of 1 for all of the hyperparameters of the Gamma priors. For the MH proposal, they showed a value of 10 for λ, although we couldn't find there a default value for π. Thus, we resorted to using $\pi = 0.1$, which was mentioned in the paper as yielding good results.

[2]`https://github.com/davidaknowles/nsfa/code/defaultsettings.m`

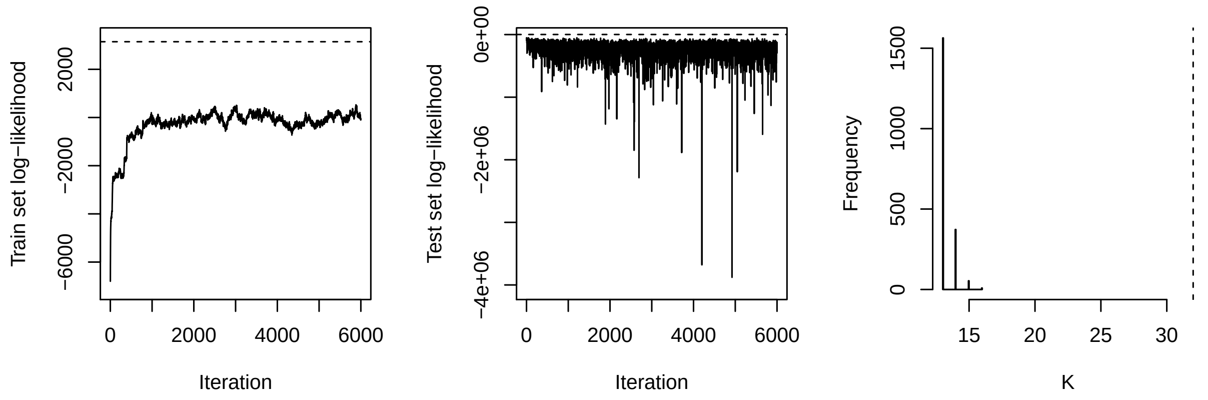


Figure 1: Results of the simulation using a toy dataset after 6,000 iterations of the Gibbs sampler. The first panel shows the trace of the log-likelihood in the training set; the second shows the same for the test set, while the final panel shows the histogram for K computed by considering just the last third of the chain. The dashed lines in the first two panels show the log-likelihood evaluated with the actual parameters used to build the data, while the line in the last panel shows the actual number of columns of the Z matrix used to simulate Y.

We run the Gibbs sampler for the NSFA for 6,000 iterations. Figure 1 shows the results of this process. The first panel shows the trace of the log-likelihood in the training set, the second shows the same for the test set, and the final panel shows the histogram for K computed by considering just the last third of the chain. The dashed lines in the first two panels show the log-likelihood evaluated with the actual parameters used to build the data. Similarly, the dashed line in the last panel shows the actual number of columns of the Z matrix used to simulate the data.

The first thing we notice is that the training set log-likelihood quickly increases and then stabilizes after less than 1,000 iterations, which is a good sign of a correct algorithmic implementation. However, the trace never reaches the dashed line. This is probably related to the fact that the number of features was underestimated by a factor of 1/3, with virtually no mass put in the region of the true $K = 32$. By themselves, these facts are not concerning, as the model might be trading-off likelihood by imposing regularization so as to obtain a more parsimonious model (we are not plotting the log-posterior). Nevertheless, we do notice that the performance in the held-out test set does not improve as iterations go by. This is something to be concerned, and will be investigated also in the Section 5.

It is worth noting that we also tried to run the simulation using the authors' expression for the MH acceptance ratio. The presence of the $(2\pi)^{N\kappa_d/2}$ term resulted in all of the proposals being accepted. Note that for $N = 150$ and $\kappa_d = 2$, one obtains $(2\pi)^{N\kappa_d/2} \approx 10^{120}$. Therefore, the model quickly exploded in terms of the number of features considered, making inference intractable. We think this is the final piece of evidence needed to accept our derivation as the correct one, so that we do not consider their version in what follows.

5 Application to a real dataset

5.1 Description of the MNIST dataset

To test the performance of our implementation of NSFA, we applied it to the MNIST handwritten digit dataset [19] (we used the CSV version provided by [20]). The dataset is composed of a training set of size 60,000 and a test set of size 10,000, each comprised of images of size 28×28. Additionally, the datasets containg a label associating each image with a digit between 0 and 9. We chose MNIST because of the fact that these observations are images which only represent 10 digits. Thus, it is reasonable to assume that the information contained in them can be summarized into fewer features than the $28 \times 28 = 784$ pixels that compose them.

