Probabilistic semi-nonnegative matrix factorization: a Skellam-based framework

Benoît Fuentes, Gael Richard

Probabilistic semi-nonnegative matrix factorization: a Skellam-based framework

Benoit Fuentes and Gaël Richard

Abstract—We present a new probabilistic model to address semi-nonnegative matrix factorization (SNMF), called Skellam-SNMF. It is a hierarchical generative model consisting of prior components, Skellam-distributed hidden variables and observed data. Two inference algorithms are derived: Expectation-Maximization (EM) algorithm for maximum *a posteriori* estimation and Variational Bayes EM (VBEM) for full Bayesian inference, including the estimation of parameters prior distribution. From this Skellam-based model, we also introduce a new divergence $\mathcal{D}$ between a real-valued target data x and two nonnegative parameters λ_0 and λ_1 such that $\mathcal{D}\left(x \mid \lambda_0, \lambda_1\right) = 0 \Leftrightarrow x = \lambda_0 - \lambda_1$, which is a generalization of the Kullback-Leibler (KL) divergence. Finally, we conduct experimental studies on those new algorithms in order to understand their behavior and prove that they can outperform the classic SNMF approach on real data in a task of automatic clustering.

Index Terms—Semi-Nonnegative Matrix Factorization, Skellam Distribution, Clustering, Bayesian inference

1 Introduction

Matrix factorization, which consists in expressing or approximating a given matrix $\boldsymbol{X}$ as the product of two matrices $\boldsymbol{W}$ (called *atoms* in this paper) and $\boldsymbol{\lambda}$ (called *activations* in this paper), *i.e.* $\boldsymbol{X} = \boldsymbol{W}\boldsymbol{\lambda}$ or $\boldsymbol{X} \approx \boldsymbol{W}\boldsymbol{\lambda}$, has been widely used in data analysis, signal processing and machine learning over many decades. There is a large number of techniques to address this problem, including principal component analysis (PCA), independent component analysis (ICA) [1], or Dictionary Learning [2] just to name a few. In some applications where observed matrix $\boldsymbol{X} \geq 0$, an additional constraint can be added to factors $\boldsymbol{W}$ and $\boldsymbol{\lambda}$ so they remain within the positive orthant, leading to the nonnegative matrix factorization problem (NMF) [3]. Beyond the innumerable applications of NMF that can be found in the literature, in fields such as astronomy [4], audio signal processing [5], bioinformatics [6], text mining [7], etc., there exists a great variety of theoretical work on NMF, focusing on different aspects of the problem [8]. Without being exhaustive, one can mention studies on objective functions [9], [10], efficient algorithms [11] or probabilistic interpretation [12], [13], [14].

More recently, in domains such as energy efficiency [15], gene clustering [16], template matching [17], hidden representation learning [18], or computational imaging [19], a number of studies have made use of an alternative model called semi-nonnegative matrix factorization (SNMF), where observed matrix $\boldsymbol{X}$ is real-valued and where a nonnegativity constraint is added on the $\boldsymbol{\lambda}$ factor, leaving $\boldsymbol{W}$ unconstrained. SNMF was first introduced by Ding *et. al.* [20]. They define this problem as a classic optimization problem: given a matrix $\boldsymbol{X} \in \mathbb{R}^{I \times J}$, and a rank K, solve

$$\min_{\boldsymbol{W} \in \mathbb{R}^{I \times K}, \boldsymbol{\lambda} \in \mathbb{R}^{K \times J}} \|\boldsymbol{X} - \boldsymbol{W}\boldsymbol{\lambda}\|_F^2 \text{ such that } \boldsymbol{\lambda} \geq 0, \quad (1)$$

where $\|\cdot\|_F$ is the Frobenius norm. As for classical NMF [3], this problem is solved by alternatively updating $\boldsymbol{W}$ and $\boldsymbol{\lambda}$. Update of $\boldsymbol{W}$ is performed via least square method and update of $\boldsymbol{\lambda}$ is performed via some multiplicative update which ensures the nonnegativity of $\boldsymbol{\lambda}$. Following this first article on SNMF, a few theoretical studies have been conducted in order to better understand this problem or to provide alternative solutions. In [21] and [22], the notion of semi-nonnegative rank of matrix $\boldsymbol{X}$ is introduced and SNMF algorithms under exact reconstruction constraint are developed. Gillis *et. al.* [22] also put forward improvements to the original SNMF algorithm in order to overcome some former drawbacks such as numerical instability or slowness of convergence. Other studies focus on interpretability of parameters $\boldsymbol{W}$ and $\boldsymbol{\lambda}$ by adding extra regularization terms. In [23], a constraint on $\boldsymbol{W}$ is introduced in order to minimize the maximum angle between any two columns of $\boldsymbol{W}$. In [15], the regularization term is designed to minimize the total variation of each row of $\boldsymbol{\lambda}$.

Although there seems to be a growing interest in this problem, knowledge about SNMF is limited compared to that of NMF. In order to make our contribution, in this paper we formulate SNMF as a statistical inference problem by developing a probabilistic framework suitable for this type of semi-nonnegative model. This framework, called Skellam-SNMF, is based on the Skellam distribution. It is a generalization to signed data of either Poisson NMF [12] or probabilistic latent semantic analysis (PLSA) [13] – also known as probabilistic latent component analysis (PLCA) [14] – and its development into the fully-probabilistic latent Dirichlet allocation (LDA) model [24]. This will lead us to introduce a generalization to signed data of the Kullback-Leibler divergence, as an alternative to the classic Euclidean norm. We will also explain how to add priors on the model

• *The authors are with LTCI, Institut Polytechnique de Paris, Télécom Paris, Palaiseau, France, e-mail: bf@benoit-fuentes.fr, gael.richard@telecom-paris.fr.*

This research leading to this paper has been conducted by Benoit Fuentes while working in Smart Impulse, a french company.

parameters as a way to perform regularization, and two inference algorithms will be developed in order to estimate factor matrices $\boldsymbol{W}$ and $\boldsymbol{\lambda}$: one for standard maximum *a posteriori* estimation, and one for full Bayesian inference. Finally, we will see how to automatically infer the prior distribution of the parameters which shall open the path for online algorithms. By formulating it as a generalization of existing probabilistic models, the SNMF problem will benefit for future research from all improvements and enhancements that have been made on probalistic NMF. We think for instance of generalized tensor factorizations [25], dynamic models [26], sophisticated *ad hoc* models [27], etc.

The paper is organized as follows. After having presented in section 2 some properties about the Skellam distribution, the Skellam-SNMF model is introduced in section 3. Sections 4 and 5 are dedicated to the derivation of two inference algorithms. In section 6 we conduct experiments on toy examples in order to better understand the behavior of our algorithms and we compare Skellam-SNMF with other SNMF methods on real data on a simple clustering problem. Finally, we present our conclusions and ideas for future work in section 7.

Before tackling the subject, let us present the notations that will be used in the sequel. The bold letters, whether upper of lower case, always refer to sets of scalars, including tensors or matrices. The letters X and Z are dedicated to observed data and hidden sources respectively. The letter λ is always used for nonnegative parameters of Poisson or Skellam distributions. A bar an top indicates that the parameter is expressed as a function of other basic parameters (*i.e.* $\bar{\lambda}_s = \sum_n \lambda_{sn}$). Finally, the letter $\boldsymbol{\theta}$ is used to designate a set of nonnegative parameters subject to normalization constraints.

