Discrete Component Analysis

Wray Buntine[1] and Aleks Jakulin[2]

[1] Helsinki Institute for Information Technology (HIIT)
Dept. of Computer Science, PL 68
00014, University of Helsinki, Finland
`Wray.Buntine@hiit.fi`
[2] Department of Knowledge Technologies
Jozef Stefan Institute
Jamova 39, 1000 Ljubljana, Slovenia
`jakulin@acm.org`

Abstract. This article presents a unified theory for analysis of components in discrete data, and compares the methods with techniques such as independent component analysis, non-negative matrix factorisation and latent Dirichlet allocation. The main families of algorithms discussed are a variational approximation, Gibbs sampling, and Rao-Blackwellised Gibbs sampling. Applications are presented for voting records from the United States Senate for 2003, and for the Reuters-21578 newswire collection.

1 Introduction

Principal component analysis (PCA) [MKB79] is a key method in the statistical engineering toolbox. It is well over a century old, and is used in many different ways. PCA is also known as the Karhünen-Loève transform or Hotelling transform in image analysis, and a variation is latent semantic analysis (LSA) in text analysis [DDL$^+$90]. It is a kind of eigen-analysis since it manipulates the eigen-spectrum of the data matrix. It is usually applied to measurements and real valued data, and used for feature extraction or data summarization. LSA might not perform the centering step (subtracting the mean from each data vector prior to eigen-analysis) on the word counts for a document to preserve matrix sparseness, or might convert the word counts to real-valued `tf*idf` [BYRN99]. The general approach here is *data reduction.*

Independent component analysis (ICA, see [HKO01]) is in some ways an extension of this general approach, however it also involves the estimation of so-called latent, unobservable variables. This kind of estimation follows the major statistical methodology that deals with general unsupervised methods such as clustering and factor analysis. The general approach is called *latent structure analysis* [Tit], which is more recent, perhaps half a century old. The data is modelled in a way that admits unobservable variables, that influence the observable variables. Statistical inference is used to "reconstruct" the unobservable variables from the data jointly with general characteristics of the unobservable variables themselves. This is a theory with particular assumptions (i.e., a “model”), so the method may arrive at poor results.

Relatively recently the statistical computing and machine learning community has become aware of seemingly similar approaches for discrete observed data that appears under many names. The best known of these in this community are probabilistic latent semantic indexing (PLSI) [Hof99], non-negative matrix factorisation (NMF) [LS99] and latent Dirichlet allocation (LDA) [BNJ03]. Other variations are discussed later in Section 5. We refer to these methods jointly as *Discrete Component Analysis* (DCA), and this article provides a unifying model for them.

All the above approaches assume that the data is formed from individual observations (documents, individuals, images), where each observation is described through a number of variables (words, genes, pixels). All these approaches attempt to summarize or explain the similarities between observations and the correlations between variables by inferring latent variables for each observation, and associating latent variables with observed variables.

These methods are applied in the social sciences, demographics and medical informatics, genotype inference, text and image analysis, and information retrieval. By far the largest body of applied work in this area (using citation indexes) is in genotype inference due to the Structure program [PSD00]. A growing body of work is in text classification and topic modelling (see [GS04, BPT04]), and language modelling in information retrieval (see [AGvR03, BJ04, Can04]). As a guide, argued in the next section, the methods apply when PCA or ICA might be used, but the data is discrete.

Here we present in Section 3 a unified theory for analysis of components in discrete data, and compare the methods with related techniques in Section 5. The main families of algorithms discussed in Section 7 are a variational approximation, Gibbs sampling, and Rao-Blackwellised Gibbs sampling. Applications are presented in Section 8 for voting records from the United States Senate for 2003, and the use of components in subsequent classification.

2 Views of DCA

One interpretation of the DCA methods is that they are a way of approximating large sparse discrete matrices. Suppose we have a $500,000$ documents made up of $1,500,000$ different words. A document such as a page out of Dr. Seuss’s *The Cat in The Hat*, is first given as a *sequence of words*.

> So, as fast as I could, I went after my net. And I said, “With my net I can bet them I bet, I bet, with my net, I can get those Things yet!”

It can be put in the *bag of words* representation, where word order is lost. This yields a list of words and their counts in brackets:

> after(1) and(1) as(2) bet(3) can(2) could(1) fast(1) get(1) I(7) my(3) net(3) said(1) so(1) them(1) things(1) those(1) went(1) with(2) yet(1) .

Although the word 'you' never appears in the original, we do not include 'you (0)' in the representation since zeros are suppressed. This sparse vector can be represented as a vector in full word space with $1,499,981$ zeroes and the counts above making the non-zero entries in the appropriate places. Given a matrix made up of rows of such vectors of non-negative integers dominated by zeros, it is called here a *large sparse discrete matrix*.

Bag of words is a basic representation in information retrieval [BYRN99]. The alternative is a sequence of words. In DCA, either representation can be used and the models act the same, up to any word order effects introduced by incremental algorithms. This detail is made precise in subsequent sections.

In this section, we argue from various perspectives that large sparse discrete data is not well suited to standard PCA or ICA methods.

2.1 Issues with PCA

PCA has been normally applied to numerical data, where individual instances are vectors of real numbers. However, many practical datasets are based on vectors of integers, non-negative counts or binary values. For example, a particular word cannot have a negative number of appearances in a document. The vote of a senator can only take three values: Yea, Nay or Not Voting. We can transform all these variables into real numbers using `tf*idf`, but this is a linear weighting that does not affect the shape of a distribution.

With respect to modelling count data in linguistic applications, Dunning makes the following warning [Dun94]:

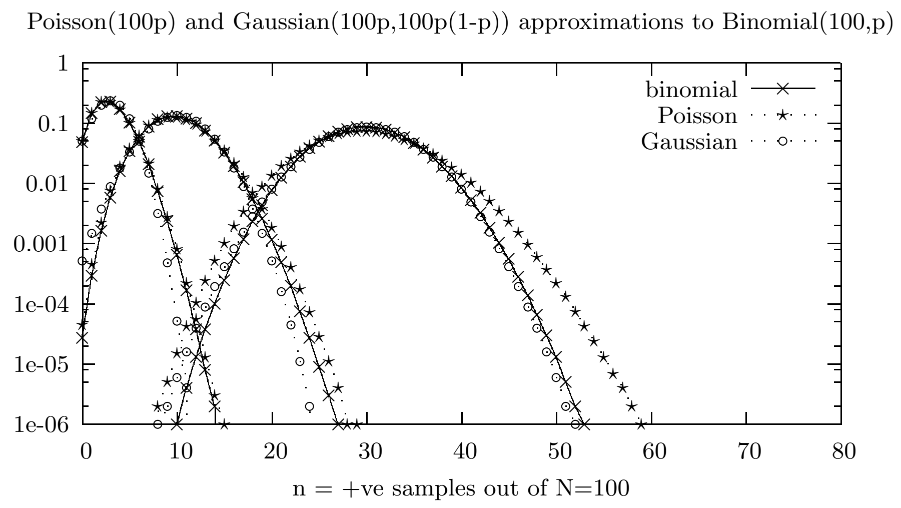
> Statistics based on the assumption of normal distribution are invalid in most cases of statistical text analysis unless either enormous corpora are used, or the analysis is restricted to only the very most common words (that is, the ones least likely to be of interest). This fact is typically ignored in much of the work in this field. Using such invalid methods may seriously overestimate the significance of relatively rare events. Parametric statistical analysis based on the binomial or multinomial distribution extends the applicability of statistical methods to much smaller texts than models using normal distributions and shows good promise in early applications of the method.

While PCA is not always considered a method based on Gaussians, it can be justified using Gaussian distributions [Row98, TB99]. Moreover, PCA is justified using a least squares distance measure, and most of the properties of Gaussians follow from the distance measure alone. Rare events correspond to points far away under an L_2 norm.

Fundamentally, there are two different kinds of large sample approximating distributions that dominate discrete statistics: the Poisson and the Gaussian. For instance, a large sample binomial is approximated as a Poisson[3] when the

[3] This is a distribution on integers where a rate is given for events to occur, and the distribution is over the total number of events counted.

probability is small and as a Gaussian otherwise [Ros89]. Figure 2.1 illustrates this by showing the Gaussian and Poisson approximations to a binomial with sample size $N = 100$ for different proportions ($p = 0.03, 0.01, 0.03$). Plots are done with probability in log scale so the errors for low probability values are highlighted. One can clearly see the problem here: the Gaussian provides a rea-



sonable approximate for medium values of the proportion p but for small values it severely underestimates low probabilities. When these low probability events occur, as they always will, the model becomes distorted.

Thus in image analysis based on analogue to digital converters, where data is counts, Gaussian errors can sometimes be assumed, but the Poisson should be used if counts are small. DCA then avoids Gaussian modelling of the data, using a Poisson or multinomial directly.

Another critique of the general style of PCA comes from the psychology literature, this time it is used as a justification for DCA [GS02]. Griffiths and Steyvers argue against the least squares distance of PCA:

> While the methods behind LSA were novel in scale and subject, the suggestion that similarity relates to distance in psychological space has a long history (Shepard, 1957). Critics have argued that human similarity judgments do not satisfy the properties of Euclidean distances, such as symmetry or the triangle inequality. Tversky and Hutchinson (1986) pointed out that Euclidean geometry places strong constraints on the number of points to which a particular point can be the nearest neighbor, and that many sets of stimuli violate these constraints.

They also considered power law arguments which PCA violates for associated words.

2.2 Component Analysis as Approximation

In the data reduction approach for PCA, one seeks to reduce each J-dimensional data vector to a smaller K-dimensional vector. This can be done by approximating the full data matrix as a product of smaller matrices, one representing the reduced vectors called the component/factor *score matrix*, and one representing a data independent part called the component/factor *loading matrix*, as shown in Figure 1. In PCA according to least squares theory, this approximation is made by eliminating the lower-order eigenvectors, the least contributing components [MKB79].

$$\text{I documents}\left\{\underbrace{\overbrace{\begin{pmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,J} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,J} \\ \vdots & \vdots & \ddots & \vdots \\ w_{I,1} & w_{I,2} & \cdots & w_{I,J} \end{pmatrix}}^{J\text{ words}}}_{\text{data matrix}}\right. \simeq \underbrace{\overbrace{\begin{pmatrix} l_{1,1} & \cdots & l_{1,K} \\ l_{2,1} & \cdots & l_{2,K} \\ \vdots & \ddots & \vdots \\ l_{I,1} & \cdots & l_{I,K} \end{pmatrix}}^{K\text{ components}}}_{\text{score matrix}} * \underbrace{\left.\overbrace{\begin{pmatrix} \theta_{1,1} & \theta_{2,1} & \cdots & \theta_{J,1} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{1,K} & \theta_{2,K} & \cdots & \theta_{J,K} \end{pmatrix}}^{J\text{ words}}\right\}}_{\text{loading matrix}^T}\text{K components}$$

Fig. 1. The matrix approximation view

If there are I documents, J words and K components, then the matrix on the left has $I * J$ entries and the two matrices on the right have $(I + J) * K$ entries. This represents a simplification when $K \ll I, J$. We can view DCA methods as seeking the same goal in the case where the matrices are sparse and discrete.

When applying PCA to large sparse discrete matrices, or LSA using word count data interpretation of the components, if it is desired, becomes difficult (it was not a goal of the original method [DDL$^+$90]). Negative values appear in the component matrices, so they cannot be interpreted as "typical documents" in any usual sense. This applies to many other kinds of sparse discrete data: low intensity images (such as astronomical images) and verb-noun data used in language models introduced by [PTL93], for instance.

The cost function being minimized then plays an important role. DCA places constraints on the approximating score matrix and loading matrix in Figure 1 so that they are also non-negative. It also uses an entropy distance instead of a least squares distance.

2.3 Independent Components

Independent component analysis (ICA) was also developed as an alternative to PCA. Hyvänen and Oja [HO00] argue that PCA methods merely find uncorrelated components. ICA then was developed as a way of representing multivariate data with truly *independent* components. In theory, PCA approximates this also if the data is Gaussian [TB99], but in practice it rarely is.

The basic formulation is that a K-dimensional data vector $\boldsymbol{w}$ is a linear invertible function of K independent components represented as a K-dimensional latent vector $\boldsymbol{l}$, $\boldsymbol{w} = \boldsymbol{\Theta l}$ for a square invertible matrix $\boldsymbol{\Theta}$. Note the ICA assumes $J = K$ in our notation. $\boldsymbol{\Theta}$ plays the same role as the loading matrix above. For some univariate density model U, the independent components are distributed as $p(\boldsymbol{l} \,|\, U) = \prod_k p(l_k \,|\, U)$, thus one can get a likelihood formula $p(\boldsymbol{w} \,|\, \boldsymbol{\Theta}, U)$ using the above equality[4].