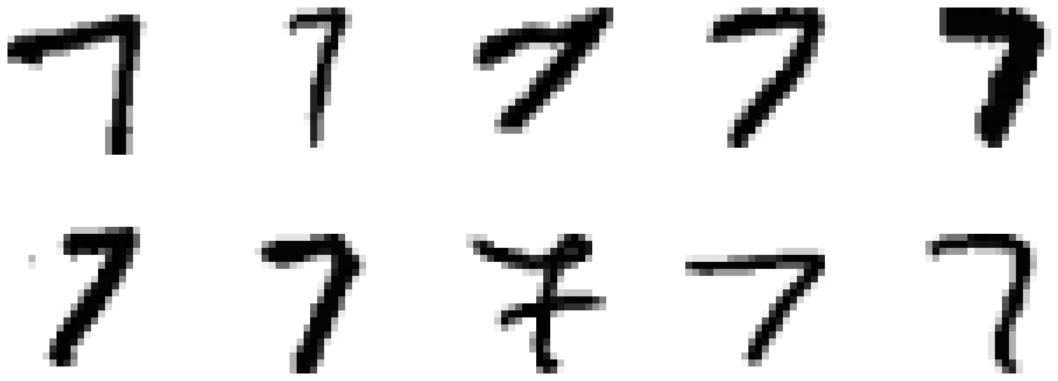


Figure 2: Ten samples of the digit "7" in the MNIST training dataset.

However, we certainly do not expect to find a posterior distribution concentrated around $K = 10$ latent features, because of the amount of variability that exists within each digit. Indeed, in Figure 2 we can observe just how different are eight observations of the digit "7". Some people wrote the number with a stroke on the middle, while others didn't. The width of the lines is also very different between samples, but similar within each image. On the other hand, there is a shared structure among them: the horizontal line above (of variable length), and the diagonal stroke (of somewhat fixed length but varying angle). Therefore, we intuitively think that the NSFA could pick up these more subtle shared characteristics as latent features.

Now, because of the size of the dataset and the limited computational resources at hand, we needed to subsample the data so that we could obtain results in reasonable time. To this end, we obtained $D = 200$ samples both from the training and test set. We did an stratified sampling so that the distribution of the digits was preserved.

5.2 Results

We applied the NSFA model to the data described above. The hyperparameters we used were the same that we utilized in Section 4. Additionally, we fit the reduced NSFA, which was the model used in that same section to simulate the data. We used $\alpha = 1$, as this was also tested in the paper, and $b = d = 2$. Since we called this the "reduced NSFA", we named the complete version "full NSFA". Moreover, the reduced version was fit with two settings for the MH parameters: $(\lambda = 1, \pi = 0)$ (i.e., using the prior), and $(\lambda = 10, \pi = 0.1)$ (i.e., the same settings used for the full NSFA).

We run the three models for 6,000 iterations. The full model took 7 hours to complete, while the reduced models took about 5 hours each. The results of the iterations are summarized in Figure 6 (Appendix C). The first thing we notice is that the full NSFA is the only one that can be assumed to have barely reached convergence, at least for the last 100 iterations or so, as the traces for the training set log-likelihood show. The reduced versions are nowhere near stationarity, steadily increasing even after 6,000 iterations.

Second, we notice that the full NSFA model achieves the best out of sample evaluation of the three (note the scale of the plots), even though the average performance decreases monotonically as iterations go by. Sadly, the authors did not show trace plots of the log-likelihood in test sets, just averages for the 100 samples in the chain. Therefore, we cannot know if this phenomenon was present in the authors' experiments. Nevertheless, there are clear symptoms of poor generalization, which was the same conclusion obtained in Section 4. We will discuss ways to address this in Section 6.

Lastly, the large amount of features included in the final stages of the three models are striking, with the full NSFA showing the highest value. Moreover, the traces for K show even less signs of convergence to a stationary distribution.