2 SKELLAM DISTRIBUTION

Skellam-SNMF is based on a linear source mixture model where individual sources are modeled as Skellam random variables and we present in this section important properties about this distribution. All proofs are reported in the supplementary material. A Skellam random variable (r.v.) X is defined as the difference of two independent Poisson random variables:

$$\begin{cases} X_0 & \sim \text{Pois}\left(\lambda_0\right) \\ X_1 & \sim \text{Pois}\left(\lambda_1\right) \end{cases} \Leftrightarrow X = X_0 - X_1 \sim \text{Skell}\left(\lambda_0, \lambda_1\right). \quad (2)$$

Parameters λ_0 and λ_1 are nonnegative and mean and variance of X are given by

$$\langle X \rangle = \lambda_0 - \lambda_1, \quad (3)$$

$$\text{Var}\left(X\right) = \lambda_0 + \lambda_1. \quad (4)$$

There exists several equivalent expressions for the Skellam distribution [28] and the one that will be used in this paper is the following:

$$P\left(X = x\right) = \frac{{}_0F_1\left(|x|+1, \lambda_0\lambda_1\right)}{\Gamma\left(|x|+1\right)} \prod_{s\in\{0,1\}} e^{-\lambda_s} \lambda_s^{\max((-1)^s x, 0)} \quad (5)$$

where ${}_0F_1$ is the confluent hypergeometric limit function (which is closely related to the modified Bessel function of the first kind) and where $x \in \mathbb{Z}$. It is easy to verify that this distribution is simplified into the Poisson distribution if $\lambda_1 = 0$. A key property is that the sum of independant Skellam r.v. $Z_n \sim \text{Skell}\left(\lambda_{0,n}, \lambda_{1,n}\right)$ is a also a Skellam r.v.:

$$X = \sum_n Z_n \sim \text{Skell}\left(\sum_n \lambda_{0,n}, \sum_n \lambda_{1,n}\right). \quad (6)$$

We refer $\{Z_n\}$ as the *hidden Skellam sources* and X as the *observed mixture* or *observed data*. Besides, the underlying Poisson r.v. $\{Z_{sn} \sim \text{Pois}\left(\lambda_{sn}\right)\}_{s\in\{0,1\},n}$ such that $Z_n = Z_{0,n} - Z_{1,n}$ are called *hidden Poisson sources.*

Now, we are interested in the posterior distribution of those hidden Poisson sources given the observed mixture, since it will be useful during the derivation of the statistical inference algorithms used later on. First, it can be proven that the expectation of this posterior distribution is given by:

$$\langle Z_{sn} \mid X = x \rangle = \lambda_{sn} \left[\frac{\max\left((-1)^s x, 0\right)}{\bar{\lambda}_s} + \frac{\bar{\lambda}_{1-s}}{|x| + 1 + \sqrt{\bar{\lambda}_0\bar{\lambda}_1}\,\text{R}_{|x|+1}\left(2\sqrt{\bar{\lambda}_0\bar{\lambda}_1}\right)} \right] \quad (7)$$

where $\bar{\lambda}_s = \sum_n \lambda_{sn}$ for $s \in \{0,1\}$ and where $\text{R}_x\left(z\right)$ is the ratio of modified Bessel functions of the first king [29]:

$$\text{R}_x\left(z\right) = \frac{\text{I}_{x+1}\left(z\right)}{\text{I}_x\left(z\right)}. \quad (8)$$

Then, using Bayes rule, one can give the full posterior distribution of $\boldsymbol{Z} = \{Z_{sn}\}$ with respect to X and parameters $\boldsymbol{\lambda} = \{\lambda_{sn}\}$:

$$P\left(\boldsymbol{Z} = \boldsymbol{z} \mid X = x\right) = p\left(\boldsymbol{z}; \boldsymbol{\lambda}, x\right) \quad (9)$$

with

$$p\left(\boldsymbol{z}; \boldsymbol{\lambda}, x\right) = \text{D}\left(\boldsymbol{\lambda}, x\right) \frac{\prod_{sn} \lambda_{sn}^{z_{sn}}}{\prod_{sn} z_{sn}!} \mathbb{1}_{\{x = \sum_n z_{0,n} - z_{1,n}\}} \quad (10)$$

where $\mathbb{1}$ is the indicator function and where

$$\text{D}\left(\boldsymbol{\lambda}, x\right) = \frac{\prod_s \left(\sum_n \lambda_{sn}\right)^{-\max((-1)^s x, 0)}}{{}_0F_1\left(|x|+1, \prod_s \sum_n \lambda_{sn}\right)} \Gamma\left(|x|+1\right) \quad (11)$$

is the normalization factor. To our knowledge, such a distribution has not yet been introduced in the literature. We decide to name it the ***diffnomial*** distribution, as a reference to the equivalent *multinomial* law in the Poisson mixture case[1] :

$$\left(\boldsymbol{Z} \mid X = x\right) \sim \text{DiffNomial}\left(x, \boldsymbol{\lambda}\right). \quad (12)$$

It is easy to verify that the diffnomial law is indeed simplified into a multinomial law if $\lambda_{1,n}$ and $z_{1,n}$ are set to 0 for all n.

Now we have presented all necessary background preliminaries, the Skellam-SNMF model can be introduced.

1. It is a well known result that if $Z_n \sim \text{Pois}\left(\lambda_n\right)$ for $n = 1 \ldots N$ and if $X = \sum_n Z_n$, then $\{Z_n\} \mid X$ follows a multinomial distribution.

3 SKELLAM-SNMF: THE GENERATIVE MODEL

We aim at approximating a matrix $\boldsymbol{X}$ as a factorization of two matrices $\boldsymbol{X} \approx \boldsymbol{W}\boldsymbol{\lambda}$ where $\boldsymbol{W} \in \mathbb{R}^{I\times K}$ contains real values and $\boldsymbol{\lambda} \in \mathbb{R}_+^{K\times J}$ only nonnegative ones. The main idea in Skellam-NMF is to express atoms matrix $\boldsymbol{W}$ as the difference between two nonnegative matrices $\boldsymbol{W} = \boldsymbol{\theta}_0 - \boldsymbol{\theta}_1$ and then to consider that each coefficient X_{ij} is drawn from a Skellam distribution $X_{ij} \sim \text{Skell}\left([\boldsymbol{\theta}_0\boldsymbol{\lambda}]_{ij}, [\boldsymbol{\theta}_1\boldsymbol{\lambda}]_{ij}\right)$. An appropriate estimator for parameters $\boldsymbol{\theta}_0$, $\boldsymbol{\theta}_1$ and $\boldsymbol{\lambda}$ will try to make the expected value of Skellam distribution $[\boldsymbol{\theta}_0\boldsymbol{\lambda}]_{ij} - [\boldsymbol{\theta}_1\boldsymbol{\lambda}]_{ij}$ as closed as possible to the observed data and then have the best possible approximation $\boldsymbol{X} \approx \hat{\boldsymbol{X}} = \boldsymbol{\theta}_0\boldsymbol{\lambda} - \boldsymbol{\theta}_1\boldsymbol{\lambda} = \boldsymbol{W}\boldsymbol{\lambda}$. With this generative model, only integers are allowed for the coefficients of $\boldsymbol{X}$. For real-valued data, the idea is to consider $\boldsymbol{X}$ as the mean of M Skellam-distributed matrices $\boldsymbol{X} = \frac{1}{M}\sum_{m=1}^{M}\boldsymbol{X}^m$ and then make M tends towards ∞.

3.1 Normalization constraints on atoms

From now on, we gather the two matrices $\boldsymbol{\theta}_0$ and $\boldsymbol{\theta}_1$ in a single tensor $\boldsymbol{\theta} = \{\theta_{sik}\}_{s\in\{0,1\},i=1\dots I,k=1\dots K}$ also called *atoms* herein. We decide to add the following normalization constraint on $\boldsymbol{\theta}$:

$$\forall k,\ \sum_{si}\theta_{sik} = 1. \tag{13}$$

In order to notify such a constraint, we will use notation $\theta_{si|k}$ instead of θ_{sik}. This presents many advantages. First of all, it overcomes an homogeneity flaw that happens when the observed data $\boldsymbol{X}$ has a physical dimension, such as Watts (W), lumen (lm), etc.: since $\hat{\boldsymbol{X}} = \boldsymbol{W}\boldsymbol{\lambda}$ should have the same physical dimension, it makes more sense to have one normalized factor with no dimension whatsoever and one factor that carries the physical dimension, than two factors that would be expressed in square root of the dimension. Then, a practical advantage is that this constraint leads to the following simplification

$$\sum_{ijks}\theta_{si|k}\lambda_{kj} = \sum_{kj}\lambda_{kj} \tag{14}$$

which facilitates the derivation of both inference algorithms presented in sections 4 and 5. It also naturally prevents any estimation algorithm from numerical stability problems, with for instance atoms tending towards very small values and activations tending towards very high ones. Finally, it removes a well known identifiability problem, namely the scale invariance between columns of $\boldsymbol{W}$ and rows of $\boldsymbol{\lambda}$. Note that the choice to apply a normalization constraint on atoms $\boldsymbol{\theta}$ is arbitrary and we could have normalized activations instead.

3.2 Priors on parameters

We consider the possibility of adding priors on parameters as a way to both get rid of all identifiability problems that might remain and to add regularization terms in the objective function to be optimized. This can help to find more relevant estimates for the parameters, depending on the application. In order to stay in a easy-to-compute probabilistic framework, we suggest the use of conjugate priors for the Skellam likelihood function, which happens to be Gamma priors for non-normalized parameters λ_{kj} and Dirichlet priors for normalized parameters $\theta_{si|k}$.