The Fast ICA algorithm [HO00] can be interpreted as a maximum likelihood approach based on this model and likelihood formula. In the sparse discrete case, however, this formulation breaks down for the simple reason that $\boldsymbol{w}$ is mostly zeros: the equation can only hold if $\boldsymbol{l}$ and $\boldsymbol{\Theta}$ are discrete as well and thus the gradient-based algorithms for ICA cannot be justified. To get around this in practice, when applying ICA to documents [BKG03], word counts are sometimes first turned into `tf*idf` scores [BYRN99].

To arrive at a formulation more suited to discrete data, we can relax the equality in ICA (i.e., $\boldsymbol{w} = \boldsymbol{\Theta l}$) to be an expectation:

$$\mathbb{E}_{\boldsymbol{w} \sim p(\boldsymbol{w}|\boldsymbol{l},U)} \left[\boldsymbol{w}\right] \; = \; \boldsymbol{\Theta l} \; .$$

We still have independent components, but a more robust relationship between the data and the score vector. Correspondence between ICA and DCA has been noted in [BJ04, Can04]. With this expectation relationship, the dimension of $\boldsymbol{l}$ can now be less than the dimension of $\boldsymbol{w}$, $K < J$, and thus $\boldsymbol{\Theta}$ would be a rectangular matrix.

3 The Basic Model

A good introduction to these models from a number of viewpoints is by [BNJ03, Can04, BJ04]. Here we present a general model. The notation of words, bags and documents will be used throughout, even though other kinds of data representations also apply. In statistical terminology, a word is an observed variable, and a document is a data vector (a list of observed variables) representing an instance. In machine learning terminology, a word is a feature, a bag is a data vector, and a document is an instance. Notice that the bag collects the words in

[4] By a change of coordinates

$$p(\boldsymbol{w} \,|\, \boldsymbol{\Theta}, U) \; = \; \frac{1}{\det(\boldsymbol{\Theta})} \prod_k p\left(\left(\boldsymbol{\Theta}^{-1}\boldsymbol{w}\right)_k \,|\, U\right)$$

the document and loses their ordering. The bag is represented as a data vector $\boldsymbol{w}$. It is now J-dimensional. The latent, hidden or unobserved vector $\boldsymbol{l}$ called the component *scores* is K-dimensional. The term *component* is used here instead of topic, factor or cluster. The parameter matrix is the previously mentioned component loading matrix $\boldsymbol{\Theta}$, and is $J \times K$.

At this point, it is also convenient to introduce the symbology used throughout the paper. The symbols summarised in Table 1 will be introduced as we go.

I	number of documents
(i)	subscript to indicate document, sometimes dropped
J	number of different words, size of the dictionary
K	number of components
$L_{(i)}$	number of words in document i
S	number of words in the collection, $\sum_i L_{(i)}$
$\boldsymbol{w}_{(i)}$	vector of J word counts in document i, row totals of $\boldsymbol{V}$, entries $w_{j,(i)}$
$\boldsymbol{c}_{(i)}$	vector of K component counts for document i, column totals of $\boldsymbol{V}$
$\boldsymbol{V}$	matrix of word counts per component, dimension $J \times K$, entries $v_{j,k}$
$\boldsymbol{l}_{(i)}$	vector of K component scores for document i, entries $l_{k,(i)}$
$\boldsymbol{m}_{(i)}$	$\boldsymbol{l}_{(i)}$ normalised, entries $m_{k,(i)}$
$\boldsymbol{k}_{(i)}$	vector of $L_{(i)}$ sequential component assignments for the words in document i, entries $k_{l,(i)} \in [1, \ldots, K]$
$\boldsymbol{\Theta}$	component loading matrix, dimension $J \times K$, entries $\theta_{j,k}$
$\boldsymbol{\theta}_{\cdot,k}$	component loading vector for component k, a column of $\boldsymbol{\Theta}$
$\boldsymbol{\alpha}, \boldsymbol{\beta}$	K-dimensional parameter vectors for component priors

Table 1. Summary of major symbols

3.1 Bags or Sequences of Words?

For a document $\boldsymbol{x}$ represented as a sequence of words, if $\boldsymbol{w} = \text{bag}(\boldsymbol{x})$ is its bagged form, the bag of words, represented as a vector of counts. In the simplest case, one can use a multinomial with sample size $L = |\boldsymbol{x}|$ and vocabulary size $J = |\boldsymbol{w}|$ to model the bag, or alternatively L independent discrete distributions[5] with J outcomes to model each x_l. The bag $\boldsymbol{w}$ corresponds to the sequence $\boldsymbol{x}$ with the order lost, thus there are $\frac{(\sum_j w_j)!}{\prod_j w_j!}$ different sequences that map to the same bag $\boldsymbol{w}$. The likelihoods for these two simple models thus differ by just this combinatoric term.

Note that some likelihood based methods such as maximum likelihood, some Bayesian methods, and some other fitting methods (for instance, a cross validation technique) use the likelihood as a black-box function. They take values or

[5] The *discrete* distribution is the multivariate form of a Bernoulli where an index $j \in \{0, 1, ..., J-1\}$ is sampled according to a J-dimensional probability vector.

derivatives but otherwise do not further interact with the likelihood. The combinatoric term mapping bag to sequence representations can be ignored here safely because it does not affect the fitting of the parameters for $\mathcal{M}$. Thus for these methods, it is irrelevant whether the data is treated as a bag or as a sequence. This is a general property of multinomial data.

Thus, while we consider bag of words in this article, most of the theory applies equally to the sequence of words representation[6]. Implementation can easily address both cases with little change to the algorithms, just to the data handling routines.

3.2 General DCA

The general formulation introduced in Section 2.3 is an unsupervised version of a linear model, and it applies to the bag of words $\boldsymbol{w}$ as

$$\mathbb{E}_{\boldsymbol{w}\sim p(\boldsymbol{w}|\boldsymbol{l},\boldsymbol{\Theta})}\left[\boldsymbol{w}\right] \;=\; \boldsymbol{\Theta}\boldsymbol{l} \tag{1}$$

The expected value (or mean) of the data is given by the dot product of the component loading matrix $\boldsymbol{\Theta}$ and some latent component scores $\boldsymbol{l}$.

In full probability (or Bayesian) modelling [GCSR95], we are required to give a distribution for all the non-deterministic values in a model, including model parameters and the latent variables. In likelihood modelling [CB90], we are required to give a distribution for all the data values in a model, including observed and latent variables. These are the core methodologies in computational statistics, and most others extend these two. The distribution for the data is called a likelihood in both methodologies.

The likelihood of a document is the primary way of evaluating a probabilistic model. Although likelihood is not strictly a probability in classical statistics, we can interpret them as a probability that a probabilistic model $\mathcal{M}$ would generate a document $\boldsymbol{x}$, $P(\boldsymbol{x}|\mathcal{M})$. On the other hand, it is also a way of determining whether the document is usual or unusual: documents with low likelihood are often considered to be outliers or anomalies. If we trust our documents, low likelihoods indicate problems with the model. If we trust out model, a low likelihood indicates problems with a document.

Thus to complete the above formulation for DCA, we need to give distributions matching the constraint in Equation (1), to specify the likelihood. Distributions are needed for:

- how the sequence $\boldsymbol{x}$ or bag $\boldsymbol{w}$ is distributed given its mean $\boldsymbol{\Theta}\boldsymbol{l}$ formed from the component loading matrix,

[6] Some advanced fitting methods such as Gibbs sampling do not treat the likelihood as a black-box. They introduce latent variables that expands the functional form of the likelihood, and they may update parts of a document in turn. For these, ordering effects can be incurred by bagging a document, since updates for different parts of the data will now be done in a different order. But the combinatoric term mapping bag to sequence representations will still be ignored and the algorithms are effectively the same up to the ordering affects.

- how the component scores $\boldsymbol{l}$ are distributed,
- and if full probability modelling is used, how the component loading matrix $\boldsymbol{\Theta}$ is distributed apriori, as well as any parameters.

The formulation of Equation (1) is also called an *admixture model* in the statistical literature [PSD00]. This is in contrast with a *mixture model* [GCSR95] which uses a related constraint

$$\mathbb{E}_{\boldsymbol{w}\sim p(\boldsymbol{w}|\boldsymbol{l})}\left[\boldsymbol{w}\right] \;=\; \boldsymbol{\theta}_{\cdot,k} \ ,$$

for some latent variable k representing the single latent component for $\boldsymbol{w}$. Since k is unobserved, this also corresponds to making a weighted sum of the probability distributions for each $\boldsymbol{\theta}_{\cdot,k}$.

4 The Model Families

This section introduces some forms of DCA using specific distributions for the sequence $\boldsymbol{x}$ or bag $\boldsymbol{w}$ and the component scores $\boldsymbol{l}$. The fundamental model here is the Gamma-Poisson Model (GP model for short). Other models can be presented as variations. The probability for a document is given for each model, both for the case where the latent variables are known (and thus are on the right-hand side), and for the case where the latent variables are included in the left-hand side.

4.1 The Gamma-Poisson Model

The general Gamma-Poisson form of DCA, introduced as GaP [Can04] is now considered in more detail:

- Document data is supplied in the form of *word counts*. The word count for each word type is w_j. Let L be the total count, so $L = \sum_j w_j$.
- The document also has *component scores* $\boldsymbol{l}$ that indicate the amount of the component in the document. These are latent or unobserved. The entries l_k are independent and gamma distributed

$$l_k \;\sim\; \text{Gamma}(\alpha_k, \beta_k) \qquad\qquad \text{for } k = 1, \ldots, K.$$

  The β_k affects scaling of the components[7], while α_k changes the shape of the distribution, shown in Figure 2.
- There is a *component loading matrix* $\boldsymbol{\Theta}$ of size $J \times K$ with entries $\theta_{j,k}$ that controls the partition of features amongst each component. In the matrix, each column for component k is normalised across the features, meaning that $\sum_j \theta_{j,k} = 1$. Thus each column represents the proportions of words/features in component k.

[7] Conventions for the gamma vary. Sometimes a parameter $1/\beta_k$ is used. Our convention is revealed in Equation (2).

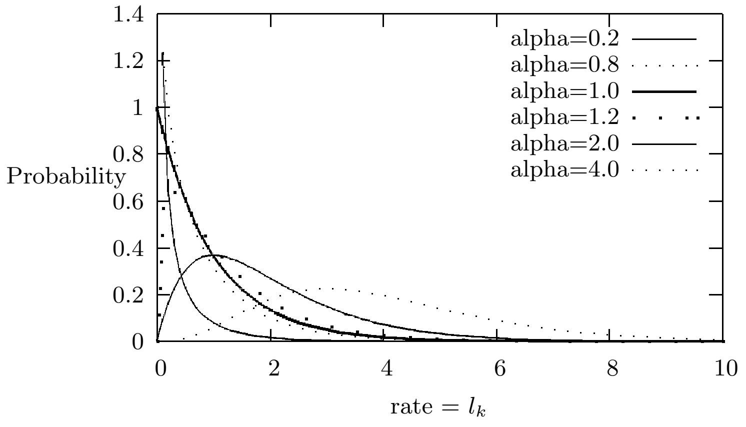


Fig. 2. Gamma distribution for different values of α_k.

- The observed data $\boldsymbol{w}$ is now Poisson distributed, for each j

$$w_j \sim \text{Poisson}\left((\boldsymbol{\Theta}\boldsymbol{l})_j\right) .$$

- The K parameters (α_k, β_k) to the gamma distributions give K-dimensional parameter vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. Initially these vectors will be treated as constants, and their estimation is the subject of later work.
- When using Bayesian of full probability modelling, a prior is needed for $\boldsymbol{\Theta}$. A Dirichlet prior can be used for each k-th component of $\boldsymbol{\Theta}$ with J prior parameters γ_j, so $\boldsymbol{\theta}_{\cdot,k} \sim \text{Dirichlet}_J(\boldsymbol{\gamma})$. In practice we use a Jeffreys' prior, which has $\gamma_j = 0.5$. The use of a Dirichlet has no strong justification other than being conjugate [GCSR95], but the Jeffreys' prior has some minimax properties [CB94] that make it more robust.