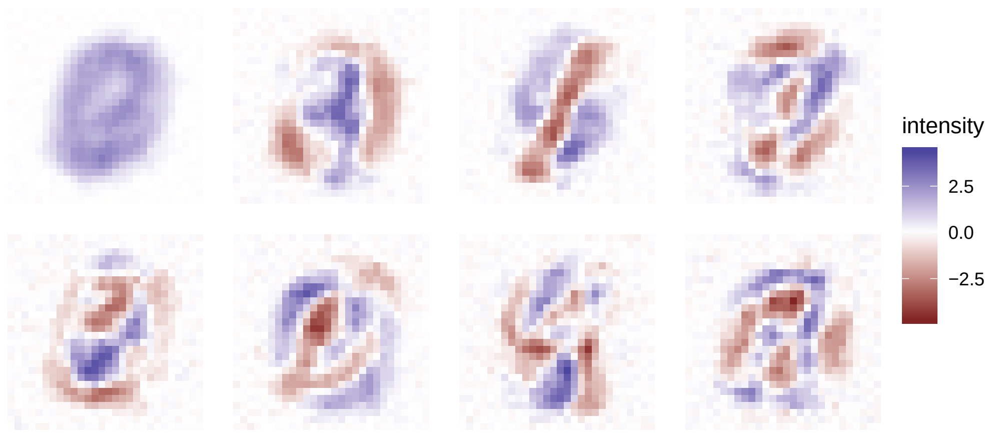


Figure 3: Eight most relevant features in X for the last sample of the chain.

Since the full NSFA model was the one that achieved the best (or least worse) out-of-sample performance, we wanted to investigate deeper the characteristics of its posterior distribution. To this end, we show in Figure 3 a visual inspection of the eight most relevant features selected by the model. We define relevance as having the most absolute total weights assigned to it across all rows of G. The features are ordered from left to right and top to bottom in terms of their relevance. It is interesting that the most relevant feature is a fairly homogeneous shape at the center of the image, where most of the numbers lie, acting like the intercept of a linear regression. The rest of the features add or substract from this ground to create complex shapes. However, aside from the second (which looks like a 3 or the right half of an 8) and third (which looks like the diagonal stroke of a seven), they cannot be easily interpreted.

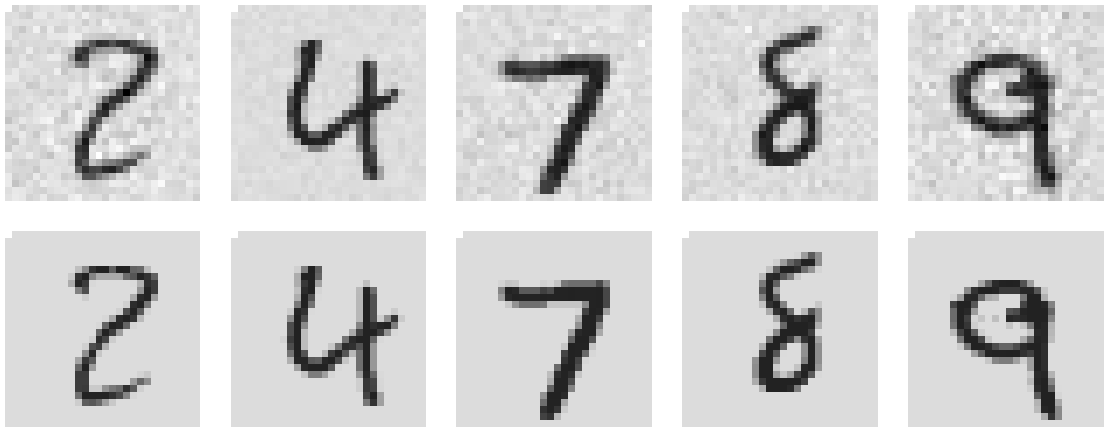


Figure 4: Fitted (above) and actual (below) image for one sample of each of the five most difficult to represent digits.