3.3 The generative model

Now, we can detail the full generative model of Skellam-SNMF. In order to consider both cases according to whether $\boldsymbol{X}$ is composed of integers or real numbers, we let the number M undefined. Just be aware that the two values of interest are $M = 1$ for integer data or $M \to \infty$ for real data. Note also that we are going to artificially over-parameterize our model by defining M atoms tensors and M activations matrices. This will be discussed at the end of this section. The first step of the generative model is to draw parameters from their prior distributions:

$$\forall m = 1\dots M,\ \left\{\theta^m_{si|k}\right\}_{si} \sim \text{Dirichlet}\left(\left\{\alpha_{\varphi(s,i,k)}\right\}_{si}\right), \tag{15}$$

$$\lambda^m_{kj} \sim \text{Gamma}\left(\alpha_{\upsilon(k,j)}, \beta_{\omega(k,j)}\right). \tag{16}$$

Here, $\boldsymbol{\alpha} = \{\alpha_a\}$ and $\boldsymbol{\beta} = \{\beta_b\}$ are two sets of non-negative shape and rate hyperparameters, and φ, υ and ω are functions that map the parameters to the hyperparameters. Using such maps allows us to keep the possibility for several parameters to share a same hyperparameter, reducing then their number. Later, we will see how hyperparameters can be learned from the data, and such feature can be useful in order to avoid overfitting. However, we do not permit Gamma and Dirichlet parameters to share a same shape hyperparameter and one must have

$$\{\varphi(s,i,k)\}_{s,i,k} \cap \{\upsilon(k,j)\}_{k,l} = \emptyset. \tag{17}$$

The second step is to draw Poisson hidden variables (or hidden sources) depending on the parameters:

$$Z^m_{sikj} \sim \text{Pois}\left(\bar{\lambda}^m_{sikj}\right) \tag{18}$$

with

$$\bar{\lambda}^m_{sikj} = \theta^m_{si|k}\lambda^m_{kj}. \tag{19}$$

Finally, observed data are computed as:

$$X_{ij} = \frac{1}{M}\sum_{mk}Z^m_{s=0,ikj} - \sum_{mk}Z^m_{s=1,ikj}, \tag{20}$$

leading to Skellam independent random variables for $M\times$ the observed data:

$$MX_{ij} \sim \text{Skell}\left(\sum_m\bar{\lambda}^m_{s=0,ij}, \sum_m\bar{\lambda}^m_{s=1,ij}\right) \tag{21}$$

with

$$\bar{\lambda}^m_{sij} = \sum_k\bar{\lambda}^m_{sikj}. \tag{22}$$

Possibly, we can also consider that some data are missing, meaning that $\{X_{ij}\}$ is observed only for a subset

$$\mathcal{O} \subset \{1,\dots I\}\times\{1,\dots J\} \tag{23}$$

of indexes (i,j). We redefine then the set of observed data as

$$\boldsymbol{X}_{\mathcal{O}} = \{X_{ij}\}_{ij\in\mathcal{O}}. \tag{24}$$

The reason for over-parameterizing the model by drawing M independent pairs of atoms and activations is that otherwise the log-prior probability of the parameters would become negligible compared to the log-likelihood of the data as M grows, making the addition of priors useless. With M draws, priors are "counted" M times, which solves the problem. Now, the trick to getting back to a single atoms factor and a single activations factor is to constrain, when we seek to infer the parameter values, the estimates of $\boldsymbol{\theta}^m$ and $\boldsymbol{\lambda}^m$ to be pairwise equals, *i.e.* $\exists\left(\boldsymbol{\theta},\boldsymbol{\lambda}\right), \forall m,\ \left(\boldsymbol{\theta}^m,\boldsymbol{\lambda}^m\right) = \left(\boldsymbol{\theta},\boldsymbol{\lambda}\right)$.

At this point, any statistical inference algorithms can be applied in order to estimate the best value for the parameters given observed data $\boldsymbol{X}$. In the two following sections, we focus on the Expectation-Maximization (EM) algorithm and the Variational Bayes EM algorithm.

4 Skellam-SNMF with EM algorithm

4.1 Objective function and emergence of a new divergence

In order to estimate the parameters of our model, one can use the Expectation-Maximization algorithm [30] which aims at finding a local maximum of the log-posterior probability of the parameters given the data. In the case of the generative model presented in previous section, the Bayes rule can be used to compute it:

$$\begin{aligned}\ln P\left(\{\boldsymbol{\theta}^m\},\{\boldsymbol{\lambda}^m\} \mid M\boldsymbol{X}\right) = \ln P\left(M\boldsymbol{X} \mid \{\boldsymbol{\theta}^m\}_m,\{\boldsymbol{\lambda}^m\}_m\right)\\ + \sum_m \ln P\left(\boldsymbol{\theta}^m\right) + \sum_m \ln P\left(\boldsymbol{\lambda}^m\right) + cst\end{aligned} \tag{25}$$

where cst does not depend on the parameters. Each of the other terms can be computed using equations (15), (16), (21) and the definition of the corresponding distributions. If now we add a normalization factor $\frac{1}{M}$ so that this quantity does not tend toward $-\infty$ as M goes to $+\infty$, and if we consider, as justified before, only values of parameters such as $\forall m, (\boldsymbol{\theta}^m,\boldsymbol{\lambda}^m) = (\boldsymbol{\theta},\boldsymbol{\lambda})$, we can define the objective function that the EM algorithm will optimize as:

$$\begin{aligned}f^M_{\boldsymbol{X}}\left(\boldsymbol{\theta},\boldsymbol{\lambda}\right) =& \frac{1}{M}\ln P\left(\{\boldsymbol{\theta}^m=\boldsymbol{\theta}\}_m,\{\boldsymbol{\lambda}^m=\boldsymbol{\lambda}\}_m \mid M\boldsymbol{X}\right)\\ =& \mathcal{L}^M_{\boldsymbol{X}}\left(\boldsymbol{\theta},\boldsymbol{\lambda}\right)\\ &+ \sum_{kj}\left(\alpha_{\upsilon(k,j)}-1\right)\ln\lambda_{kj} - \beta_{\omega(k,j)}\lambda_{kj}\\ &+ \sum_{sik}\left(\alpha_{\varphi(s,i,k)}-1\right)\ln\theta_{si|k} + cst/M\end{aligned} \tag{26}$$

where

$$\mathcal{L}^M_{\boldsymbol{X}}\left(\boldsymbol{\theta},\boldsymbol{\lambda}\right) = \frac{1}{M}\ln P\left(M\boldsymbol{X} \mid \{\boldsymbol{\theta}^m=\boldsymbol{\theta}\}_m,\{\boldsymbol{\lambda}^m=\boldsymbol{\lambda}\}_m\right) \tag{27}$$

can be interpreted as a data fitting term, and the other terms as regularization terms (cst/M is ignored thereafter). If $\boldsymbol{X}$ contains integer values ($M=1$), equation (5) gives:

$$\begin{aligned}\mathcal{L}^1_{\boldsymbol{X}}\left(\boldsymbol{\theta},\boldsymbol{\lambda}\right) = \sum_{ij\in\mathcal{O}} \ln\frac{{}_0F_1\left(|X_{ij}|+1,\sigma_{ij}\right)}{\Gamma\left(|X_{ij}|+1\right)} +\\ \sum_s -\bar{\lambda}_{sij} + \max\left((-1)^s X_{ij},0\right)\ln\bar{\lambda}_{sij}\end{aligned} \tag{28}$$

with

$$\sigma_{ij} = \bar{\lambda}_{s=0,ij}\bar{\lambda}_{s=1,ij}. \tag{29}$$

and $\bar{\lambda}_{sij}$ given by equations (19) and (22). If $\boldsymbol{X}$ is real-valued ($M=+\infty$), it can be proven using asymptotic expansion of ${}_0F_1$ (see supplementary material) that:

$$\mathcal{L}^\infty_{\boldsymbol{X}}\left(\boldsymbol{\theta},\boldsymbol{\lambda}\right) = -\sum_{ij\in\mathcal{O}}\mathcal{D}\left(X_{ij} \mid \bar{\lambda}_{s=0,ij},\bar{\lambda}_{s=1,ij}\right) \tag{30}$$

with

$$\begin{aligned}\mathcal{D}\left(x \mid \lambda_0,\lambda_1\right) = \sum_{s\in\{0,1\}}\lambda_s - \max\left((-1)^s x,0\right)\ln\lambda_s\\ - \sqrt{x^2+4\lambda_0\lambda_1} + |x|\ln\left(\frac{|x|+\sqrt{x^2+4\lambda_0\lambda_1}}{2}\right).\end{aligned} \tag{31}$$

The function $\mathcal{D}\left(x \mid \lambda_0,\lambda_1\right)$ can be seen as a divergence function: it is indeed always positive or null by construction and vanishes if and only if $x=\lambda_0-\lambda_1$. It is actually a generalization to signed data of the Kullback-Leibler (KL) divergence $\mathcal{D}_{\mathrm{KL}}$ [3] since $\mathcal{D}\left(x \mid \lambda_0,0\right) = \mathcal{D}_{\mathrm{KL}}\left(x \mid \lambda_0\right)$ for nonnegative values of x. Note also that as for the KL divergence, it respects the following property:

$$\forall\mu>0,\ \mathcal{D}\left(\mu x \mid \mu\lambda_0,\mu\lambda_1\right) = \mu\mathcal{D}\left(x \mid \lambda_0,\lambda_1\right). \tag{32}$$

To our knowledge, this divergence has never been introduced in the literature.