The hidden or latent variables here are the component scores $\boldsymbol{l}$. The model parameters are the gamma parameters $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$, and the component loading matrix $\boldsymbol{\Theta}$. Denote the model as GP standing for Gamma-Poisson. The full likelihood for each document, $p(\boldsymbol{w}, \boldsymbol{l} \,|\, \boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\Theta}, K, \text{GP})$, is composed of two parts. The first part comes from K independent gamma distributions for l_k, and the second part comes from J independent Poisson distributions with parameters $\sum_k l_k \theta_{j,k}$.

$$\overbrace{\prod_k \frac{\beta_k^{\alpha_k} l_k^{\alpha_k - 1} \exp\{-\beta_k l_k\}}{\Gamma(\alpha_k)}}^{\text{likelihood of } \boldsymbol{l}} \overbrace{\prod_j \frac{\left(\sum_k l_k \theta_{j,k}\right)^{w_j} \exp\left\{-\left(\sum_k l_k \theta_{j,k}\right)\right\}}{w_j!}}^{\text{likelihood of } \boldsymbol{w} \text{ given } \boldsymbol{l}} \tag{2}$$

4.2 The Conditional Gamma-Poisson Model

In practice, when fitting the parameters $\boldsymbol{\alpha}$ in the GP or DM model, it is often the case that the α_k go very small. Thus, in this situation, perhaps 90% of the component scores l_k are negligible, say less than 10^{-8} once normalised. Rather than maintaining these negligible values, we can allow component scores to be zero with some finite probability. The Conditional Gamma-Poisson Model, denoted CGP for short, introduces this capability. In retrospect, CGP is a sparse GP with an additional parameter per component to encourage sparsity.

The CGP model extends the Gamma-Poisson model by making the l_k zero sometimes. In the general case, the l_k are independent and zero with probability ρ_k and otherwise gamma distributed with probability $1-\rho_k$.

$$l_k \sim \text{Gamma}(\alpha_k, \beta_k) \qquad \text{for } k = 1, \ldots, K .$$

Denote the model as CGP standing for Conditional Gamma-Poisson, and the full likelihood is now $p(\boldsymbol{w}, \boldsymbol{l} \,|\, \boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\rho}, \boldsymbol{\Theta}, K, \text{CGP})$. The full likelihood for each document, modifying the above Equation (2), replaces the term inside $\prod_k$ with

$$(1-\rho_k)\frac{\beta_k^{\alpha_k} l_k^{\alpha_k - 1}\exp\{-\beta_k l_k\}}{\Gamma(\alpha_k)} + \rho_k 1_{l_k=0} \tag{3}$$

4.3 The Dirichlet-Multinomial Model

The Dirichlet-multinomial form of DCA was introduced as MPCA. In this case, the normalised latent variables $\boldsymbol{m}$ are used, and the total word count L is not modelled.

$$\boldsymbol{m} \sim \text{Dirichlet}_K(\boldsymbol{\alpha}) \ , \qquad \boldsymbol{w} \sim \text{Multinomial}(L, \boldsymbol{\Theta}\boldsymbol{m})$$

The first argument to the multinomial is the total count, the second argument is the vector of probabilities. Denote the model as DM, and the full likelihood is now $p(\boldsymbol{w}, \boldsymbol{m} \,|\, L, \boldsymbol{\alpha}, \boldsymbol{\Theta}, K, \text{DM})$. The full likelihood for each document becomes:

$$C^L_{w_1,\ldots,w_J}\Gamma\left(\sum_k \alpha_k\right)\prod_k \frac{m_k^{\alpha_k-1}}{\Gamma(\alpha_k)}\prod_j\left(\sum_k m_k\theta_{j,k}\right)^{w_j} \tag{4}$$

where $C^L_{\boldsymbol{w}}$ is L choose $w_1, \ldots, w_J$. This model can also be derived from the Gamma-Poisson model, shown in the next section.

4.4 A Multivariate Version

Another variation of the methods is to allow grouping of the count data. Words can be grouped into separate variable sets. These groups might be "title words," "body words," and "topics" in web page analysis or "nouns," "verbs" and "adjectives" in text analysis. The groups can be treated with separate discrete distributions, as below. The J possible word types in a document are partitioned into G groups $B_1, \ldots, B_G$. The total word counts for each group g is denoted $L_g = \sum_{j \in B_g} w_j$. If the vector $\boldsymbol{w}$ is split up into G vectors $\boldsymbol{w}_g = \{w_j : j \in B_g\}$, and the matrix $\boldsymbol{\Theta}$ is now normalised by group in each row, so $\sum_{j \in B_g} \theta_{j,k} = 1$, then a multivariate version of DCA is created so that for each group g,

$$\boldsymbol{w}_g \;\sim\; \text{Multinomial}\left(L_g, \left\{ \sum_k m_k \theta_{j,k} \;:\; j \in B_g \right\} \right) \; .$$

Fitting and modelling methods for this variation are related to LDA or MPCA, and will not be considered in more detail here. This has the advantage that different kinds of words have their own multinomial and the distribution of different kinds is ignored. This version is demonstrated subsequently on US Senate voting records, where each multinomial is now a single vote for a particular senator.

5 Related Work

These sections begins by relating the main approaches to each other, then placing them in the context of exponential family models, and finally a brief history is recounted.

5.1 Correspondences

Various published cases of DCA can be represented in terms of this format, as given in Table 2. A multinomial with total count L and J possible outcomes is the bagged version of L discrete distributions with J possible outcomes. In the table, *NA* indicates that this aspect of the model was not required to be specified because the methodology made no use of it. Note that NMF used a cost function

Table 2. Previously Published Models

Name	Bagged	Components	$p(\boldsymbol{x}/\boldsymbol{w} \mid \boldsymbol{\Theta}, \boldsymbol{l})$	$p(\boldsymbol{l}/\boldsymbol{m})$
NMF [LS99]	yes	$\boldsymbol{l}$	Poisson	*NA*
PLSI [Hof99]	no	$\boldsymbol{m}$	discrete	*NA*
LDA [BNJ03]	no	$\boldsymbol{m}$	discrete	Dirichlet
MPCA [Bun02]	yes	$\boldsymbol{m}$	multinomial	Dirichlet
GaP [Can04]	yes	$\boldsymbol{l}$	Poisson	gamma

formulation, and thus avoided defining likelihood models. It is shown later that its cost function corresponds to a Gamma-Poisson with parameters $\boldsymbol{\alpha} = \boldsymbol{\beta} = \boldsymbol{0}$ (i.e., all zero).

LDA has the multinomial of MPCA replaced by a sequence of discrete distributions, and thus the choose term drops, as per Section 3.1. PLSI is related to LDA but lacks a prior distribution on $\boldsymbol{m}$. It does not model these latent variables using full probability theory, but instead using a weighted likelihood method

[Hof99]. Thus PLSI is a non-Bayesian version of LDA, although its weighted likelihood method means it accounts for over-fitting in a principled manner.

LDA and MPCA also have a close relationship to GaP (called GP here). If the parameter $\boldsymbol{\alpha}$ is treated as known and not estimated from the data, and the $\boldsymbol{\beta}$ parameter vector has the same value for each β_k, then L is *aposteriori* independent of $\boldsymbol{m}$ and $\boldsymbol{\Theta}$. In this context LDA, MPCA and GaP are equivalent models ignoring representational issues.

Lemma 1. *Given a Gamma-Poisson model of Section 4.1 where the* $\boldsymbol{\beta}$ *parameter is a constant vector with all entries the same,* β*, the model is equivalent to a Dirichlet-multinomial model of Section 4.3 where* $m_k = l_k / \sum_k l_k$*, and in addition*

$$L \sim \textit{Poisson-Gamma}\left(\sum_k \alpha_k, \beta, 1\right)$$

Proof. Consider the Gamma-Poisson model. The sum $L = \sum_j w_j$ of Poisson variables $\boldsymbol{w}$ has the distribution of a Poisson with parameter given by the sum of their means. When the sum of Poisson variables is known, the set of Poisson variables has a multinomial distribution conditioned on the sum (the total count) [Ros89]. The Poisson distributions on $\boldsymbol{w}$ then is equivalent to:

$$L \sim \text{Poisson}\left(\sum_k l_k\right) , \qquad \boldsymbol{w} \sim \text{Multinomial}\left(L, \frac{1}{\sum_k l_k}\boldsymbol{\Theta l}\right) .$$

Moreover, if the $\boldsymbol{\beta}$ parameter is constant, then $m_k = l_k / \sum_k l_k$ is distributed as $\text{Dirichlet}_K(\boldsymbol{\alpha})$, and $\sum_k l_k$ is distributed independently as a $\text{Gamma}(\sum_k \alpha_k, \beta)$. The second distribution above can then be represented as

$$\boldsymbol{w} \sim \text{Multinomial}\left(L, \boldsymbol{\Theta m}\right) .$$

Note also, that marginalising out $\sum_k l_k$ convolves a Poisson and a gamma distribution to produce a Poisson-Gamma distribution for L [BS94].

If $\boldsymbol{\alpha}$ is estimated from the data in GaP, then the presence of the observed L will influence $\boldsymbol{\alpha}$, and thus the other estimates such as of $\boldsymbol{\Theta}$. In this case, LDA and MPCA will no longer be effectively equivalent to GaP. Note, Canny recommends fixing $\boldsymbol{\alpha}$ and estimating $\boldsymbol{\beta}$ from the data [Can04].

To complete the set of correspondences, note that in Section 7.1 it is proven that NMF corresponds to a maximum likelihood version of GaP, and thus it also corresponds to a maximum likelihood version of LDA, MPCA, and PLSI.

5.2 Notes on the Exponential Family

For the general DCA model of Section 3.2, when $p(\boldsymbol{w} \,|\, \boldsymbol{\Theta l})$ is in the so-called exponential family distributions [GCSR95], the expected value of $\boldsymbol{w}$ is referred to as the *dual parameter*, and it is usually the parameter we know best. For the Bernoulli with probability p, the dual parameter is p, for the Poisson with rate λ, the dual parameter is λ, and for the Gaussian with mean μ, the dual parameter is the mean. Our formulation, then, can be also be interpreted as letting $\boldsymbol{w}$ be exponential family with dual parameter given by $(\boldsymbol{\Theta l})$. Our formulation then generalises PCA in the same way that a linear model [MN89] generalises linear regression.

Note, an alternative has also been presented [CDS01] where $\boldsymbol{w}$ has an exponential family distribution with natural parameters given by $(\boldsymbol{\Theta l})$. For the Bernoulli with probability p, the natural parameter is $\log(p/(1-p))$, for the Poisson with rate λ, the natural parameter is $\log \lambda$ and for the Gaussian with mean μ, the natural parameter is the mean. This formulation generalises PCA in the same way that a generalised linear model [MN89] generalises linear regression.

5.3 Historical notes

Several independent groups within the statistical computing and machine learning community have contributed to the development of the DCA family of methods. Some original research includes the following: grade of membership (GOM) [WM82], probabilistic latent semantic indexing (PLSI) [Hof99], non-negative matrix factorisation (NMF) [LS99], genotype inference using admixtures [PSD00], latent Dirichlet allocation (LDA) [BNJ03], and Gamma-Poisson models (GaP) [Can04]. Modifications and algorithms have also been explored as multinomial PCA (MPCA) [Bun02] and multiple aspect modelling [ML02].

The first clear enunciation of the large-scale model in its Poisson form comes from [LS99], and in its multinomial form from [Hof99] and [PSD00]. The first clear expression of the problem as a latent variable problem is given by [PSD00]. The relationship between LDA and PLSI and that NMF was a Poisson version of LDA was first pointed out by [Bun02], and proven in [GG05]. The connections to ICA come from [BJ04] and [Can04]. The general Gamma-Poisson formulation, perhaps the final generalisation to this line of work, is in [Can04].

Related techniques in the statistical community can be traced back to Latent Class Analysis developed in the 1950's, and a rich theory has since developed relating the methods to correspondence analysis and other statistical techniques [vGv99].

6 Component Assignments for Words

In standard mixture models, each document in a collection is assigned to one latent component. The DCA family of models can be interpreted as making each word in each document be assigned to one latent component. To see this, we introduce another latent vector which represents the component assignments for different words. As in Section 3.1, this can be done using a bag of components or a sequence of components representation, and no effective change occurs in the basic models, or in the algorithms so derived. What this does is expand out the term $\boldsymbol{\Theta l}$ into parts, treating it as if it is the result of marginalising out some latent variable.

We introduce a K-dimensional discrete latent vector $\boldsymbol{c}$ whose total count is L, the same as the word count. The count c_k gives the number of words in the document appearing in the k-th component. Its posterior mean makes a good diagnostic and interpretable result. A document from the sports news might have 50 "football" words, 10 "German" words, 20 "sports fan" words and 20 "general vocabulary" words.