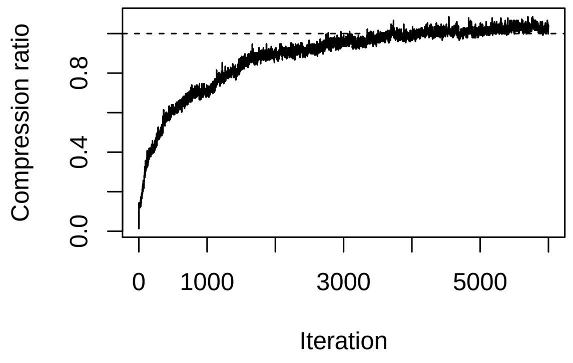
We also took a look into the fitted values $\hat{Y} = GX$ generated by the last sample of the chain, by plotting one randomly chosen image for each of the 5 most difficult to represent digits, shown in Figure 4. By difficulty we mean the numbers that on average have more non-zero weights associated to them in G, which implies that many latent factors are required to reconstruct them. The fact that the fitted values almost perfectly replicate the actual images, even the parts that can be considered "errors" or unnecessary to represent a certain digit (the digits 8 and 9 are notable cases), is a clear sign that the model is overfitting to the training set. This of course is consistent with the poor out-of-sample performance shown in Figure 6.

5.3 Compression capabilites of the NSFA model

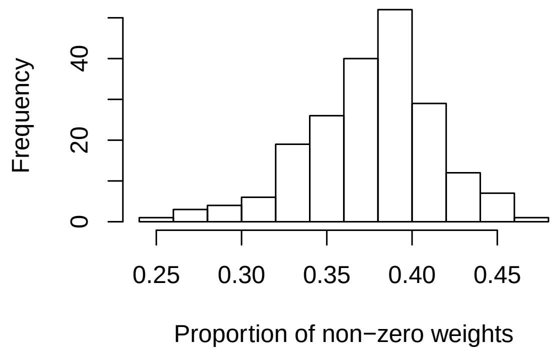
We can define a compression ratio as the amount of numbers required to represent the dataset in the latent space, versus in their original raw form. In other words, this is the ratio of elements in G plus the ones in X, to the elements in Y:

$$R \triangleq \frac{DK + KN}{DN} = \frac{K}{N}\left(1 + \frac{N}{D}\right) \tag{34}$$

Figure 5a shows the trace of R for the full NSFA model. Again, as the ratio is just a scaled version of K, we observe the same pattern of slow convergence. But, more importantly, the ratio



(a) Trace of the compression ratio R.



(b) Histogram of non-zero weights in rows of G.

Figure 5: Trace of the compression ratio and histogram for the proportion of non-zero weights in each row of G (last sample from the chain).

eventually reaches 1 and stays there. Assuming the chain achieved stationarity at the end, then this means that the full NSFA model does not deliver any actual compression, in spite of the fact that K/N – the amount of features used to the total number of pixels – is about 20%. Again, this issue appears consistent with the idea that the model is overfitting.

Now, Equation 34 shows that, if one increased the number of objects in the training set D, and the model selected the same amount of features K, then the compression ratio should improve. Therefore, we experimented with doubling the size of the sub-sample of digits, so that now $D = 400$. This resulted in the model including even more features – so much more that iterations slowed considerably, making it impossible to let the program finish. The simulation was stopped at iteration 1,590 when $K = 256$, which yields $R \approx 0.97$, very close to 1. Hence, the issue appears to be inherently related to the performance of the NSFA model on the MNIST dataset.

On the other hand, the definition of R does not take into account the sparseness of G: it counts every entry in the matrix, independent of it being 0 or not. Since the NSFA is meant to promote sparse weight matrices, it is expected that would have a low count of non-zero entries. To investigate this, we constructed an histogram (Figure 5b) of ratios computed by counting the number of non-zero entries in each row of the last G in the chain, and then dividing that by N (thus, the histogram involves K different values). It is interesting to note that most of the rows have between 30% and 45% of their entries activated, and that none of them contains more than 50%. Therefore, if instead of the number of components in G we considered its memory size when stored as a sparse matrix, we would expect to see gains in this sense of compression. However, these would have to be then weighed against the overhead in terms of CPU time of having to perform matrix operations on sparse matrices, which are far more expensive. In fact, 30% of non-zero entries is too much for most standards when considering this trade-off.

6 Discussion

In this report we gave a thorough and independent derivation of the relevant results in [8]. In this process, we found some inconsistencies with the original paper, ranging from minor typos to more relevant discrepancies. The latter was especially true for the Gibbs updates derived in Section 3.1.2. We proved that our derivations were correct either by referencing third party sources, by checking that our results yielded consistent consequences, or by finding that the equations in the original paper led to contradictions or problematic conclusions. We think this process was in itself a contribution towards improving the work of the authors.