4.2 Derivation of the EM algorithm

Each iteration of the EM algorithm consists of two steps. First the expectation step, were the log-likelihood of the complete data $\boldsymbol{Y}$ (observed and latent variables) is computed as well as its conditional expectation given current estimates for the parameters. Then the maximization step, where this last quantity is maximized with respect to the parameters. For the definition of $\boldsymbol{Y}$, we can either include or exclude missing data and latent sources that are linked to them. We decide to include them for a practical reason: when deriving the algorithm, it allows to perform simplification (14) at some point, and without it, we would not have a simple closed form solution in the maximization stage. The downside in return is that it might slow down the speed of convergence, since it is a known feature of the EM algorithm that the more hidden variables compared to number of observed data, the slowest the convergence. Curious readers may refer to the supplementary material, where the derivation of the EM algorithm is fully detailed. The computation is quite straightforward once the formula of the posterior expectation of the hidden sources (7) is known. The resulting update rules for the parameters are summarized in Algorithm 1.

5 Full Bayesian inference

5.1 VBEM: Motivations and general guidelines

Whether it is to estimate the posterior distribution of the parameters given the observed data and the hyperparameters, to perform hyperparameters estimation or to compare two given models, full Bayesian methods can be very useful. Here we focus on one of them called Variational

Algorithm 1: EM algorithm for Skellam-SNMF.

Input: $\boldsymbol{X}$, $\mathcal{O}$, M, $\epsilon = 0$ if shape hyperparameters $\boldsymbol{\alpha} \geq 1$ else $\epsilon > 0$

Output: $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\lambda}}$

initialize $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\lambda}}$

repeat

/* Compute the model and the multiplicative updates */

$\bar{\lambda}_{sij} \leftarrow \sum_k \hat{\theta}_{si|k} \hat{\lambda}_{kj}$

optional: compute objective function $f_{\boldsymbol{X}}^{M}\left(\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\lambda}}\right)$ (equation (26))

$\sigma_{ij} \leftarrow \bar{\lambda}_{s=0,ij} \bar{\lambda}_{s=1,ij}$

$$U_{sij} \leftarrow \begin{cases} 1 & \text{if } ij \notin \mathcal{O} \\ \frac{\max((-1)^s X_{ij}, 0)}{\bar{\lambda}_{sij}} + \frac{\bar{\lambda}_{1-s,ij}}{|X_{ij}|+1+\sqrt{\sigma_{ij}}\,\mathrm{R}_{|X_{ij}|+1}\left(2\sqrt{\sigma_{ij}}\right)} & \text{if } ij \in \mathcal{O} \text{ and } M = 1 \\ \frac{\max((-1)^s X_{ij}, 0)}{\bar{\lambda}_{sij}} + \frac{2\bar{\lambda}_{1-s,ij}}{|X_{ij}|+\sqrt{X_{ij}^2+4\sigma_{ij}}} & \text{if } ij \in \mathcal{O} \text{ and } M = \infty \end{cases}$$

$U_{kj}^{\text{act}} \leftarrow \sum_{si} U_{sij} \hat{\theta}_{si|k}$ $\qquad$ $U_{sik}^{\text{atoms}} \leftarrow \sum_j U_{sij} \hat{\lambda}_{kj}$

/* Update parameters */

$\hat{\lambda}_{kj} \leftarrow \hat{\lambda}_{kj} U_{kj}^{\text{act}} + \alpha_{\upsilon(k,j)} - 1$ $\qquad$ $\hat{\theta}_{sik} \leftarrow \hat{\theta}_{si|k} U_{sik}^{\text{atoms}} + \alpha_{\varphi(s,i,k)} - 1$

$\hat{\lambda}_{kj} \leftarrow \min\left(\hat{\lambda}_{kj}, \epsilon\right)$ $\qquad$ $\hat{\theta}_{sik} \leftarrow \min\left(\hat{\theta}_{sik}, \epsilon\right)$

$\hat{\lambda}_{kj} \leftarrow \hat{\lambda}_{kj} / \left(1 + \beta_{\omega(k,j)}\right)$ $\qquad$ $\hat{\theta}_{si|k} \leftarrow \hat{\theta}_{si|k} / \sum_{s'i'} \hat{\theta}_{s'i'|k}$

until *convergence*;

Bayesian EM (VBEM) [31]. It allows both to find an approximation of the posterior distribution of parameters and hidden variables (we regroup them into a single variable $\boldsymbol{W} = (\boldsymbol{Z}, \boldsymbol{\theta}, \boldsymbol{\lambda})$):

$$\mathcal{Q}(\boldsymbol{W}) \approx P(\boldsymbol{W} \mid \boldsymbol{X}_{\mathcal{O}}) \tag{33}$$

and to compute the Evidence Lower BOund (ELBO) $\mathcal{E}$, a lower bound for the log-evidence of the data, which has generally no closed-form solution:

$$\mathcal{E}(\mathcal{Q}; \boldsymbol{X}_{\mathcal{O}}) \leq \ln P(\boldsymbol{X}_{\mathcal{O}}) \tag{34}$$

with

$$\mathcal{E}(\mathcal{Q}; \boldsymbol{X}_{\mathcal{O}}) = \sum_{\boldsymbol{W}} \mathcal{Q}(\boldsymbol{W}) \ln \frac{P(\boldsymbol{W}, \boldsymbol{X}_{\mathcal{O}})}{\mathcal{Q}(\boldsymbol{W})} \tag{35}$$

and

$$P(\boldsymbol{X}_{\mathcal{O}}) = \sum_{W} P(\boldsymbol{W}, \boldsymbol{X}_{\mathcal{O}}). \tag{36}$$

The goal of VBEM is to maximize $\mathcal{E}(\mathcal{Q}; \boldsymbol{X}_{\mathcal{O}})$ with respect to $\mathcal{Q}$. To do so, $\mathcal{Q}(\boldsymbol{W})$ is usually factorized as

$$\mathcal{Q}(\boldsymbol{W}) = \prod_{n=1}^{N} q_n(\boldsymbol{W}_n) \tag{37}$$

where $W_1, \dots W_N$ is some partition of all latent variables $\boldsymbol{W}$. It is shown that the following update rules for the q_n distribution make the ELBO non decreasing (the notation $\langle f(x_1, \dots) \rangle_{q_1(x_1), \dots}$ is used for the expected value of $f(x_1, \dots)$ taking $q_1, \dots$ as the probability distributions for $x_1, \dots$):

$$\ln q_n(\boldsymbol{W}_n) = \left\langle \ln P(\boldsymbol{W}_1, \dots \boldsymbol{W}_N, \boldsymbol{X}_{\mathcal{O}}) \right\rangle_{\{q_{n'}(\boldsymbol{W}_{n'})\}_{n' \neq n}} + cst. \tag{38}$$

For the following, we define the normalized ELBO as

$$g^M(\mathcal{Q}; \boldsymbol{X}_{\mathcal{O}}) = \frac{1}{M} \mathcal{E}(\mathcal{Q}; \boldsymbol{X}_{\mathcal{O}}), \tag{39}$$

which turns out to be well defined when M tends towards infinity. This corresponds to the objective function to be maximized.

5.2 Derivation of VBEM algorithm

Because this will lead to VBEM algorithm that is "easy" to derive, we decide to take a fully factorized distribution for $\mathcal{Q}$:

$$\mathcal{Q}(\boldsymbol{W}) = \prod_{ij} q_{\boldsymbol{Z}_{ij}}\left(\left\{Z_{sikj}^m\right\}_{msk}\right) \prod_{mkj} q_{\lambda_{kj}^m}\left(\lambda_{kj}^m\right) \prod_{mk} q_{\boldsymbol{\theta}_k^m}\left(\left\{\theta_{si|k}^m\right\}_{si}\right). \tag{40}$$

Due to the symmetry with respect to m of the generative process described in section 3.3, VBEM will give similar definitions and update rules for $q_{\lambda_{kj}^m}$ and $q_{\boldsymbol{\theta}_k^m}$ for all m. This means that on condition that they are all initialized the same way – which we will suppose –, they will all be equals over the iterations and we can therefore ignore superscripts m. Note also that, for the same practical reason as for the EM algorithm (see subsection 4.2), hidden sources that are linked to missing data $\left\{Z_{sikj}^m\right\}_{ij \notin \mathcal{O}, s, k, m}$ are not excluded from $\boldsymbol{W}$. At each iteration, we update factor distributions according to equation (38) in the following order: first, updates of parameter distributions $\left\{q_{\lambda_{kj}^m}\right\}_{mkj}$ and $\left\{q_{\boldsymbol{\theta}_k^m}\right\}_{mkj}$[2], and then updates of source distributions

2. It turns out that due to normalization constraint on atoms (13), those updates can be performed independently from each other, and therefore the order does not matter.