This latent vector is derived from a larger latent matrix, $\boldsymbol{V}$ of size $J \times K$ and entries $v_{j,k}$. This has row totals w_j as given in the observed data and column totals c_k. Vectors $\boldsymbol{w}$ and $\boldsymbol{c}$ are these word appearance counts and component appearance counts, respectively, based on summing rows and columns of matrix $\boldsymbol{V}$. This is shown in Figure 3.

$$
\text{J words}\left\{
\begin{array}{c}
\overbrace{\hphantom{v_{1,1}\; v_{1,2}\; \cdots\; v_{1,K}}}^{K \text{ components}} \\
\begin{pmatrix}
v_{1,1} & v_{1,2} & \cdots & v_{1,K} \\
v_{2,1} & v_{2,2} & \cdots & v_{2,K} \\
\vdots & \vdots & \ddots & \vdots \\
v_{J,1} & v_{J,2} & \cdots & v_{J,K}
\end{pmatrix} \\
\hline
c_1 \quad c_2 \quad \cdots \quad c_K
\end{array}\right.
\begin{array}{|c}
w_1 \\ w_2 \\ \vdots \\ w_J
\end{array}
$$

Fig. 3. A representation of a document as a contingency table.

The introduction of the latent matrix $\boldsymbol{V}$ changes the forms of the likelihoods, and makes the development and analysis of algorithms easier. This section catalogues the likelihood formula, to be used when discussing algorithms.

6.1 The Gamma-Poisson Model

With the new latent matrix $\boldsymbol{V}$, the distributions underlying the Gamma-Poisson model become:

$$
\begin{aligned}
l_k &\sim \text{Gamma}(\alpha_k, \beta_k) \\
c_k &\sim \text{Poisson}\,(l_k) \\
w_j &= \sum_k v_{j,k} \qquad \text{where } v_{j,k} \sim \text{Multinomial}\,(c_k, \boldsymbol{\theta}_{\cdot,k}) \ .
\end{aligned} \tag{5}
$$

The joint likelihood for a document, $p(\boldsymbol{V}, \boldsymbol{l} \,|\, \boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\Theta}, K, \text{GP})$ (the $\boldsymbol{w}$ are now derived quantities so not represented), thus becomes, after some rearrangement

$$
\prod_k \frac{\beta_k^{\alpha_k} l_k^{c_k+\alpha_k-1} \exp\{-(\beta_k+1)l_k\}}{\Gamma(\alpha_k)} \prod_{j,k} \frac{\theta_{j,k}^{v_{j,k}}}{v_{j,k}!} \ . \tag{6}
$$

Note that $\boldsymbol{l}$ can be marginalised out, yielding

$$\prod_k \frac{\Gamma(c_k+\alpha_k)}{\Gamma(\alpha_k)} \frac{\beta_k^{\alpha_k}}{(\beta_k+1)^{c_k+\alpha_k}} \prod_{j,k} \frac{\theta_{j,k}^{v_{j,k}}}{v_{j,k}!} \tag{7}$$

and the posterior mean of l_k given $\boldsymbol{c}$ is $(c_k+\alpha_k)/(1+\beta_k)$. Thus each $c_k \sim$ Poisson-Gamma$(\alpha_k, \beta_k, 1)$.

6.2 The Conditional Gamma-Poisson Model

The likelihood follows the GP case, except that with probability ρ_k, $l_k = 0$ and thus $c_k = 0$. The joint likelihood, $p(\boldsymbol{V}, \boldsymbol{l} \,|\, \boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\rho}, \boldsymbol{\Theta}, K, \mathrm{CGP})$, thus becomes, after some rearrangement

$$\begin{aligned}\prod_k \Bigg((1-\rho_k) \left(\frac{\beta_k^{\alpha_k} l_k^{c_k+\alpha_k-1} \exp\{-(\beta_k+1) l_k\}}{\Gamma(\alpha_k)} \prod_j \frac{\theta_{j,k}^{v_{j,k}}}{v_{j,k}!} \right) \qquad\qquad (8)\\ + \rho_k \left(1_{l_k=0} 1_{c_k=0} \prod_j 1_{v_{j,k}=0} \right) \Bigg) \ .\end{aligned}$$

Note that $\boldsymbol{l}$ can be marginalised out, yielding

$$\prod_k \left((1-\rho_k) \frac{\Gamma(c_k+\alpha_k)}{\Gamma(\alpha_k)} \frac{\beta_k^{\alpha_k}}{(\beta_k+1)^{c_k+\alpha_k}} + \rho_k 1_{c_k=0} \right) \prod_j \frac{\theta_{j,k}^{v_{j,k}}}{v_{j,k}!} \tag{9}$$

The $\theta_{j,k}$ can be pulled out under the constraint $\sum_j v_{j,k} = c_k$. The posterior mean of l_k given $\boldsymbol{c}$ is $(1-\rho_k)(c_k+\alpha_k)/(1+\beta_k)$.

6.3 The Dirichlet-Multinomial Model

For the Dirichlet-multinomial model, a similar reconstruction applies:

$$\begin{aligned} \boldsymbol{m} &\sim \mathrm{Dirichlet}_K(\boldsymbol{\alpha}) \qquad (10)\\ c_k &\sim \mathrm{Multinomial}\left(L, \boldsymbol{m}\right) \\ w_j &= \sum_k v_{j,k} \qquad \text{where } v_{j,k} \sim \mathrm{Multinomial}\left(c_k, \boldsymbol{\theta}_{\cdot,k}\right) \ . \end{aligned}$$

The joint likelihood, $p(\boldsymbol{V}, \boldsymbol{m} \,|\, \boldsymbol{\alpha}, \boldsymbol{\Theta}, K, \mathrm{DM})$, thus becomes, after some rearrangement

$$L!\, \Gamma\left(\sum_k \alpha_k\right) \prod_k \frac{m_k^{c_k+\alpha_k-1}}{\Gamma(\alpha_k)} \prod_{j,k} \frac{\theta_{j,k}^{v_{j,k}}}{v_{j,k}!} \ . \tag{11}$$

Again, $\boldsymbol{m}$ can be marginalised out yielding

$$L!\, \frac{\Gamma\left(\sum_k \alpha_k\right)}{\Gamma\left(L + \sum_k \alpha_k\right)} \prod_k \frac{\Gamma(c_k+\alpha_k)}{\Gamma(\alpha_k)} \prod_{j,k} \frac{\theta_{j,k}^{v_{j,k}}}{v_{j,k}!} \ . \tag{12}$$

7 Algorithms

In developing an algorithm, the standard approach is to match an optimization algorithm to the functional form of the likelihood. When using Bayesian or some other statistical methodology, this basic approach is usually a first step, or perhaps an inner loop for some more sophisticated computational statistics.

The likelihoods do not yield easily to standard EM analysis. To see this, consider the forms of the likelihood for a single document for the GP model, and consider the probability for a latent variable $\boldsymbol{z}$ given the observed data $\boldsymbol{w}$, $p(\boldsymbol{z} \,|\, \boldsymbol{w}, \boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\Theta}, K, \mathrm{GP})$. For EM analysis, one needs to be able to compute $\mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z} | \boldsymbol{w}, \boldsymbol{\Theta}, \ldots)} [\log p(\boldsymbol{w}, \boldsymbol{z} \,|\, \boldsymbol{\Theta}, \ldots)]$. There are three different forms of the likelihood seen so far depending on which latent variables $\boldsymbol{z}$ are kept on the left-hand side of the probability:

$p(\boldsymbol{w}, \boldsymbol{l} \,|\, \boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\Theta}, K, \mathbf{GP})$: from Equation (2) has the term $\left(\sum_k l_k \theta_{j,k}\right)^{w_j}$, which means there is no known simple posterior distribution for $\boldsymbol{l}$ given $\boldsymbol{w}$.

$p(\boldsymbol{w}, \boldsymbol{l}, \boldsymbol{V} \,|\, \boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\Theta}, K, \mathbf{GP})$: from Equation (6) has the term $l_k^{c_k + \alpha_k - 1}$ which links the two latent variables $\boldsymbol{l}$ and $\boldsymbol{V}$, and prevents a simple evaluation of $\mathbb{E}_{\boldsymbol{l}, \boldsymbol{V}}[v_{j,k}]$ as required for the expected log probability.

$p(\boldsymbol{w}, \boldsymbol{V} \,|\, \boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\Theta}, K, \mathbf{GP})$: from Equation (7) has the term $\Gamma(c_k + \alpha_k)$ (where $c_k = \sum_j v_j, k)$, which means there is no known simple posterior distribution for $\boldsymbol{V}$ given $\boldsymbol{w}$.

Now one could always produce an EM-like algorithm by separately updating $\boldsymbol{l}$ and $\boldsymbol{V}$ in turn according to some mean formula, but the guarantee of convergence of $\boldsymbol{\Theta}$ to a maximum posterior or likelihood value will not apply. In this spirit earlier authors point out that EM-like principles apply and use EM terminology since EM methods would apply if $\boldsymbol{l}$ was observed[8]. For the exponential family, which this problem is in, the variational approximation algorithm with Kullback-Leibler divergence corresponds to an extension of the EM algorithm [GB00, Bun02]. This variational approach is covered below.

Algorithms for this problem follow some general approaches in the statistical computing community. Three basic approaches are presented here: a variational approximation, Gibbs sampling, and Rao-Blackwellised Gibbs sampling. A maximum likelihood algorithm is not presented because it can be viewed as a simplification of the algorithms here.

7.1 Variational Approximation with Kullback-Leibler Divergence

This approximate method was first applied to the sequential variant of the Dirichlet-multinomial version of the problem by [BNJ03]. A fuller treatment of these variational methods for the exponential family is given in [GB00, Bun02].

[8] The likelihood $p(\boldsymbol{w}, \boldsymbol{V} \,|\, \boldsymbol{l}, \boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\Theta}, K, \mathrm{GP})$ can be treated with EM methods using the latent variable $\boldsymbol{V}$ and leaving $\boldsymbol{l}$ as if it was observed.

In this approach a factored posterior approximation is made for the latent variables:

$$p(\boldsymbol{l}, \boldsymbol{V} \,|\, \boldsymbol{w}, \boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\Theta}, K, \mathrm{GP}) \;\approx\; q(\boldsymbol{l}, \boldsymbol{V}) \;=\; q_{\boldsymbol{l}}(\boldsymbol{l}) q_{\boldsymbol{V}}(\boldsymbol{V})$$

and this approximation is used to find expectations as part of an optimization step. The EM algorithm results if an equality holds. The functional form of the approximation can be derived by inspection of the recursive functional forms (see [Bun02] Equation (4)):

$$\begin{aligned} q_{\boldsymbol{l}}(\boldsymbol{l}) &\propto \exp\left(\mathbb{E}_{\boldsymbol{V} \sim q_{\boldsymbol{V}}(\boldsymbol{V})}\left[\log p\left(\boldsymbol{l}, \boldsymbol{V}, \boldsymbol{w} \,|\, \boldsymbol{\Theta}, \boldsymbol{\alpha}, \boldsymbol{\beta}, K\right)\right]\right) \\ q_{\boldsymbol{V}}(\boldsymbol{V}) &\propto \exp\left(\mathbb{E}_{\boldsymbol{l} \sim q_{\boldsymbol{l}}(\boldsymbol{l})}\left[\log p\left(\boldsymbol{l}, \mathbf{v}, \boldsymbol{w} \,|\, \boldsymbol{\Theta}, \boldsymbol{\alpha}, \boldsymbol{\beta}, K\right)\right]\right) \ . \end{aligned} \tag{13}$$

An important computation used during convergence in this approach is a lower bound on the individual document log probabilities. This naturally falls out during computation (see [Bun02] Equation (6)). Using the approximation $q(\boldsymbol{l}, \boldsymbol{V})$ defined by the above proportions, the bound is given by

$$\begin{aligned} &\log p\left(\boldsymbol{w} \,|\, \boldsymbol{\Theta}, \boldsymbol{\alpha}, \boldsymbol{\beta}, K\right) \\ &\geq \quad \mathbb{E}_{\boldsymbol{l}, \boldsymbol{V} \sim q(\boldsymbol{l}, \boldsymbol{V})}\left[\log\, p\left(\boldsymbol{l}, \boldsymbol{V}, \boldsymbol{w} \,|\, \boldsymbol{\Theta}, \boldsymbol{\alpha}, \boldsymbol{\beta}, K\right)\right] + I(q_{\boldsymbol{l}}(\boldsymbol{l})) + I(q_{\boldsymbol{V}}(\boldsymbol{V})) \ . \end{aligned}$$

The variational approximation applies to the Gamma-Poisson version and the Dirichlet-multinomial version.