Once the Gibbs updates were derived, we implemented them in R. We performed a simulation analysis that showed no obvious error in our code. Then, we applied the method to a small subset of the MNIST dataset. The results yielded the following conclusions:

1. The method can fit the training set with very high accuracy.
2. However, it uses so many latent features that the total number of elements in G and X is similar to DN.
3. Also, the convergence is very slow in terms of number of iterations needed.
4. Moreover, the out-of-sample performance worsens as iterations go by.

One easy way to tackle the above problems is to extend the IBP to a two parameter formulation, suggested by its relation to the Beta Process. As [10] and [14] show, the addition of a second parameter (called the concentration parameter in the language of the Beta Process) can greatly affect the number of active columns promoted by the IBP prior. Moreover, as the case with α, this parameter could also be learned by putting a prior on it. This modification has been explored by the authors in subsequent work [16], although they assumed a fixed value and used a different parameterization, proposed in [11].

The inclusion of the 2nd parameter in the IBP might help with dealing with the number of active columns and the speed of convergence. However, it will probably not fix the issue with out of sample performance. This is because the reason behind this problem is rather a lack of a rich enough structure in the model, that can learn the underlying distribution of the data so as to be able to work as a meaningful generative model once trained. To use MNIST as a concrete example, in order to obtain a coherent digit from the predictive distribution, we need first Z_{new} to deliver a precise allocation of 0's and 1's that match one of the 10 digits, out of the 2^K possible allocations. Coordinated allocation is nearly impossible because the Bernoulli rv are independent given the beta priors, and these are the same for every row of Z_{new}.

Even, if we somehow succeeded in obtaining a coherent allocation, we now need to achieve the equally difficult task of obtaining coherent signs and magnitudes in the corresponding rows of G_{new}. The dependence of G_{new} on Z_{new} is very weak, so that it only communicates which positions are non-zero. Moreover, the components of each row of G_{new} are sampled independently of one another, and hence coordination to deliver precise weights and signs is very unlikely. Thus, we see that there are two problems with the NSFA that need to be addressed in order to improve out-of-sample performance:

1. Dependent sampling within rows of Z.
2. Increased dependence of $G_{d:}$ on $Z_{d:}$.

Adding dependence within rows of Z is difficult to achieve, because the Beta-Bernoulli process assumes conditional independence given the Beta process. Therefore, in order to fix this issue, we need to replace this component, and start over with a fixed K. Then, we could tackle both of the above issues by altering the NSFA as follows:

$$\begin{aligned} u_d &\overset{iid}{\sim} \text{Categorical}(1/L, ..., 1/L) \\ V_{lk} &\overset{iid}{\sim} N(0,1) \\ Z_{dk}|u_d, V &\sim \text{Bernoulli}(\sigma(V_{u_d k})) \\ G_{dk}|Z, u_d, V, \lambda_k &\overset{ind}{\sim} Z_{dk}\mathcal{N}(G_{dk}; V_{u_d k}, \lambda_k^{-1}) + (1 - Z_{dk})\delta_0(G_{dk}) \end{aligned}$$

where $\sigma(\cdot)$ is the logistic or sigmoid function. We can make sense of the matrix V of size $L \times K$ as encoding the "preference" of L latent classes for each of the K latent factors, while u_d would determine the association of the object d to one of those classes. Thus, the row vector $V_{u_d:}$ would be the preference of the d-th object for the different factors. This is why it makes sense to use it both in sampling $Z_{d:}$ and $G_{d:}$, which ensures coherence between and within them. In general, we would expect that $L \ll K$. In the concrete example of MNIST, we would like to obtain that the classes would actually correspond to the different ways of writing the 10 digits, so $L = 10w$, with w between 1 and 3 at most, so that $L \leq 30$. Note that, to build a Gibbs sampler for such a model, we would require the use of the Pólya–Gamma data augmentation strategy [21], in order to deal with the non-conjugacy induced by the logistic function.

In practice, if one wanted to implement this model for predictive uses, a good strategy would be to first run the NSFA in the training set to obtain K and also $(\{\lambda_k\}, X)$. With these values, we could run the modified NSFA mentioned above again in the training set, for a handful of values of L, initializing the chain with the known values for $(\{\lambda_k\}, X)$. Finally, one could choose the best L using some model selection criterion and then use that model to generate samples from the posterior predictive distribution.