$\left\{q_{\boldsymbol{Z}_{ij}}\right\}_{ij}$. The detailed calculations are provided in the supplementary material and we report here the main results. By following these guidelines, we end up with the following posterior distributions:

$$q_{\boldsymbol{Z}_{ij}} = \begin{cases} \prod_{msk} \text{Pois}\left(\bar{\ell}^m_{sikj}\right), & \text{if } ij \notin \mathcal{O} \\ \text{DiffNomial}\left(MX_{ij}, \left\{\bar{\ell}^m_{sikj}\right\}_{msk}\right), & \text{if } ij \in \mathcal{O} \end{cases} \tag{41}$$

$$q_{\lambda^m_{kj}} = q_{\lambda_{kj}} = \text{Gamma}\left(\hat{\alpha}_{kj}, \hat{\beta}_{kj}\right), \tag{42}$$

$$q_{\boldsymbol{\theta}^m_k} = q_{\boldsymbol{\theta}_k} = \text{Dirichlet}\left(\{\hat{\alpha}_{sik}\}_{si}\right) \tag{43}$$

where $\bar{\ell}^m_{sikj}$ can be computed from $\hat{\alpha}_{kj}, \hat{\beta}_{kj}$ and $\hat{\alpha}_{sik}$, and vice versa, leading to a EM-like alternative algorithm. A nice feature is that due to calculation simplifications, it is not necessary to explicitly compute the $\bar{\ell}^m_{sikj}$ variables. The resulting algorithm is described in algorithm 2 and is actually very closed to the EM algorithm.

Now we have a definition for the $\mathcal{Q}$ distribution, the normalized ELBO $g^M\left(\mathcal{Q}; \boldsymbol{X}_{\mathcal{O}}\right)$ (39) can be computed explicitly thanks to equation (35). The developped formula is given in Appendix A. Just know that as for the EM algorithm's objective function, it is composed of three terms that can be interpreted as a data fitting term and two regularization terms for atoms and activations.

5.3 Estimation of hyperparameters

Though it is not justified in theory, the ELBO is often used in the literature as a replacement for the log-evidence of data $\ln P\left(X_{\mathcal{O}}\right)$ (36) in order to perform model selection[3] or hyperparameters estimation [12], [33]. For instance, in the specific case of Poisson-NMF and given a single observed matrix $\boldsymbol{X}$, it is explained in [12] how to infer the model order K and how to alternatively run the VBEM algorithm with fixed hyperparameters, and update the hyperparameters with fixed distribution $\mathcal{Q}$, as a way to improve the estimation of the model parameters.

Here, we focus on an alternative scenario, namely online learning of the hyperparameters given a collection of data $\left(\boldsymbol{X}^1, \ldots \boldsymbol{X}^t, \ldots\right)$, assumed to be independent and identically distributed (i.i.d.). The idea is to update the value of the hyperparameters after each run of the VBEM algorithm on a new observation, yielding an improved VBEM algorithm that is increasingly adapted to observations. To solve this problem, we first consider the case of batch estimation where the collection $\left(\boldsymbol{X}^1, \ldots \boldsymbol{X}^T\right)$ is fully supplied, and then explain how to switch from batch estimation to online estimation.

Assume that for each data $\boldsymbol{X}^t$, VBEM has provided a posterior approximation $\mathcal{Q}^t$, characterized by the "posterior hyperparameters" $\hat{\alpha}^t_{kj}$, $\hat{\beta}^t_{kj}$ and $\hat{\alpha}^t_{sik}$. We then wish to estimate the hyperparameters via maximization of the total normalized ELBO with respect to $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ (in this section, $g^M\left(\mathcal{Q}, \boldsymbol{X}\right)$ is renamed as $g^M\left(\mathcal{Q}, \boldsymbol{X}; \boldsymbol{\alpha}, \boldsymbol{\beta}\right)$; note also that the value of M plays no role in the estimation of the hyperparameters):

$$\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}} = \underset{\substack{\boldsymbol{\alpha} > 0 \\ \boldsymbol{\beta} > 0}}{\arg\max} \sum_t g^M\left(\mathcal{Q}^t, \boldsymbol{X}^t; \boldsymbol{\alpha}, \boldsymbol{\beta}\right) \tag{44}$$

where $g^M\left(\mathcal{Q}^t, \boldsymbol{X}^t; \boldsymbol{\alpha}, \boldsymbol{\beta}\right)$ is given in appendix A. In order to perform this optimization, it is important to calculate the partial derivatives with respect to each hyperparameter with fixed $\mathcal{Q}^t$, even though $\mathcal{Q}^t$ is itself expressed as a function of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. There is no closed-form solution for this optimization, and therefore optimization algorithms must be employed. The proofs of the two following propositions can be found in the supplementary material.

Proposition 1 (Hyperparameters estimation for gamma priors)**.** *For $a \in \upsilon\left(\{(k,j)\}\right)$ (i.e. for shape hyperparameters linked to activations $\boldsymbol{\lambda}$), the following update rules make the normalized ELBO non-decreasing at each iteration (κ refers to the iteration number):*

$$\psi\left(\alpha^{(\kappa+1)}_a\right) = \frac{\sum_{(k,j)\in\upsilon^{-1}(a)} \ln \beta^{(\kappa)}_{\omega(k,j)} + \gamma^T_a}{\left|\upsilon^{-1}\left(a\right)\right|}, \tag{45}$$

$$\beta^{(\kappa+1)}_b = \frac{\sum_{(k,j)\in\omega^{-1}(b)} \alpha^{(\kappa+1)}_{\upsilon(k,j)}}{\delta^T_b}, \tag{46}$$

where $\left|\upsilon^{-1}\left(a\right)\right|$ is the cardinal of inverse image $\varphi^{-1}\left(a\right)$ and where

$$\gamma^T_a = \frac{1}{T}\sum_{t=1}^{T} \sum_{(k,j)\in\upsilon^{-1}(a)} \left(\psi\left(\hat{\alpha}^t_{kj}\right) - \ln \hat{\beta}^t_{kj}\right), \tag{47}$$

$$\delta^T_b = \frac{1}{T}\sum_{t=1}^{T} \sum_{(k,j)\in\omega^{-1}(b)} \frac{\hat{\alpha}^t_{kj}}{\hat{\beta}^t_{kj}}. \tag{48}$$

ψ is the digamma function. Its inverse can be computed using Newton's method (see Appendix C of [34]).

Proposition 2 (Hyperparameters estimation for Dirichlet priors)**.** *For $a \in \varphi\left(\{(s,i,k)\}\right)$ (i.e. for shape hyperparameters linked to atoms $\boldsymbol{\theta}$), the following update rule makes the normalized ELBO non-decreasing at each iteration (κ refers to the iteration number):*

$$\psi\left(\alpha^{(\kappa+1)}_a\right) = \frac{\sum_{sik\in\varphi^{-1}(a)} \psi\left(\sum_{s'i'} \alpha^{(\kappa)}_{\varphi(s'i',k)}\right) + \xi^T_a}{\left|\varphi^{-1}\left(a\right)\right|} \tag{49}$$

with

$$\xi^T_a = \frac{1}{T}\sum_{t=1}^{T} \sum_{sik\in\varphi^{-1}(a)} \psi\left(\hat{\alpha}^t_{sik}\right) - \psi\left(\sum_{s'i'} \hat{\alpha}^t_{s'i'k}\right). \tag{50}$$

It is quite simple to switch to online estimation since quantities γ^T_a, δ^T_b and ξ^T_a can be computed recursively:

$$\gamma^{T+1}_a = (1-c)\,\gamma^T_a + c \sum_{kj\in\upsilon^{-1}(d)} \left(\psi\left(\hat{\alpha}^T_{kj}\right) - \ln \hat{\beta}^T_{kj}\right), \tag{51}$$

$$\delta^{T+1}_b = (1-c)\,\delta^T_b + c \sum_{kj\in\omega^{-1}(e)} \frac{\hat{\alpha}^T_{kj}}{\hat{\beta}^T_{kj}}, \tag{52}$$

$$\xi^{T+1}_a = (1-c)\,\xi^T_a + c \sum_{sik\in\varphi^{-1}(a)} \psi\left(\hat{\alpha}^T_{sik}\right) - \psi\left(\sum_{s'i'} \hat{\alpha}^T_{s'i'k}\right) \tag{53}$$

3. Note that some theoretical work about the consistency of ELBO based model selection has been put forward lately [32].

Algorithm 2: VBEM algorithm for Skellam-SNMF. $\psi = \frac{\Gamma'}{\Gamma}$ is the digamma function.