For the Gamma-Poisson Model: Looking at the recursive functionals of Equation (13) and the likelihood of Equation (6), it follows that $q_{\boldsymbol{l}}()$ must be K independent Gammas one for each component, and $q_{\boldsymbol{V}}()$ must be J independent multinomials, one for each word. The most general case for the approximation $q()$ is thus

$$\begin{aligned} l_k &\sim \mathrm{Gamma}(a_k, b_k) \\ \{v_{j,k} : k = 1, \ldots, K\} &\sim \mathrm{Multinomial}(w_j, \{n_{j,k} : k = 1, \ldots, K\}) \ , \end{aligned}$$

which uses approximation parameters (a_k, b_k) for each Gamma and and $\boldsymbol{n}_{\cdot,k}$ (normalised as $\sum_k n_{j,k} = 1$) for each multinomial. These parameters form two vectors $\boldsymbol{a}, \boldsymbol{b}$ and a matrix $\boldsymbol{N}$ respectively. The approximate posterior takes the form $q_{\boldsymbol{l}}(\boldsymbol{l} \,|\, \boldsymbol{a}, \boldsymbol{b}) q_{\boldsymbol{V}}(\boldsymbol{V} \,|\, \boldsymbol{N})$.

Using these approximating distributions, and again looking at the recursive functionals of Equation (13), one can extract the rewrite rules for the parameters:

$$\begin{aligned} n_{j,k} &= \frac{1}{Z_j} \theta_{j,k} \exp\left(\mathbb{E}\left[\log\, l_k\right]\right) \ , \\ a_k &= \alpha_k + \sum_j w_j n_{j,k} \ , \\ b_k &= 1 + \beta_k \ , \\ \textbf{where } \mathbb{E}\left[\log\, l_k\right] &\equiv \mathbb{E}_{l_k \sim p(l_k \,|\, a_k . b_k)}\left[\log\, l_k\right] \;=\; \Psi_0(a_k) - \log b_k \ , \\ Z_j &\equiv \sum_k \theta_{j,k} \exp\left(\mathbb{E}\left[\log\, l_k\right]\right) \ . \end{aligned} \tag{14}$$

Here, $\Psi_0()$ is the digamma function, defined as $\frac{\mathrm{d}\ln\Gamma(x)}{\mathrm{d}x}$ and available in most scientific libraries. These equations form the first step of each major cycle, and are performed on each document.

The second step is to re-estimate the model parameters $\boldsymbol{\Theta}$ using the posterior approximation by maximising the expectation of the log of the full posterior probability

$$\mathbb{E}_{\boldsymbol{l},\boldsymbol{V}\sim q_l(\boldsymbol{l})q_{\boldsymbol{V}}(\boldsymbol{V})}\left[\log p\left(\boldsymbol{l},\boldsymbol{V},\boldsymbol{w},\boldsymbol{\Theta}\,|\,\boldsymbol{\alpha},\boldsymbol{\beta},K\right)\right] \ .$$

This incorporates Equation (6) for each document, and a prior for each k-th column of $\boldsymbol{\Theta}$ of Dirichlet$_J(\boldsymbol{\gamma})$ (the last model item in Section 4.1). Denote the intermediate variables $n_{j,k}$ for the i-th document by adding a (i) subscript, as $n_{j,k,(i)}$, and likewise for $w_{j,(i)}$. All these log probability formulas yield linear terms in $\theta_{j,k}$, thus with the normalising constraints for $\boldsymbol{\Theta}$ one gets

$$\theta_{j,k} \ \propto \ \sum_i w_{j,(i)} n_{j,k,(i)} + \gamma_j \ . \tag{15}$$

The lower bound on the log probability of Equation (14), after some simplification and use of the rewrites of Equation (14), becomes

$$\log\frac{1}{\prod_j w_j!} - \sum_k \log\frac{\Gamma(\alpha_k) b_k^{a_k}}{\Gamma(a_k)\beta_k^{\alpha_k}} + \sum_k (\alpha_k - a_k)\mathbb{E}\left[\log l_k\right] + \sum_j w_j \log Z_j \ . \tag{16}$$

The variational approximation algorithm for the Gamma-Poisson version is summarised in Figure 4. An equivalent algorithm is produced if words are presented sequentially instead of being bagged.

```
1. Initialise $\boldsymbol{a}$ for each document. The uniform initialisation would be $a_k = \left(\sum_k \alpha_k + L\right)/K$. Note $\boldsymbol{N}$ is not stored.
2. Do for each document:
    (a) Using Equations (14), recompute $\boldsymbol{N}$ and update $\boldsymbol{a}$ in place.
    (b) Concurrently, compute the log-probability bound of Equation (16), and add to a running total.
    (c) Concurrently, maintain the sufficient statistics for $\boldsymbol{\Theta}$, the total $\sum_i w_{j,(i)} n_{j,k,(i)}$ for each $j,k$ over documents.
    (d) Store $\boldsymbol{a}$ for the next cycle and discard $\boldsymbol{N}$.
3. Update $\boldsymbol{\Theta}$ using Equation (15), normalising appropriately.
4. Report the total log-probability bound, and repeat, starting at Step 2.
```

Fig. 4. K-L Variational Algorithm for Gamma-Poisson

Complexity: Because Step 2(a) only uses words appearing in a document, the full Step 2 is $O(SK)$ in time complexity where S is the number of words in the full collection. Step 3 is $O(JK)$ in time complexity. Space complexity is $O(IK)$

to store the intermediate parameters $\boldsymbol{a}$ for each document, and the $O(2JK)$ to store $\boldsymbol{\Theta}$ and its statistics. In implementation, Step 2 for each document is often quite slow, and thus both $\boldsymbol{a}$ and the document word data can be stored on disk and streamed, thus the main memory complexity is $O(2JK)$ since the $O(S)$ and $O(IK)$ terms are on disk. If documents are very small (e.g., $S/I \ll K$, for instance "documents" are sentences or phrases), then this does not apply.

Correspondence with NMF: A precursor to the GaP model is non-negative matrix factorisation (NMF) [LS99], which is based on the matrix approximation paradigm using Kullback-Leibler divergence. The algorithm itself, converted to the notation used here, is as follows

$$l_{k,(i)} \longleftarrow l_{k,(i)} \sum_j \frac{\theta_{j,k}}{\sum_j \theta_{j,k}} \frac{w_{j,(i)}}{\sum_k \theta_{j,k} l_{k,(i)}} \qquad \theta_{j,k} \longleftarrow \theta_{j,k} \sum_i \frac{l_{k,(i)}}{\sum_i l_{k,(i)}} \frac{w_{j,(i)}}{\sum_k \theta_{j,k} l_{k,(i)}}$$

Notice that the solution is indeterminate up to a factor ψ_k. Multiply $l_{k,(i)}$ by ψ_k and divide $\theta_{j,k}$ by ψ_k and the solution still holds. Thus, without loss of generality, let $\theta_{j,k}$ be normalised on j, so that $\sum_j \theta_{j,k} = 1$.

Lemma 2. *The NMF equations above, where $\boldsymbol{\Theta}$ is returned normalised, occur at a maxima w.r.t. $\boldsymbol{\Theta}$ and $\boldsymbol{l}$ for the Gamma-Poisson likelihood $\prod_i p(\boldsymbol{w}_{(i)} \mid \boldsymbol{\Theta}, \boldsymbol{l}_{(i)}, \boldsymbol{\alpha} = 0, \boldsymbol{\beta} = 0, K, GP)$.*

Proof. To see this, the following will be proven. Take a solution to the NMF equations, and divide $\theta_{j,k}$ by a factor $\psi_k = \sum_j \theta_{j,k}$, and multiply $l_{k,(i)}$ by the same factor. This is equivalent to a solution for the following rewrite rules

$$l_{k,(i)} \longleftarrow l_{k,(i)} \sum_j \theta_{j,k} \frac{w_{j,(i)}}{\sum_k \theta_{j,k} l_{k,(i)}} \qquad \theta_{j,k} \propto \theta_{j,k} \sum_i l_{k,(i)} \frac{w_{j,(i)}}{\sum_k \theta_{j,k} l_{k,(i)}}$$

where $\theta_{j,k}$ is kept normalised on j. These equations hold at a maxima to the likelihood $\prod_i p(\boldsymbol{w}_{(i)} \mid \boldsymbol{\Theta}, \boldsymbol{l}_{(i)}, \boldsymbol{\alpha} = 0, \boldsymbol{\beta} = 0, K, \mathrm{GP})$. The left equation corresponds to a maxima w.r.t. $\boldsymbol{l}_{(i)}$ (note the Hessian for this is easily shown to be negative indefinite), and the right is the EM equations for the likelihood. w.r.t. $\boldsymbol{\Theta}$.

To show equivalence of the above and the NMF equations, first prove the forward direction. Take the scaled solution to NMF. The NMF equation for $l_{k,(i)}$ is equivalent to the equation for $l_{k,(i)}$ in the lemma. Take the NMF equation for $\theta_{j,k}$ and separately normalise both sides. The $\sum_i l_{k,(i)}$ term drops out and one is left with the equation for $\theta_{j,k}$ in the lemma. Now prove the backward direction. It is sufficient to show that the NMF equations hold for the solution to the rewrite rules in the lemma, since $\theta_{j,k}$ is already normalised. The NMF equation for $l_{k,(i)}$ clearly holds. Assuming the rewrite rules in the lemma hold, then

$$\begin{aligned}
\theta_{j,k} &= \frac{\theta_{j,k} \sum_i \left(l_{k,(i)} w_{j,(i)} / \sum_k \theta_{j,k} l_{k,(i)} \right)}{\sum_j \theta_{j,k} \sum_i \left(l_{k,(i)} w_{j,(i)} / \sum_k \theta_{j,k} l_{k,(i)} \right)} \\
&= \frac{\theta_{j,k} \sum_i \left(l_{k,(i)} w_{j,(i)} / \sum_k \theta_{j,k} l_{k,(i)} \right)}{\sum_i l_{k,(i)} \sum_j \left(\theta_{j,k} w_{j,(i)} / \sum_k \theta_{j,k} l_{k,(i)} \right)} && \text{(reorder sum)} \\
&= \frac{\theta_{j,k} \sum_i \left(l_{k,(i)} w_{j,(i)} / \sum_k \theta_{j,k} l_{k,(i)} \right)}{\sum_i l_{k,(i)}} && \text{(apply first rewrite rule)}
\end{aligned}$$

Thus the second equation for NMF holds.

Note, including a latent variable such as $\boldsymbol{l}$ in the likelihood (and not dealing with it using EM methods) does not achieve a correct maximum likelihood solution for the expression $\prod_i p(\boldsymbol{w}_{(i)} \,|\, \boldsymbol{\Theta}, \boldsymbol{\alpha} = 0, \boldsymbol{\beta} = 0, K, \mathrm{GP})$. In practice, this is a common approximate method for handling latent variable problems, and can lead more readily to over-fitting.

For the Dirichlet-Multinomial Model: The variational approximation takes a related form. The approximate posterior is given by:

$$\begin{aligned} \boldsymbol{m} &\sim \mathrm{Dirichlet}(\boldsymbol{a}) \\ \{v_{j,k} : k = 1, \ldots, K\} &\sim \mathrm{multinomial}(w_j, \{n_{j,k} : k = 1, \ldots, K\}) \end{aligned}$$

This yields the same style update equations as Equations (14) except that $\beta_k = 1$

$$\begin{aligned} n_{j,k} &= \frac{1}{Z_j} \theta_{j,k} \exp\left(\mathbb{E}\left[\log\, m_k\right]\right) \ , \qquad (17) \\ a_k &= \alpha_k + \sum_j w_j n_{j,k} \ , \\ \textbf{where } \mathbb{E}\left[\log\, m_k\right] \equiv \mathbb{E}_{m_k \sim p(m_k \,|\, \boldsymbol{a})}\left[\log\, m_k\right] &= \Psi_0(a_k) - \Psi_0\left(\sum_k a_k\right) \ , \\ Z_j &\equiv \sum_k \theta_{j,k} \exp\left(\mathbb{E}\left[\log\, m_k\right]\right) \ . \end{aligned}$$

Equation (15) is also the same. The lower bound on the individual document log probabilities, $\log p\left(\boldsymbol{w} \,|\, \boldsymbol{\Theta}, \boldsymbol{\alpha}, K, \mathrm{DM}\right)$ now takes the form

$$\log\left(C^L_{\boldsymbol{w}}\right) - \log \frac{\Gamma\left(\sum_k a_k\right) \prod_k \Gamma(\alpha_k)}{\Gamma\left(\sum_k \alpha_k\right) \prod_k \Gamma(a_k)} + \sum_k (\alpha_k - a_k) \mathbb{E}\left[\log\, m_k\right] + \sum_j w_j \log Z_j \ . \qquad (18)$$

The correspondence with Equation (16) is readily seen.

The algorithm for Dirichlet-multinomial version is related to that in Figure 4. Equations (17) replace Equations (14), Equation (18) replaces Equation (16), and the initialisation for a_k should be 0.5, a Jeffreys prior.