References

[1] C. Spearman, “”general intelligence”, objectively determined and measured,” *The American Journal of Psychology*, vol. 15, no. 2, pp. 201–292, 1904.

[2] J. K. Ford, R. C. MacCallum, and M. Tait, “The application of exploratory factor analysis in applied psychology: A critical review and analysis,” *Personnel psychology*, vol. 39, no. 2, pp. 291–314, 1986.

[3] R. Bro, “Review on multiway analysis in chemistry—2000–2005,” *Critical reviews in analytical chemistry*, vol. 36, no. 3-4, pp. 279–293, 2006.

[4] G. Punj and D. W. Stewart, “Cluster analysis in marketing research: Review and suggestions for application,” *Journal of marketing research*, pp. 134–148, 1983.

[5] J. B. Schreiber, A. Nora, F. K. Stage, E. A. Barlow, and J. King, “Reporting structural equation modeling and confirmatory factor analysis results: A review,” *The Journal of educational research*, vol. 99, no. 6, pp. 323–338, 2006.

[6] H. F. Lopes and M. West, “Bayesian model assessment in factor analysis,” *Statistica Sinica*, pp. 41–67, 2004.

[7] D. Knowles and Z. Ghahramani, “Infinite sparse factor analysis and infinite independent components analysis,” in *International Conference on Independent Component Analysis and Signal Separation*. Springer, 2007, pp. 381–388.

[8] ——, “Nonparametric bayesian sparse factor models with application to gene expression modeling,” *The Annals of Applied Statistics*, pp. 1534–1552, 2011.

[9] T. L. Griffiths and Z. Ghahramani, “Infinite latent feature models and the indian buffet process,” in *Advances in neural information processing systems*, 2006, pp. 475–482.

[10] ——, “The indian buffet process: An introduction and review,” *Journal of Machine Learning Research*, vol. 12, no. Apr, pp. 1185–1224, 2011.

[11] J. Paisley and L. Carin, “Nonparametric factor analysis with beta process priors,” in *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 2009, pp. 777–784.

[12] P. Rai and H. Daumé, “The infinite hierarchical factor regression model,” in *Advances in Neural Information Processing Systems*, 2009, pp. 1321–1328.

[13] R. Durrett, *Probability: theory and examples*, 5th ed., 2018.

[14] R. Thibaux and M. I. Jordan, “Hierarchical beta processes and the indian buffet process,” in *Artificial Intelligence and Statistics*, 2007, pp. 564–571.

[15] T. J. Mitchell and J. J. Beauchamp, “Bayesian variable selection in linear regression,” *Journal of the American Statistical Association*, vol. 83, no. 404, pp. 1023–1032, 1988.

[16] A. Shah, D. Knowles, and Z. Ghahramani, “An empirical study of stochastic variational inference algorithms for the beta bernoulli process,” in *International Conference on Machine Learning*, 2015, pp. 1594–1603.

[17] E. Meeds, Z. Ghahramani, R. M. Neal, and S. T. Roweis, “Modeling dyadic data with binary latent factors,” in *Advances in neural information processing systems*, 2007, pp. 977–984.

[18] W. R. Gilks, *Markov chain Monte Carlo in practice*. Chapman & Hall: London, 1996.

[19] Y. LeCun, C. Cortes, and C. Burges, “The mnist dataset of handwritten digits (images),” 1999. [Online]. Available: http://yann.lecun.com/exdb/mnist

[20] J. Redmon. Mnist in csv. Accessed 2018-11-20. [Online]. Available: https://pjreddie.com/projects/mnist-in-csv/

[21] N. G. Polson, J. G. Scott, and J. Windle, “Bayesian inference for logistic models using pólya–gamma latent variables,” *Journal of the American statistical Association*, vol. 108, no. 504, pp. 1339–1349, 2013.