Input: $\boldsymbol{X}$, $\mathcal{O}$, M

Output: $\{\hat{\alpha}_{kj}\}$, $\left\{\hat{\beta}_{kj}\right\}$ and $\{\hat{\alpha}_{sik}\}$

$\hat{\beta}_{kj} \leftarrow \beta_{\omega(k,j)} + 1$
initialize $\{\hat{\alpha}_{kj}\}$ and $\{\hat{\alpha}_{sik}\}$
$\ell_{kj} \leftarrow \exp\psi\left(\hat{\alpha}_{kj}\right) / \hat{\beta}_{kj}$, $\quad h_{sik} \leftarrow \exp\psi\left(\hat{\alpha}_{sik}\right) / \exp\psi\left(\sum_{s'i'} \hat{\alpha}_{s'i'k}\right)$
repeat
  /* Compute the model and the multiplicative updates */
  $\bar{\ell}_{sij} \leftarrow \sum_k h_{sik}\ell_{kj}$
  optional: compute objective function $g^M\left(\mathcal{Q}, \boldsymbol{X}_{\mathcal{O}}\right)$ (see Appendix A)
  $\sigma_{ij} \leftarrow \bar{\ell}_{s=0,ij}\bar{\ell}_{s=1,ij}$
  $$U_{sij} \leftarrow \begin{cases} 1 & \text{if } ij \notin \mathcal{O} \\ \frac{\max((-1)^s X_{ij}, 0)}{\bar{\ell}_{sij}} + \frac{\bar{\ell}_{1-s,ij}}{|X_{ij}|+1+\sqrt{\sigma_{ij}}\,\mathrm{R}_{|X_{ij}|+1}\left(2\sqrt{\sigma_{ij}}\right)} & \text{if } ij \in \mathcal{O} \text{ and } M = 1 \\ \frac{\max((-1)^s X_{ij}, 0)}{\bar{\ell}_{sij}} + \frac{2\bar{\ell}_{1-s,ij}}{|X_{ij}|+\sqrt{X_{ij}^2+4\sigma_{ij}}} & \text{if } ij \in \mathcal{O} \text{ and } M = \infty \end{cases}$$
  $U^{\text{act}}_{kj} \leftarrow \sum_{si} U_{sij}h_{sik}$ $\quad U^{\text{atoms}}_{sik} \leftarrow \sum_j U_{sij}\ell_{kj}$
  /* Update parameters */
  $\hat{\alpha}_{kj} \leftarrow \ell_{kj}U^{\text{act}}_{kj} + \alpha_{\upsilon(k,j)}$ $\quad \hat{\alpha}_{sik} \leftarrow h_{sik}U^{\text{atoms}}_{sik} + \alpha_{\varphi(s,i,k)}$
  $\ell_{kj} \leftarrow \exp\psi\left(\hat{\alpha}_{kj}\right) / \hat{\beta}_{kj}$, $\quad h_{sik} \leftarrow \exp\psi\left(\hat{\alpha}_{sik}\right) / \exp\psi\left(\sum_{s'i'} \hat{\alpha}_{s'i'k}\right)$
until *convergence*;

with $c = \frac{1}{T+1}$. Therefore, on the condition that a record of those three quantities is kept, as well as the number T, each time a new observation is provided, one can run VBEM, update $\boldsymbol{\gamma}$, $\boldsymbol{\delta}$ and $\boldsymbol{\xi}$ and finally update $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ using propositions 1 and 2.

Note that it may be interesting to fix the value c once for all, independent of T, since it has two advantages. First, it prevents overfitting in the early stages of the process (T small), when there is still little data to learn from, and moreover when estimation of $\hat{\alpha}_{kj}^{t\leq T}$ and $\hat{\alpha}_{sik}^{t\leq T}$ might be poor due to inappropriate values for the hyperparameters. Second, it gradually erases the contributions of past observations, allowing the process to be resilient in case observations $\left(\boldsymbol{X}^1, \ldots \boldsymbol{X}^t, \ldots\right)$ were not strictly identically distributed. c can then be seen as a *learning rate*, and be set to a small value (*e.g.* $c = 0.02$). Finally, γ_a, δ_b and ξ_a can be initialized using equations (47), (48) and (50), with $T = 1$, $\hat{\alpha}_{kj} = \alpha_{\nu(k,j)}$, $\hat{\beta}_{kj} = \beta_{\omega(k,j)}$ and $\hat{\alpha}_{sik} = \alpha_{\varphi(s,i,k)}$.

6 Experimental studies

We conduct several experiments in order to both study the intrinsic performance and characteristics of our estimation algorithms on synthetic data and to evaluate Skellam-SNMF for automatic clustering on real data in comparison with the original SNMF algorithm. Acronyms and other information about the algorithms that will be used in this section are presented in Table 1. All our Skellam-SNMF algorithms are implemented using the Wonterfact python package [35]. This package, developed by the main author of this paper, allows the design of any kind of tensor factorization model, included Skellam-SNMF, with an automatic derivation of the EM or the VBEM algorithm. The code to reproduce all the experiments in this section can be found in the "jupyter" directory of the Wonterfact repository.

TABLE 1
SNMF algorithms. Note that K-means can be interpreted as SNMF with only binary entries for the activations.

Acronym	Description	Remark
Sk_M	Skellam-SNMF with EM algorithm	$M = 1$ or ∞ (see section 3.3)
Sk_M-VB	Skellam-SNMF with VBEM algorithm	$M = 1$ or ∞ (see section 3.3)
Ding'10	Original SNMF algorithm [20]	We used implementation [36]
K-means	Classic K-means algorithm	We used implementation [36]

6.1 Parameter estimation on synthetic data

In the first experiment, we generate integer data according to the generative process presented in subsection 3.3 with $M = 1$ and we study the performance of Sk_1 and Sk_1-VB (see Table 1) in the ideal case where the hyperparameters used to generate data are known. In order to obtain easily interpretable and visualizable results, we decide to generate a large number ($J = 5000$) of 3-dimensional data ($I = 3$) with two components ($K = 2$). Mapping functions φ, ν and ω are chosen such that each coefficient of atoms $\boldsymbol{\theta}$ has its own hyperparameter and that activation's hyperparameters only depend on component k:

$$\left\{\theta_{si|k}\right\}_{si} \sim \text{Dirichlet}\left(\left\{\alpha_{sik}\right\}_{si}\right), \tag{54}$$

$$\lambda_{kj} \sim \text{Gamma}\left(\alpha_k, \beta_k\right). \tag{55}$$

In order to set the values of the hyperparameters, we randomly draw shapes α_{sik} and manually set shapes α_k,

according to two target levels of prior uncertainty for the parameters. Rates β_k are set such that mean value of activations is 300, leading to observations in the order of magnitude of a hundred. For each of the uncertainty level, we add a "low variance" option: activating this option corresponds to setting for each dimension i and component k $\alpha_{sik} = \epsilon$ for $s = 0$ or 1, where ϵ is some small value. Doing so insure that either $\theta_{s=0,i|k}$ or $\theta_{s=1,i|k}$ is closed to zero and thus that variance of the Skellam hidden sources $Z_{ikj} = Z_{s=0,ikj} - Z_{s=1,ikj} \sim \text{Skell}\left(\theta_{s=0,i|k}\lambda_{kj}, \theta_{s=1,i|k}\lambda_{kj}\right)$ is minimal. A low variance of the hidden sources can be interpreted as a low level of noise in the observed data. The values of the hyperparameters are summarized in Table 2.