7.2 Direct Gibbs Sampling

There are two styles of Gibbs sampling that apply to DCA. The first is a basic Gibbs sampling first proposed by Pritchard, Stephens and Donnelly [PSD00]. Gibbs sampling is a conceptually simple method. Each unobserved variable in the problem is resampled in turn according to its conditional distribution. We compute its posterior distribution conditioned on all other variables, and then sample a new value for the variable using the posterior. For instance, an ordering we might use in this problem is: $\boldsymbol{l}_{(1)}, \boldsymbol{V}_{(1)}$, $\boldsymbol{l}_{(2)}, \boldsymbol{V}_{(2)}$, $\ldots$, $\boldsymbol{l}_{(I)}, \boldsymbol{V}_{(I)}$, $\boldsymbol{\Theta}$. All the low level sampling in this section use well known distributions such as gamma or multinomial, and are available in standard scientific libraries.

To develop this approach for the Gamma-Poisson, look at the full posterior, which is a product of individual document likelihoods with the prior for $\boldsymbol{\Theta}$ from the last model item in Section 4.1. The constant terms have been dropped.

$$\prod_i \left(\prod_k \frac{\beta_k^{\alpha_k} l_{k,(i)}^{c_{k,(i)}+\alpha_k-1} \exp\{-(\beta_k+1) l_{k,(i)}\}}{\Gamma(\alpha_k)} \prod_{j,k} \frac{\theta_{j,k}^{v_{j,k,(i)}}}{v_{j,k,(i)}!} \right) \prod_{j,k} \theta_{j,k}^{\gamma_j} \tag{19}$$

Each of the conditional distributions used in the Gibbs sampling are proportional to this. The first conditional distribution is $p(\boldsymbol{l}_{(i)} \,|\, \boldsymbol{V}_{(i)}, \boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\Theta}, K, \text{GP})$. From this, isolating the terms just in $\boldsymbol{l}_{(i)}$, we see that each $l_{k,(i)}$ is conditionally gamma distributed. Likewise, each $\boldsymbol{v}_{\cdot,k,(i)}$ is multinomial distributed given $\boldsymbol{l}_{(i)}$ and $\boldsymbol{\Theta}$, and each $\boldsymbol{\theta}_{\cdot,k}$ is Dirichlet distributed given all the $\boldsymbol{l}_{(i)}$ and $\boldsymbol{V}_{(i)}$ for each i. The other models are similar. An additional effort is required to arrange the parameters and sequencing for efficient use of memory.

The major differentiator for Gibbs sampling is the resampling of the latent component vector $\boldsymbol{l}$. The sampling schemes used for each version are given in Table 3. Some care is required with the conditional Gamma-Poisson. When $c_k = 0$, the sampling for l_k needs to decide whether to use the zero case or the non-zero case. This uses Equation (9) to make the decision, and then resorts to Equation (8) if it is non-zero.

Model	Sampling
GP	$l_k \sim \text{Gamma}(c_k + \alpha_k, 1 + \beta_k)$.
CGP	If $c_k = 0$, then Conditional Gamma-Poisson with rate $\frac{p_k(1+\beta_k)^{\alpha_k}}{(1-p_k)\beta_k^{\alpha_k} + p_k(1+\beta_k)^{\alpha_k}}$ and $\text{Gamma}(\alpha_k, 1+\beta_k)$. If $c_k \neq 0$, revert to the above Gamma-Poisson case.
DM	$\boldsymbol{m} \sim \text{Dirichlet}(\{c_k + \alpha_k \,:\, k\})$.

Table 3. Sampling components for direct Gibbs on a single document

The direct Gibbs algorithm for the general case is given in Figure 5. This Gibbs scheme turns out to correspond to the variational approximation, excepting that sampling is done instead of maximisation or expectation.

The log probability of the words $\boldsymbol{w}$ can also accumulated in step 1(c). While they are in terms of the latent variables, they still represent a reasonably unbiased estimate of the likelihoods such as $p\left(\boldsymbol{w}_{(1)}, \ldots, \boldsymbol{w}_{(I)} \,|\, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\Theta}, K, \text{GP}\right)$.

7.3 Rao-Blackwellised Gibbs Sampling

Rao-Blackwellisation of Gibbs sampling [CR96] combines closed form updates of variables with Gibbs sampling. It does so by a process called marginalisation or

1. For each document i, retrieve the last $\boldsymbol{c}_{(i)}$ from store, then
   (a) Sample the latent component variables $\boldsymbol{l}_{(i)}$ (or its normalised counterpart $\boldsymbol{m}_{(i)}$) as per Table 3.
   (b) For each word j in the document with positive count $w_{j,(i)}$, the component counts vector, from Equation (5) and Equation (10),
   $$\{v_{j,k,(i)} \,:\, k = 1, \ldots, K\} \;\sim\; \text{Multinomial}\left(w_{j,(i)}, \left\{\frac{l_{k,(i)}\theta_{j,k}}{\sum_k l_{k,(i)}\theta_{j,k}} \,:\, k\right\}\right) \; .$$
   Alternatively, if the sequence-of-components version is to be used, the component for each word can be sampled in turn using the corresponding Bernoulli distribution.
   (c) Concurrently, accumulate the log-probability $p\left(\boldsymbol{w}_{(i)} \,|\, \boldsymbol{l}_{(i)}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\Theta}, K, \text{GP}\right)$, $p\left(\boldsymbol{w}_{(i)} \,|\, \boldsymbol{l}_{(i)}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\rho}, \boldsymbol{\Theta}, K, \text{CGP}\right)$, or $p\left(\boldsymbol{w}_{(i)} \,|\, \boldsymbol{m}_{(i)}, L_{(i)}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\Theta}, K, \text{DM}\right)$.
   (d) Concurrently, maintain the sufficient statistics for $\boldsymbol{\Theta}$, the total $\sum_i v_{j,k,(i)}$ for each j, k over documents.
   (e) Store $\boldsymbol{c}_{(i)}$ for the next cycle and discard $\boldsymbol{V}_{(i)}$.
2. Using a Dirichlet prior for rows of $\boldsymbol{\Theta}$, and having accumulated all the counts $\boldsymbol{V}_{(i)}$ for each document in sufficient statistics for $\boldsymbol{\Theta}$, then its posterior has rows that are Dirichlet. Sample.
3. Report the total log-probability, and report.

Fig. 5. One Major Cycle of Gibbs Algorithm for DCA

variable elimination. When feasible, it can lead to significant improvements, the general case for DCA. Griffiths and Steyvers [GS04] introduced this algorithm for LDA, and it easily extends to the Gamma-Poisson model and its conditional variant with little change to the sampling routines.

When using this approach, the first step is to consider the full posterior probability and see which variables can be marginalised out without introducing computational complexity in the sampling. For the GP model, look at the posterior given in Equation (19). Equations (7) shows that the $\boldsymbol{l}_{(i)}$'s can be marginalised out. Likewise, $\boldsymbol{\Theta}$ can be marginalised out because it is an instance of a Dirichlet. This yields a Gamma-Poisson posterior $p\left(\boldsymbol{V}_{(1)}, \ldots, \boldsymbol{V}_{(I)} \,|\, \boldsymbol{\alpha}, \boldsymbol{\beta}, K, \text{GP}\right)$, with constants dropped:

$$\prod_i \left(\prod_k \frac{\Gamma(c_{k,(i)} + \alpha_k)}{(1+\beta_k)^{c_{k,(i)}+\alpha_k}} \prod_{j,k} \frac{1}{v_{j,k,(i)}!} \right) \prod_k \frac{\prod_j \Gamma\left(\gamma_j + \sum_i v_{j,k,(i)}\right)}{\Gamma\left(\sum_j \gamma_j + \sum_i c_{k,(i)}\right)} \tag{20}$$

Below it is shown that a short sampling routine can be based on this.

A similar formula applies in the conditional GP case using Equation (9) for the marginalisation of $\boldsymbol{l}_{(i)}$'s. The first term with $\prod_k$ in Equation (20) becomes

$$(1-\rho_k)\frac{\Gamma(c_{k,(i)} + \alpha_k)}{\Gamma(\alpha_k)} \frac{\beta_k^{\alpha_k}}{(1+\beta_k)^{c_{k,(i)}+\alpha_k}} + \rho_k 1_{c_{k,(i)}=0} \; .$$

Likewise a similar formula applies in the Dirichlet-multinomial version using Equation (12):

$$\prod_i \left(\prod_k \Gamma(c_{k,(i)} + \alpha_k) \prod_{jk} \frac{1}{v_{j,k,(i)}!} \right) \prod_k \frac{\prod_j \Gamma\left(\gamma_j + \sum_i v_{j,k,(i)}\right)}{\Gamma\left(\sum_j \gamma_j + \sum_i c_{k,(i)}\right)} \tag{21}$$

Here a term of the form $\Gamma\left(\sum_k (c_{k,(i)} + \alpha_k)\right)$ drops out because $\sum_k c_{k,(i)} = L_{(i)}$ is known and thus constant.

Now the posterior distributions have been marginalised for each of the three models, GP, CGP and DM, a Gibbs sampling scheme needs to be developed. Each set $\{v_{j,k,(i)} : k \in 1, \ldots, K\}$ sums to $w_{j,(i)}$, moreover the forms of the functions in Equations (20) and (21) are quite nasty. A way out of this mess is to convert the scheme from a bag of words model, implicit in the use of $\boldsymbol{V}_{(i)}$ and $\boldsymbol{w}_{(i)}$, to a sequence of words model.

This proceeds as follows. Run along the $L_{(i)}$ words in a document and update the corresponding component assignment for each word. Component assignments for the i-th document are in a $L_{(i)}$-dimensional vector $\boldsymbol{k}_{(i)}$, where each entry takes a value from $1, \ldots, K$. Suppose the l-th word has word index j_l. In one step, change the counts $\{v_{j_l,k,(i)} : k \in 1, \ldots, K\}$ by one (one is increased and one is decreased) keeping the total $w_{j_l,(i)}$ constant. For instance, if a word is originally in component k_1 but updating by Gibbs sampling to k_2, then decrease $v_{j_l,k_1,(i)}$ by one and increase $v_{j_l,k_2,(i)}$ by one. Do this for $L_{(i)}$ words in the document, for each document. Thus at word l for the i-th document, we sample component assignment $k_{l,(i)}$ according to the posterior for $k_{l,(i)}$ with all other assignments fixed. This posterior is proportional to (the denominator is a convenient constant)

$$\frac{p\left(\boldsymbol{V} \mid \text{sequential}, \boldsymbol{\alpha}, \boldsymbol{\beta}, K, \text{GP}\right)\big|_{v_{j_l,k,(i)} \leftarrow v_{j_l,k,(i)} + 1_{k \neq k_l}}}{p\left(\boldsymbol{V} \mid \text{sequential}, \boldsymbol{\alpha}, \boldsymbol{\beta}, K, \text{GP}\right)\big|_{v_{j_l,k_l,(i)} \leftarrow v_{j_l,k_l,(i)} - 1}} ,$$

where the notation "sequential" is added to the right-hand side because the combinatoric terms $v_{j,k,(i)}!$ of Equation (20) need to be dropped. This formula simplifies dramatically because $\Gamma(x+1)/\Gamma(x) = x$.

Derived sampling schemes are given in Table 3. The (i) subscript is dropped and assumed for all counts, and $j = j_l$ is the word index for the word whose component index is being resampled. Since k_l is being sampled, a K dimensional probability vector is needed. The table gives the unnormalised form.

This Rao-Blackwellised Gibbs algorithm is given in Figure 6. As before, an approximately unbiased log probability can be recorded in Step 2(c). This requires a value for $\boldsymbol{\Theta}$. While the sufficient statistics could be used to supply the current mean estimate for $\boldsymbol{\Theta}$, this is not a true sampled quantity. An alternative method is to make a sample of $\boldsymbol{\Theta}$ in each major cycle and use this.