A Lemma 1

Lemma 1. *Let A be a symmetric matrix of size q with real entries, let I_q be the identity matrix of size q, and let $\nu > 0$. Then, the matrix $A + \nu I_q$ is positive definite, which implies that it is non-singular.*

Proof. Since A is symmetric and has real entries, it is positive semi-definite. Thus, its real eigenvalues $\{\epsilon_i\}_{i=1\dots q}$ are non-negative. Let $\{e_i\}_{i=1\dots q}$ be their associated eigenvectors. Then

$$\begin{aligned} &Ae_i = \epsilon_i e_i \\ \Leftrightarrow &Ae_i + \nu e_i = \epsilon_i e_i + \nu e_i \\ \Leftrightarrow &(A + \nu I_q)e_i = (\epsilon_i + \nu)e_i \end{aligned}$$

We see that the eigenvalues of $(A + \nu I_q)$ are $\{\epsilon_i + \nu\}_{i=1\dots q}$. Therefore, since $\nu > 0$, we have that these eigenvalues are all positive. Hence, the matrix is positive definite. □

B Additional properties of the IBP

Distribution of the number of non-zero columns: using the stochastic process construction of the IBP, and recalling that each customer draws new dishes independently of each other, we have that K_+ is the sum of independent Poisson random variables. Therefore, its distribution is also Poisson, with parameter equal to the sum of the parameters:

$$K_+|\alpha \sim \text{Poisson}\left(\alpha \sum_{j=1}^{D} \frac{1}{j}\right) = \text{Poisson}\left(\alpha H_D\right) \tag{35}$$

Expected number of non-zero entries: to obtain this, we need to go back to the finite version of the IBP, in order to derive the marginal distribution of each component Z_{dk}. To this end, we will integrate out π_k from the conditional distribution shown in Equation 2

$$\begin{aligned} p(Z_{dk}|\alpha) &= \int_{\pi_k} p(Z_{dk}|\pi_k)p(\pi_k)d\pi_k \\ &= \int_{\pi_k} \pi_k^{Z_{dk}}(1-\pi_k)^{1-Z_{dk}} \frac{\Gamma(\alpha/K+1)}{\Gamma(\alpha/K)} \pi_k^{\alpha/K-1} d\pi_k \\ &= \frac{\Gamma(\alpha/K+1)}{\Gamma(\alpha/K)} \int_{\pi_k} \pi_k^{Z_{dk}+\alpha/K-1}(1-\pi_k)^{1-Z_{dk}} d\pi_k \\ &= \frac{\Gamma(\alpha/K+1)}{\Gamma(\alpha/K)} \frac{\Gamma(Z_{dk}+\alpha/K)\Gamma(2-Z_{dk})}{\Gamma(\alpha/K+2)} \\ &= \frac{\Gamma(\alpha/K+1)}{\Gamma(\alpha/K)} \frac{\Gamma(Z_{dk}+\alpha/K)\Gamma(2-Z_{dk})}{\Gamma(\alpha/K+1)(\alpha/K+1)} \\ &= \frac{\Gamma(Z_{dk}+\alpha/K)\Gamma(2-Z_{dk})}{\Gamma(\alpha/K)(\alpha/K+1)} \end{aligned}$$

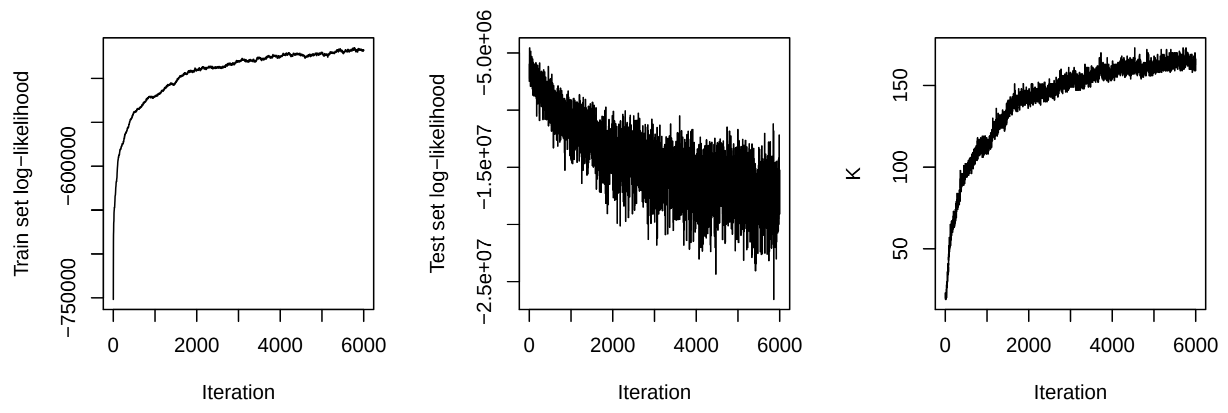
Thus:

$$\mathbb{P}(Z_{dk} = 1|\alpha) = \frac{\Gamma(1+\alpha/K)}{\Gamma(\alpha/K)(\alpha/K+1)} = \frac{\Gamma(\alpha/K)(\alpha/K)}{\Gamma(\alpha/K)(\alpha/K+1)} = \frac{\alpha/K}{\alpha/K+1}$$