TABLE 2
Values of hyperparameters according to the target level of prior uncertainty . If low variance option is activated, then $\min_s \alpha_{sik}$ is set to 0.02.

low uncertainty	high uncertainty
$\alpha_{sik} \in [1\ 10]$	$\alpha_{sik} \in [0.5\ 1]$
$\alpha_1 = 5, \alpha_2 = 50$	$\alpha_1 = 0.8, \alpha_2 = 0.5$
$\beta_k = \alpha_k/300$	$\beta_k = \alpha_k/300$

For each set of values for the hyperparameters, we iterate 50 times the drawing of observations according to the generative process and the running of Sk_1 and Sk_1-VB. Concerning the initialization, we set $\hat{\theta}^{(0)}_{si|k} = \alpha_{sik}/\sum_{si}\alpha_{sik}$ and $\hat{\theta}^{(0)}_{kj} = \alpha_k/(1+\beta_k)$ for Sk_1 and $\hat{\alpha}^{(0)}_{sik} = \alpha_{sik}$, and $\hat{\alpha}^{(0)}_{kj} = \alpha_k$ for Sk_1-VB. After convergence, we decide for Sk_1-VB to take the mean value of estimated posterior distributions $\hat{q}_{\theta_k}$ and $\hat{q}_{\lambda_{kj}}$ as the parameter estimates. In order to assess the quality of the estimation of both atoms and activations, we suggest to compute the mean square error (mse) on estimated parameters of the Skellam hidden sources Z_{ikj}. Besides, so we have interpretable results, we can compute separate mse for the expectation and the variance of those hidden sources:

$$\text{mse}_m = \frac{\sum_{ikj}\left(m_{ikj} - \hat{m}_{ikj}\right)^2}{I \times K \times J}, \tag{56}$$

$$\text{mse}_v = \frac{\sum_{ikj}\left(v_{ikj} - \hat{v}_{ikj}\right)^2}{I \times K \times J}, \tag{57}$$

with $m_{ikj} = W_{ik}\lambda_{kj} = \left(\theta_{s=0,i|k} - \theta_{s=1,i|k}\right)\lambda_{kj}$ and $v_{ikj} = \left(\theta_{s=0,i|k} + \theta_{s=1,i|k}\right)\lambda_{kj}$ (same definitions for $\hat{m}_{ikj}$ and $\hat{v}_{ikj}$).

Both algorithms are compared to the "dummy" algorithm consisting in taking the mean values of prior distribution as the estimation for the parameters. Results are shown in Table 3 from which several conclusions can be drawn. First of all, it can be noticed that results given by Sk_1 and Sk_1-VB are always of the same order of magnitude: at this point, we can claim that Sk_1-VB presents no significant advantage over Sk_1 for parameter estimation with known hyperparameters. Second, when the low variance option is activated, both Sk_1 and Sk_1-VB give good results with low values of mean and standard deviation. The difference with the dummy algorithm is particularly important in the high uncertainty scenario. This is expected since the greater the uncertainty on the *a priori* value of the parameters, the more crucial the observed data are in order to give a good estimation. A more surprising result is the poor quality of the parameter estimates when the low variance option is not activated. A probable explanation is that in objective functions of both Sk_1 and Sk_1-VB, the data fitting term prevails over the prior terms, meaning that these algorithms prefer minimizing the reconstruction error, *i.e.* the variance of the Skellam hidden sources, than complying to the priors.

6.2 Online hyperparameter estimation on synthetic data

In this experiment, we show that Sk_1-VB along with hyperparameter estimation can be used to perform unsupervised learning. As a proof of concept, we decide to generate a dataset $\left\{\boldsymbol{X}^1, \ldots \boldsymbol{X}^t, \ldots \boldsymbol{X}^T\right\}$ according to the same "low uncertainty, low variance" scenario as in previous subsection, with always the same hyperparameters. Contrary to the previous experiment, those hyperparameters are unknown and to be estimated. To do so, we first initialize hyperparameters for the Sk_1-VB algorithm with neutral values (all shape hyperparameters are set to 1 and rate hyperparameters are set to 0.001), and then for each data $\boldsymbol{X}^t$, we run Sk_1-VB until convergence and then update hyperparameters as described in section 5.3. In figure 1, it is showed how the mse of estimated hyperparameters with respect to the number of analyzed data is globally decreasing in a first learning stage, and then globally stable and close to 0 at convergence. It is also showed that while the estimated hyperparmeters are getting closer to the ground truth, mse of the parameters (that is mse_m and mse_v as defined in the previous subsection) are also getting better and better. This proves that our parameter estimation algorithm can automatically improves itself as data are collected.

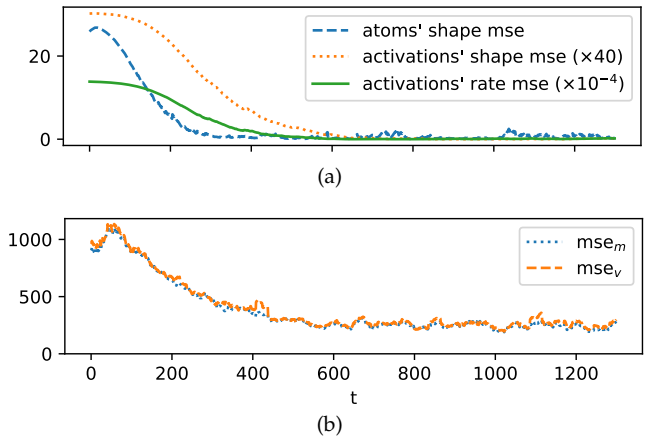


Fig. 1. Hyperparameters (a) and parameters (b) estimation error is globally decreasing with respect to the number t of analyzed data. The best permutation of components k is found before the computations of each mse. mse_m and mse_v are averaged over 30 consecutive results in order to smooth out the performance variability.

Note that we ran this experiment five times, with several random values for the ground truth hyperparameters, and those promising results were achieved only twice out of the five runs. In the other cases, the process degenerated into some vicious circle, where the more biased the hyperparameter estimates, the more biased the estimation of the posterior distributions for the parameters given a new observed

TABLE 3
Mean | standard deviation of the different metrics on 50 runs with respect to the prior uncertainty level and the low variance option

metric	algorithm	low uncertainty	low uncertainty, low variance	high uncertainty	high uncertainty, low variance
mse_m ($\times 10^3$)	Dummy	1.01 | 0.54	1.95 | 0.52	15.74 | 06.76	31.57 | 8.33
	Sk_1	**0.66** | 0.49	0.18 | 0.03	**5.99** | 12.27	0.73 | 0.97
	Sk_1-VB	0.90 | 0.68	**0.12** | 0.03	6.63 | 13.23	**0.57** | 1.34
mse_v ($\times 10^3$)	Dummy	**1.75** | 0.36	1.98 | 0.51	25.42 | 05.77	32.63 | 7.27
	Sk_1	3.09 | 2.31	0.19 | 0.05	13.82 | 13.38	**1.37** | 1.99
	Sk_1-VB	2.84 | 2.35	**0.13** | 0.06	**13.45** | 12.79	1.43 | 3.43

data, and *vice versa*, leading to quite bad results. We have not yet conducted any research in order to better understand and circumvent this issue, hence the qualification of this experience as a proof of concept. We believe however that two leads should be explored. The first one would be to supervise the VBEM algorithm in the early stages of this process. It would assure that the parameters posterior distributions are well estimated at the beginning, and then prevent the hyperparameters estimation from taking a wrong direction. The second one would be to attenuate somehow the role of the parameters prior distribution during the last iterations of each VBEM algorithm. Doing so would prevent priors to take precedence over the data in case they were too strong.

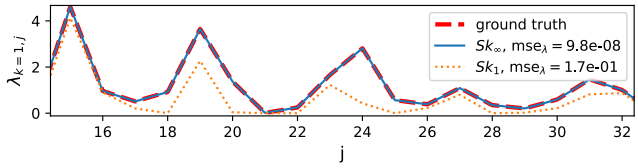
6.3 Difference between Sk_1 and Sk_∞ for SNMF

The goal of this subsection is to better understand the concrete differences between Sk_1 and Sk_∞ in SNMF problems, besides the fact that one is theoretically supposed to process only integer data and the other only real-valued data. To this aim, we generate a real matrix $\boldsymbol{X}$ as the product of a ground truth atoms matrix $\boldsymbol{W}$ and a ground truth activations matrix $\boldsymbol{\lambda}$, with no addition of noise, and we ask Sk_1 and Sk_∞ to give an estimate $\hat{\boldsymbol{\lambda}}$ of $\boldsymbol{\lambda}$, given $\boldsymbol{W}$, meaning that atoms $\boldsymbol{\theta}$ are fixed and set up to ($\propto$ is for "proportional to")

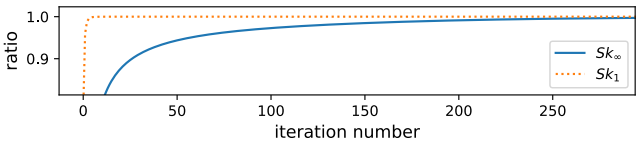
$$\theta_{si|k} \propto \frac{|W_{ik}| + (-1)^s W_{ik}}{2}. \tag{58}$$