Implementation notes: Due to Rao-Blackwellisation, both the $\boldsymbol{l}_{(i)}$'s and $\boldsymbol{\Theta}$ are effectively re-estimated with each sampling step, instead of once after the

Model	Sampling Proportionality
GP	$\frac{\gamma_j + \sum_i v_{j,k}}{\sum_j \gamma_j + \sum_i c_k} \frac{c_k + \alpha_k}{1+\beta_k}$
CGP	When $c_k > 0$ use the proportionality of the GP case, and otherwise $\frac{\gamma_j + \sum_i v_{j,k}}{\sum_j \gamma_j + \sum_i c_k} \frac{\alpha_k}{1+\beta_k} \frac{(1-\rho_k)\beta_k^{\alpha_k}}{(1-\rho_k)\beta_k^{\alpha_k} + \rho_k(1+\beta_k)^{\alpha_k}}$
DM	$\frac{\gamma_j + \sum_i v_{j,k}}{\sum_j \gamma_j + \sum_i c_k}(c_k + \alpha_k).$

Table 4. Sampling $k_l = k$ given $j = j_l$ for Rao-Blackwellised Gibbs

1. Maintain the sufficient statistics for $\boldsymbol{\Theta}$, given by $\sum_i v_{j,k,(i)}$ for each j and k, and the sufficient statistics for the component proportions $\boldsymbol{l}_{(i)}/\boldsymbol{m}_{(i)}$ given by $\boldsymbol{c}_{(i)}$.
2. For each document i, retrieve the $L_{(i)}$ component assignments for each word then:
   (a) Recompute statistics for $\boldsymbol{l}_{(i)}/\boldsymbol{m}_{(i)}$ given by $c_{k,(i)} = \sum_j v_{j,k,(i)}$ for each k from the individual component assignment for each word.
   (b) For each word l with word index j_l and component assignment k_l in the document, resample the component assignment for this word according to the marginalised likelihoods in this section.
      i. First decrement $v_{j_l,k_l,(i)}$ and $c_{k_l,(i)}$ by one to remove the component assignment for the word.
      ii. Sample $k_l = k$ proportionally as in Table 4.
      iii. Increment $v_{j_l,k_l,(i)}$ and $c_{k_l,(i)}$.
   (c) Concurrently, record the log-probability such as $p\left(\boldsymbol{w}_{(i)} \,|\, \boldsymbol{V}_{(i)}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\Theta}, K, \mathrm{GP}\right)$ for the appropriate model.
   (d) Concurrently, update the sufficient statistics for $\boldsymbol{l}_{(i)}/\boldsymbol{m}_{(i)}$ and $\boldsymbol{\Theta}$.

Fig. 6. One Major Cycle of Rao-Blackwellised Gibbs Algorithm for DCA

full pass over documents. This is most effective during early stages, and explains the superiority of the method observed in practice. Moreover, it means only one storage slot for $\boldsymbol{\Theta}$ is needed (to store the sufficient statistics), whereas in direct Gibbs two slots are needed (current value plus the sufficient statistics). This represents a major saving in memory. Finally, the $\boldsymbol{l}_{(i)}$'s and $\boldsymbol{\Theta}$ can be sampled at any stage of this process (because their sufficient statistics make up the totals appearing in the formula), thus Gibbs estimates for them can be made as well during the MCMC process.

7.4 Historical notes

Some previous algorithms can now be placed into context.

NMF: Straight maximum likelihood, e.g. in [LS99], expressed in terms of Kullback-Leibler divergence minimization, where optimisation jointly applies to the latent variables (see Section 7.1).

PLSI: Annealed maximum likelihood [Hof99], best viewed in terms of its clustering precursor such as by [HB97],

Various Gibbs: Gibbs sampling on $\boldsymbol{V}_{(i)}$, $\boldsymbol{l}_{(i)}/\boldsymbol{m}_{(i)}$ and $\boldsymbol{\Theta}$ in turn using a full probability distribution by [PSD00], or Gibbs sampling on $\boldsymbol{V}_{(i)}$ alone (or equivalently, component assignments for words in the sequence of words representation) after marginalising out $\boldsymbol{l}_{(i)}/\boldsymbol{m}_{(i)}$ and $\boldsymbol{\Theta}$ by [GS04],

LDA: variational approximation with Kullback-Leibler divergence by [BNJ03], a significant introduction because of its speed.

Expectation propagation [ML02] requires $O(KS)$ latent variables stored, a prohibitive expense compared to the $O(S)$ or $O(KI)$ of other algorithms. Thus it has not been covered here.

7.5 Other Aspects for Estimation and Use

A number of other algorithms are needed to put these models into regular use.

Component parameters: The treatment so far has assumed the parameter vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are given. It is more usual to estimate these parameters with the rest of the estimation tasks as done by [BNJ03, Can04]. This is feasible because the parameters are shared across all the data, unlike the component vectors themselves.

Estimating the number of components K: The number of components K is usually a constant assumed a priori. But it may be helpful to treat as a fittable parameter or a random variable that adapts to the data. In popular terms, this could be used to find the "right" number of components, though in practice and theory such a thing might not exist. To obtain best-fitting K, we can employ cross-validation, or we assess the *evidence* (or marginal likelihood) for the model given a particular choice of K [CC95, BJ04]. In particular, evidence is the posterior probability of the data given the choice of K after all other parameters have been integrated out.

Use on new data: A typical use of the model requires performing inference related to a particular document. Suppose, for instance, one wished to estimate how well a snippet of text, a query, matches a document. Our document's components are summarised by the latent variables $\boldsymbol{m}$ (or $\boldsymbol{l}$). If the new query is represented by $\boldsymbol{q}$, then $p(\boldsymbol{q}|\boldsymbol{m}, \boldsymbol{\Theta}, K, \text{GP})$ is the matching quantity one would like ideally. Since $\boldsymbol{m}$ is unknown, we must average over it. Various methods have been proposed [ML02, BJ04].

Alternative components: Hierarchical components have been suggested [BJ04] as a way of organising an otherwise large flat component space. For instance, the Wikipedia with over half a million documents can easily support the discovery of several hundred components. Dirichlet processes have been developed as an alternative to the K-dimensional component priors in the Dirichlet-multinomial/discrete model [YYT05], although in implementation the effect is to use K-dimensional Dirichlets for a large K and delete low performing components.

8 Applications

This section briefly discusses two applications of the methods.

8.1 Voting Data

One type of political science data are the *roll calls.* There were 459 roll calls in the US Senate in the year 2003. For each of those, the vote of every senator was recorded in three ways: 'Yea', 'Nay' and 'Not Voting'. The outcome of the roll call can be positive (e.g., Bill Passed, Nomination Confirmed) corresponding to 'Yea', or negative (e.g., Resolution Rejected, Veto Sustained). Hence, the outcome of the vote can be interpreted as the 101st senator, by associating positive outcomes with 'Yea' and negative outcomes with 'Nay'.

Application of the Method: We can now map the roll call data to the DCA framework. For each senator X we form two 'words', where $w_{X,y}$ implies that X voted 'Yea', and $w_{X,n}$ implies that X voted 'Nay'. Each roll call can be interpreted as a document containing a single occurrence of some of the available words. The pair of words $w_{X,y}, w_{X,n}$ is then treated as a binomial, so the multivariate formulation of Section 4.4 is used. Priors for $\boldsymbol{\Theta}$ were Jeffreys priors, $\boldsymbol{\alpha}$ was (0.1,0.1,...,0.1), and regular Gibbs sampling was used.

Special-purpose models are normally used for interpreting roll call data in political science, and they often postulate a model of rational decision making. Each senator is modelled as a position or an *ideal point* in a continuous spatial model of preferences [CJR04]. For example, the first dimension often delineates the liberal-conservative preference, and the second region or social issues preference. The proximities between ideal points 'explain' the positive correlations between the senators' votes. The ideal points for each senator can be obtained either by optimization, for instance, with the optimal classification algorithm [Poo00], or through Bayesian modelling [CJR04].

Unlike the spatial models, the DCA interprets the correlations between votes through membership of the senators in similar blocs. Blocs correspond to latent component variables. Of course, we can speak only of the probability that a particular senator is a member of a particular bloc. The corresponding probability vector is normalized and thus assures that a senator is always a member of one bloc on the average. The outcome of the vote is also a member of several blocs, and we can interpret the membership as a measure of how influential a particular bloc is.

Our latent senator (bloc) can be seen as casting votes in each roll call. We model the behavior of such latent blocs across the roll calls, and record it: it has a behavior of its own. In turn, we also model the membership of each senator to a particular bloc, which is assumed to be constant across all the blocs.

A related family of approaches is based on modelling relations or networks using blocks or groups. There, a roll call would be described by one network, individual senators would be nodes in that network, and a pair of nodes is connected if the two senators agreed. Discrete latent variables try to explain the existence of links between entities in terms of senators' membership to blocks, e.g., [HLL83, SN97].

Several authors prefer the block-model approach to modelling roll call data [WMM05]. The membership of senators to the same block along with a high probability for within-block agreements will explain the agreements between senators. While a bloc can be seen as having an opinion about each issue, a block does not (at least not explicitly). The authors also extended this model to 'topics', where the membership of senator to a particular block depends on the topic of the issue; namely, the agreement between senators depends on what is being discussed. The topic is also associated with the words that appear in the description of an issue.

Visualization: We can analyze two aspects of the DCA model as applied to the roll call data: we can examine the membership of senators in blocs, and we can examine the actions of blocs for individual issues. The approach to visualization is very similar, as we are visualizing a set of probability vectors. We can use the gray scale to mirror the probabilities ranging from 0 (white) to 1 (black).

As yet, we have not mentioned the choice of K - the number of blocs. Although the number of blocs can be a nuisance variable, such a model is distinctly more difficult to show than one for a fixed K. We obtain the following negative logarithms to the base 2 of the model's likelihood for $K = 4, 5, 6, 7, 10$: 9448.6406, 9245.8770, 9283.1475, 9277.0723, 9346.6973. We see that $K = 5$ is overwhelmingly selected over all others, with $K = 4$ being far worse. This means that with our model, we best describe the roll call votes with the existence of five blocs. Fewer blocs do not capture the nuances as well, while more blocs would not yield reliable probability estimates given such an amount of data. Still, those models are also valid to some extent. It is just that for a single visualization we pick the best individual one of them.

We will now illustrate the membership of senators in blocs. Each senator is represented with a vertical bar of 5 squares that indicate his or her membership in blocs. We have arranged the senators from left to right using the binary PCA approach of [deL03]. This ordering attempts to sort senators from the most extreme to the most moderate and to the most extreme again. Figure 7 shows the Democrat senators and Figure 8 the Republicans.

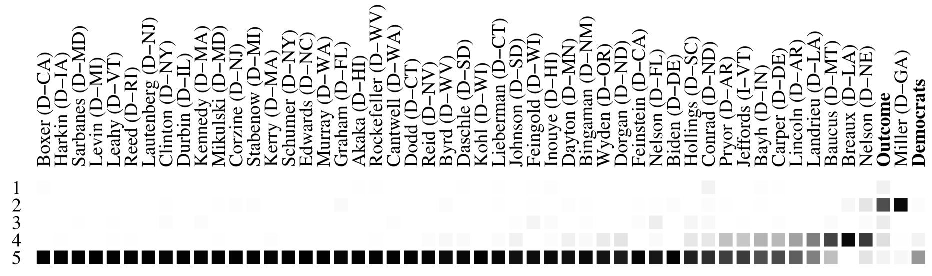


Fig. 7. Component membership for Democrats

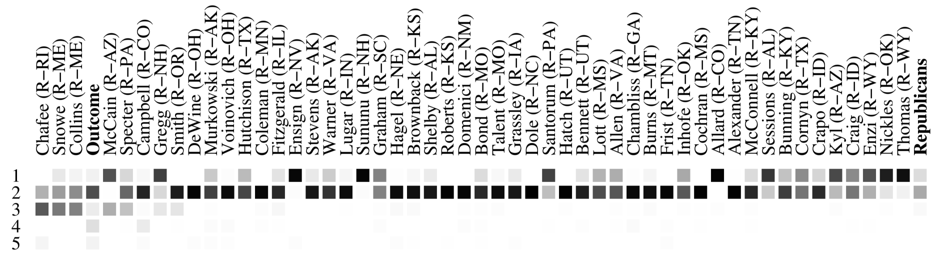


Fig. 8. Component membership for Republicans

We can observe that component 5 is the Democrat majority. It is the strongest overall component, yet quite uninfluential about the outcome. Component 4 are the moderate Democrats, and they seem distinctly more influential than the Democrats of the majority. Component 3 is a small group of Republican moderates. Component 2 is the Republican majority, the most influential bloc. Component 1 is the Republican minority, not very influential. Component 1 tends to be slightly more extreme than component 2 on the average, but the two components clearly cannot be unambiguously sorted.

9 Classification Experiments

DCA is not trying to capture those aspects of the text that are relevant for distinguishing one class of documents from another one. Assume our classification task is to distinguish newspaper articles on politics from all others. For DCA, modelling the distribution of a word such as 'the' is equally or more important than modelling the distribution of a highly pertinent word such as 'tsunami' in classifying news reports. Of course, this is only a single example of sub-optimality. Nevertheless, if there are several methods of reducing the dimensionality of text that all disregard the classification problem at hand, we can still compare them with respect to classification performance.