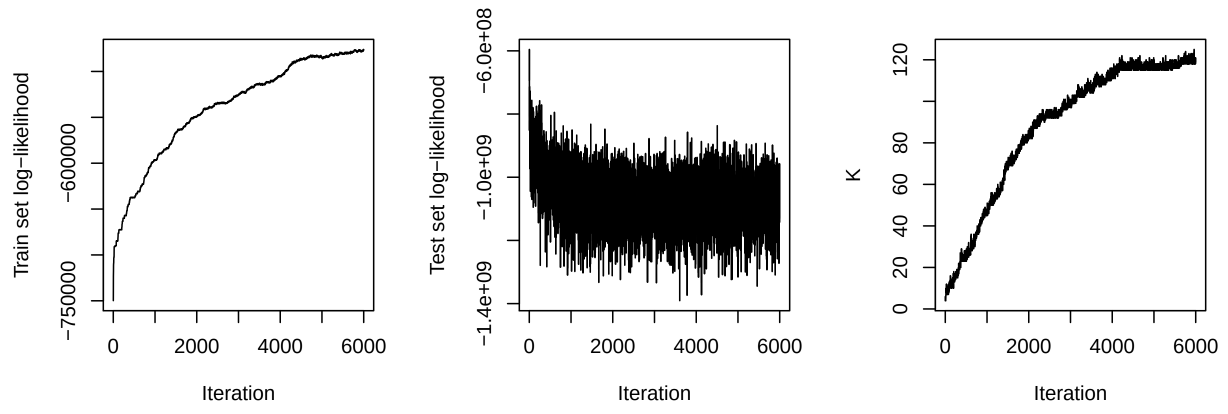
From this, the expected number of non-zero entries can be directly obtained:

$$\mathbb{E}\left(\sum_{d,k} Z_{dk}\right) = \sum_{d,k} \mathbb{E}(Z_{dk}) = \sum_{d,k} \frac{\alpha/K}{\alpha/K+1} = D\frac{\alpha}{\alpha/K+1} \overset{K\to\infty}{\to} \alpha D$$

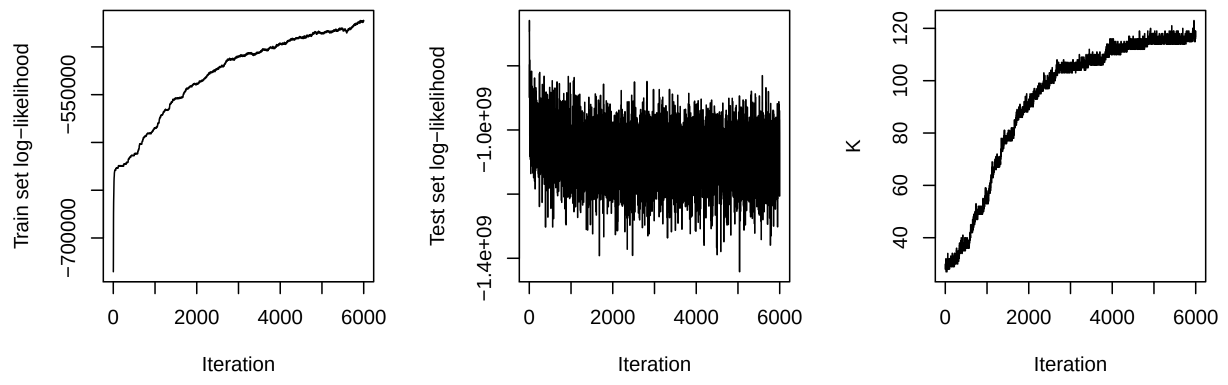
C Trace plots for the MNIST experiment



(a) Full NSFA ($\lambda = 10, \pi = 0.1$)



(b) Reduced NSFA ($\lambda = 1, \pi = 0$)



(c) Reduced NSFA ($\lambda = 10, \pi = 0.1$)

Figure 6: Traces for the log-likelihood in training set and test set, plus the trace for the number of active features K