We used MPCA and tested its use in its role as a feature construction tool, a common use for PCA and ICA, and as a classification tool. For this, we used the 20 newsgroups collection described previously as well as the Reuters-21578 collection[9]. We employed the SVM^{light} V5.0 [Joa99] classifier with default settings. For classification, we added the class as a distinct multinomial (cf. Section 4.4) for the training data and left it empty for the test data, and then predicted the class value. Note that for performance and accuracy, SVM [Joa98] is a clear winner [LYRL04]. It is interesting to see how MPCA compares.

Each component can be seen as generating a number of words in each document. This number of component-generated words plays the same role in classification as does the number of lexemes in the document in ordinary classification. In both cases, we employed the `tf*idf` transformed word and component-generated word counts as feature values. Since SVM works with sparse data matrices, we assumed that a component is not present in a document if the number of words that a component would have generated is less than 0.01. The components alone do not yield a classification performance that would be competitive with SVM, as the label has no distinguished role in the fitting. However, we may add these component-words in the default bag of words, hoping that the conjunctions of words inherent to each component will help improve the classification performance.

For the Reuters collection, we used the ModApte split. For each of the 6 most frequent categories, we performed binary classification. Further results are disclosed in Table 2[10]. No major change was observed by adding 50 components to the original set of words. By performing classification on components alone, the results were inferior, even with a large number of components. In fact, with 300 components, the results were worse than with 200 components, probably because of over-fitting. Therefore, regardless of the number of components, the SVM performance with words cannot be reproduced by component-generated words in this collection. Classifying newsgroup articles into 20 categories proved more successful. We employed two replications of 5-fold cross validation, and we achieved the classification accuracy of 90.7% with 50 additional MPCA components, and 87.1% with SVM alone. Comparing the two confusion matrices, the most frequent mistakes caused by SVM+MPCA beyond those of SVM alone were predicting talk.politics.misc as sci.crypt (26 errors) and talk.religion.misc predicted as sci.electron (25 errors). On the other hand, the components helped better identify alt.atheism and talk.politics.misc, which were misclassified as talk.religion.misc (259 fewer errors) earlier. Also, talk.politics.misc and talk.religion.misc were not misclassified as talk.politics.gun (98 fewer errors). These 50 components were not very successful alone, resulting in 18.5% classification accuracy. By increasing the number of components to 100 and 300, the classification accuracy gradually increases to 25.0% and 34.3%. Therefore, many components are needed for general-purpose classification.

[9] The Reuters-21578, Distribution 1.0 test collection is available from David D. Lewis' professional home page, currently: http://www.research.att.com/∼lewis

[10] The numbers are percentages, and 'P/R' indicates precision/recall.

Table 5. SVM Classification Results

	SVM		SVM+MPCA	
CAT	ACC.	P/R	ACC.	P/R
earn	98.58	98.5/97.1	98.45	98.2/97.1
acq	95.54	97.2/81.9	95.60	97.2/82.2
moneyfx	96.79	79.2/55.3	96.73	77.5/55.9
grain	98.94	94.5/81.2	98.70	95.7/74.5
crude	97.91	89.0/72.5	97.82	88.7/70.9
trade	98.24	79.2/68.1	98.36	81.0/69.8

	MPCA (50 comp.)		MPCA (200 comp.)	
CAT	ACC.	P/R	ACC.	P/R
earn	96.94	96.1/94.6	97.06	96.3/94.8
acq	92.63	93.6/71.1	92.33	95.3/68.2
moneyfx	95.48	67.0/33.0	96.61	76.0/54.7
grain	96.21	67.1/31.5	97.18	77.5/53.0
crude	96.57	81.1/52.4	96.79	86.1/52.4
trade	97.82	81.4/49.1	97.91	78.3/56.0

From these experiments, we can conclude that components may help with tightly coupled categories that require conjunctions of words (20 newsgroups), but not with the keyword-identifiable categories (Reuters). Judging from the ideas in [JB03], the components help in two cases: a) when the co-appearance of two words is more informative than the sum of informativeness of individual appearances of either word, and b) when the appearance of one word implies the appearance of another word, which does not always appear in the document.

10 Conclusion

In this article, we have presented a unifying framework for various approaches to discrete component analysis, presenting them as a model closely related to ICA but suited for sparse discrete data. We have shown the relationships between existing approaches here such as NMF, PLSI, LDA, MPCA and GaP. For instance, NMF with normalised results corresponds to an approximate maximum likelihood method for LDA, and GaP is the most general family of models. We have also presented the different algorithms available for three different cases, Gamma-Poisson, conditional Gamma-Poisson (allowing sparse component scores), and Dirichlet-multinomial. This extends a number of algorithms previous developed for MPCA and LDA to the general Gamma-Poisson model. Experiments with the Mpca software[11] show that a typical 3GHz desktop machine can build models in a few days with K in the hundreds for 3 gigabytes of text.

[11] http://www.componentanalysis.org

These models share many similarities with both PCA and ICA, and are thus useful in a range of feature engineering tasks in machine learning and pattern recognition. A rich literature is alsoemerging extending the model in a variety of directions. This is as much caused by the surprising performance of the algorithms, as it is by the availability of general Gibbs sampling algorithms that allow sophisticated modelling.

Acknowledgments

Wray Buntine's work was supported by the Academy of Finland under the PROSE Project, by Finnish Technology Development Center (TEKES) under the Search-Ina-Box Project, by the IST Programme of the European Community under ALVIS Superpeer Semantic Search Engine (IST-1-002068-IP) and the PASCAL Network of Excellence (IST-2002-5006778). Aleks Jakulin was supported by the Slovenian Research Agency and by the IST Programme of the European Community under SEKT Semantically Enabled Knowledge Technologies (IST-1-506826-IP). The Mpca software used in the experiments was co-developed by a number of authors, reported at the code website. The experiments were supported by the document processing environment and test collections at the CoSCo group, and the information retrieval software Ydin of Sami Perttu. Both authors would also like to acknowledge very helpful comments from the referees.

References

[AGvR03] L. Azzopardi, M. Girolami, and K. van Risjbergen. Investigating the relationship between language model perplexity and ir precision-recall measures. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 369–370, 2003.

[BJ04] W. Buntine and A. Jakulin. Applying discrete PCA in data analysis. In *UAI-2004*, Banff, Canada, 2004.

[BKG03] E. Bingham, A. Kabán, and M. Girolami. Topic identification in dynamical text by complexity pursuit. *Neural Process. Lett.*, 17(1):69–83, 2003.

[BNJ03] D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[BPT04] W.L. Buntine, S. Perttu, and V. Tuulos. Using discrete PCA on web pages. In *Workshop on Statistical Approaches to Web Mining, SAWM'04*, 2004. At ECML 2004.

[BS94] J.M. Bernardo and A.F.M. Smith. *Bayesian Theory.* John Wiley, Chichester, 1994.

[Bun02] W.L. Buntine. Variational extensions to EM and multinomial PCA. In *13th European Conference on Machine Learning (ECML'02)*, Helsinki, Finland, 2002.

[BYRN99] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* Addison Wesley, 1999.

[Can04] J. Canny. GaP: a factor model for discrete data. In *SIGIR 2004*, pages 122–129, 2004.

[CB90] G. Casella and R.L. Berger. *Statistical Inference.* Wadsworth & Brooks/Cole, Belmont, CA, 1990.

[CB94] B.S. Clarke and A.R. Barron. Jeffrey's prior is asymptotically least favorable under entropy risk. *Journal of Statistical Planning and Inference*, 41:37–60, 1994.

[CC95] B.P. Carlin and S. Chib. Bayesian model choice via MCMC. *Journal of the Royal Statistical Society B*, 57:473–484, 1995.

[CDS01] M. Collins, S. Dasgupta, and R.E. Schapire. A generalization of principal component analysis to the exponential family. In *NIPS*13*, 2001.

[CJR04] J. D. Clinton, S. Jackman, and D. Rivers. The statistical analysis of roll call voting: A unified approach. *American Political Science Review*, 98(2):355–370, 2004.

[CR96] G. Casella and C.P. Robert. Rao-Blackewellization of sampling schemes. *Biometrika*, 83(1):81–94, 1996.

[DDL$^+$90] S.C. Deerwester, S.T. Dumais, T.K. Landauer, G.W. Furnas, and R.A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[deL03] J. de Leeuw. Principal component analysis of binary data: Applications to roll-call-analysis. Technical Report 364, UCLA Department of Statistics, 2003.

[Dun94] T. Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74, 1994.

[GB00] Z. Ghahramani and M.J. Beal. Propagation algorithms for variational Bayesian learning. In *NIPS*, pages 507–513, 2000.

[GCSR95] A. Gelman, J.B. Carlin, H.S. Stern, and D.B. Rubin. *Bayesian Data Analysis.* Chapman & Hall, 1995.

[GG05] E. Gaussier and C. Goutte. Relation between PLSA and NMF and implications. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 601–602. ACM Press, 2005.

[GS02] T.L. Griffiths and M. Steyvers. A probabilistic approach to semantic representation. In *Proc. of the 24th Annual Conference of the Cognitive Science Society*, 2002.

[GS04] T.L. Griffiths and M. Steyvers. Finding scientific topics. *PNAS Colloquium*, 2004.

[HB97] T. Hofmann and J.M. Buhmann. Pairwise data clustering by deterministic annealing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(1):1–14, 1997.

[HKO01] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis.* John Wiley & Sons, 2001.

[HLL83] P. Holland, K. B. Laskey, and S. Leinhardt. Stochastic blockmodels: Some first steps. *Social Networks*, 5:109–137, 1983.

[HO00] A. Hyvärinen and E. Oja. Independent component analysis: algorithms and applications. *Neural Netw.*, 13(4-5):411–430, 2000.

[Hof99] T. Hofmann. Probabilistic latent semantic indexing. In *Research and Development in Information Retrieval*, pages 50–57, 1999.

[JB03] A. Jakulin and I. Bratko. Analyzing attribute dependencies. In N. Lavrač, D. Gamberger, H. Blockeel, and L. Todorovski, editors, *PKDD 2003*, volume 2838 of *LNAI*, pages 229–240. Springer-Verlag, September 2003.

[Joa98] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, number 1398, pages 137–142, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE.

[Joa99] T. Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1999.

[LS99] D. Lee and H. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

[LYRL04] D.D. Lewis, Y. Yand, T.G. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

[MKB79] K.V. Mardia, J.T. Kent, and J.M. Bibby. *Multivariate Analysis*. Academic Press, 1979.

[ML02] T. Minka and J. Lafferty. Expectation-propagation for the generative aspect model. In *UAI-2002*, Edmonton, 2002.

[MN89] P. McCullagh and J.A. Nelder. *Generalized Linear Models*. Chapman and Hall, London, second edition, 1989.

[Poo00] K.T. Poole. Non-parametric unfolding of binary choice data. *Political Analysis*, 8(3):211–232, 2000.

[PSD00] J.K. Pritchard, M. Stephens, and P.J. Donnelly. Inference of population structure using multilocus genotype data. *Genetics*, 155:945–959, 2000.

[PTL93] F. Pereira, N. Tishby, and L. Lee. Distributional clustering of English words. In *Proceedings of ACL-93*, June 1993.

[Ros89] S.M. Ross. *Introduction to Probability Models*. Academic Press, fourth edition, 1989.

[Row98] S. Roweis. EM algorithms for PCA and SPCA. In M.I. Jordan, M.J. Kearns, and S.A. Solla, editors, *Advances in Neural Information Processing Systems*, volume 10. The MIT Press, 1998.

[SN97] T.A.B. Snijders and K. Nowicki. Estimation and prediction for stochastic block models for graphs with latent block structure. *Journal of Classification*, 14:75–100, 1997.

[TB99] M.E. Tipping and C.M. Bishop. Probabilistic principal components analysis. *J. Roy. Statistical Society B*, 61(3):611–622, 1999.

[Tit] D.M. Titterington. Some aspects of latent structure analysis. In this volume.

[vGv99] P.G.M. van der Heijden, Z. Gilula, and L.A. van der Ark. An extended study into the relationship between correspondence analysis and latent class analysis. *Sociological Methodology*, 29:147–186, 1999.

[WM82] M.A. Woodbury and K.G. Manton. A new procedure for analysis of medical classification. *Methods Inf Med*, 21:210–220, 1982.

[WMM05] X. Wang, N. Mohanty, and A. McCallum. Group and topic discovery from relations and text. In *The 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining Workshop on Link Discovery: Issues, Approaches and Applications (LinkKDD-05)*, pages 28–35, 2005.

[YYT05] K. Yu, S. Yu, and V. Tresp. Dirichlet enhanced latent semantic analysis. In L.K. Saul, Y. Weiss, and L. Bottou, editors, *Proc. of the 10th International Workshop on Artificial Intelligence and Statistics*, 2005.