We run our algorithms with no prior on parameters. The dimensions used to generate data are $I = 10$, $K = 3$, $J = 100$, and ground truth atoms and activations are randomly drawn. In figure 2, the estimated $\hat{\boldsymbol{\lambda}}$ compared to $\boldsymbol{\lambda}$ are plot for a given k, as well as the ratio between two consecutive values of the objective function with respect to the iteration number. Two simple conclusions can be made. The first one is that Sk_1 gives biased values for the activations – especially when value of λ_{kj} is low – while Sk_∞ provides an exact estimation. The second one is that the convergence of Sk_∞ is quite slow compared to that of Sk_1. These two characteristics can be explained by visualizing on figure 3 the shape of the objective function basic terms, which are Skellam log-likelihood $\log P_{\text{skel}}(x \mid \lambda_0, \lambda_1)$ (see equation (5)) for Sk_1 and $-\mathcal{D}(x \mid \lambda_0, \lambda_1)$ (see equation (30)) for Sk_∞. The bias in activations' estimate is due to the fact that $\log P_{\text{skel}}(x \mid \lambda_0, \lambda_1)$ gets higher when $\lambda_0 - \lambda_1$ indeed gets closed to x but also when $\lambda_0 + \lambda_1$ is minimal. On the opposite, $-\mathcal{D}(x \mid \lambda_0, \lambda_1)$ is always maximal if $x = \lambda_0 - \lambda_1$, no matter the value of $\lambda_0 + \lambda_1$. As a drawback, the shape of $\mathcal{D}(x \mid \lambda_0, \lambda_1)$ becomes rather flat when both λ_0 and λ_1 go away from 0, which can explain the slowness of the convergence. Hopefully, there exists acceleration methods for the EM algorithm. We have implemented the parabolic EM algorithm [37], which showed a drastic acceleration of the convergence.

Note that in order to emphasize the difference between the two algorithms, we drew low random values for λ_{kj}, and thus for matrix $\boldsymbol{X}$: for large values of observed data, Sk_1 tends to behave like Sk_∞ even for non-integer data.



(a) Estimated vs ground truth activations for $k = 1$.



(b) Ratio between two consecutive values of the objective function. The fastest it gets to 1, the fastest the convergence.

Fig. 2. Comparison between Sk_1 and Sk_∞ in a task of supervised SNMF.

6.4 Automatic clustering on real data

The last experiment we conduct aims at comparing our new Skellam-SNMF technique with the classic SNMF with the Euclidean distance as the objective function. We chose to do so in a simple automatic clustering task, since it is the application that has been originally proposed [20]. The idea behind using SNMF for this task is that atoms can represent the centroids of the clusters while the activations can account for the cluster membership of the observed samples. All datasets used in this experiments are from the UCI repository [38] and are composed of real-valued data. They are summarized in Table 4, and include the datasets *Ionosphere* and *Wave*, that have already been used in [20].

TABLE 4
Dataset description .

	Ionosphere	Wave	Image	Shuttle
# instances (J)	351	5000	2310	14500
# attributes (I)	34	21	19	9
# classes (K)	2	3	7	7

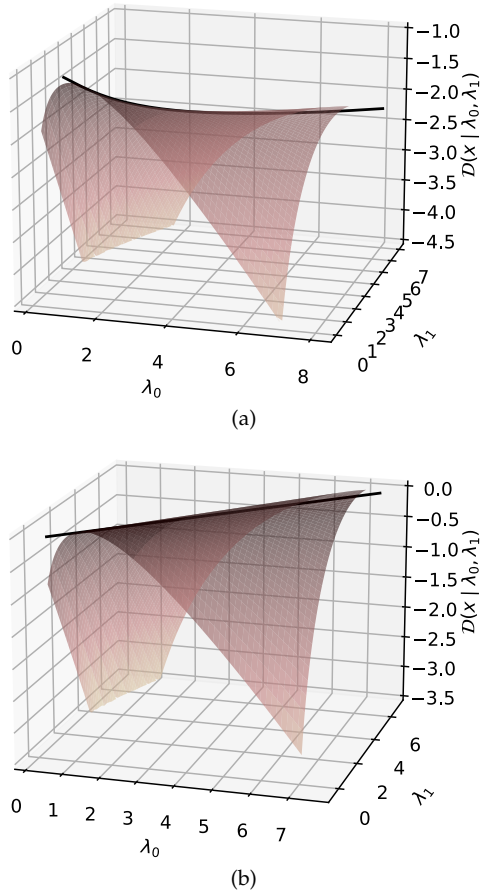


Fig. 3. 3-dimensional plots of objective function basic terms of Sk_1 (a) vs Sk_∞ (b) when observed data $x = 1$ (no prior).

We run 5 algorithms on those data: Ding'10, Sk_∞ and Sk_∞-VB with all prior shapes set to 1 and all prior rates for the activations set to 0.001 (which allows to regularize activations and to prevent them from tending towards ∞), K-means, and finally the "dummy" algorithm consisting in assigning all the samples in a single class. After convergence, the value of k for which $\hat{\lambda}_{kj}$ is maximum defines the class of sample j. As in [20], the performance measure is the clustering accuracy: first, the confusion matrix is computed, and then the columns are reordered so that the sum of the diagonal is maximal. This sum defines the metric, and represents the percentage of samples correctly clustered. Each algorithm is run 100 times, with random initialization, and mean and standard deviation are reported in Table 5. From these results, the following conclusions can be made.

TABLE 5
Mean | standard deviation of the clustering accuracy $(\%)$ on 100 runs. .

	Ionosphere	Wave	Image	Shuttle
dummy	64.1 | 0.0	33.9 | 00.0	14.3 | 0.0	**79.2** | 00.0
K-means	**70.8** | 1.6	50.2 | 00.0	**52.2** | 5.0	60.8 | 09.9
Ding'10	58.7 | 4.8	61.9 | 10.1	46.9 | 5.0	30.0 | 05.1
Sk_∞	70.6 | 0.6	**64.5** | 10.8	48.2 | 5.6	53.1 | 12.4
Sk_∞-VB	70.7 | 0.0	64.1 | 10.2	50.7 | 3.0	36.8 | 07.4

1) Sk_∞ and Sk_∞-VB always outperform Ding'10. This seems to show that the divergence $\mathcal{D}(x \mid \lambda_0, \lambda_1)$ is a good alternative to the Euclidean distance for real-valued data, the same way the Kullback-Leibler divergence is also a good alternative for nonnegative data in some applications [39]. This is our main experimental result.
2) We do not share the same conclusion as in Ding *et. al.* [20], where they find that matrix factorization models are better than K-means. On the contrary, in our experiment, K-means outperforms SNMF algorithms on 3 datasets over 4. Note that on the *Ionosphere* dataset and for the K-means algorithm, we report a mean accuracy of 70.8% when Ding *et. al.* report 42.2%. We suspect that it might be either a misprint or an error in the setting of K. Indeed, this result in addition to being very different from ours, does not make sense: in a 2 classes automatic clustering scenario, accuracy is necessarily above 50%. However, we agree that SNMF algorithms can compete with K-means.
3) In the *Shuttle* dataset, the number of sample per class is very unbalanced, with 79.2% of the samples belonging to a single class. In this case, neither K-means nor other matrix factorization models seems to be relevant as is since the dummy algorithm gives better performances. This is expected as soon as we minimize the global reconstruction error: classes with too few representatives will not affect that much the objective function.
4) Sk_∞ and Sk_∞-VB gives similar results, as in our very first experiment (subsection 6.1). This confirms the fact that VBEM algorithm does not seem to outperform EM algorithm for the parameters estimation task. Though, we observe that the standard deviation of the results is always slightly less in Sk_∞-VB than in Sk_∞ (with even a 0 standard deviation for the *Ionosphere* dataset). An explanation may be found in the role of all shape hyperparameters (set to 1 in this experiment): in the EM algorithm they play no role whatsoever in the computation of the objective function (26), whereas they do in VBEM's objective function (59). Sk_∞-VB has then an extra regularization term compared to Sk_∞ , which can reduce the variability of the results with respect to the initialization.

7 Conclusions

7.1 Main contributions

We have put forward a probabilistic model called Skellam-SNMF in order to address the SNMF problem. This model is an extension of the Poisson-NMF where the NMF with the KL divergence is interpreted as a statistical inference problem using Poisson-distributed latent sources. Skellam-SNMF is based on the Skellam distribution, and allowed us to introduce a new divergence between a real number x and two nonnegative parameters. This divergence can be interpreted as a generalization of KL divergence for real valued data. Its introduction is in our opinion the main contribution of this paper since it can be used as an alternative to the standard Euclidean distance in many other domains.

We have derived two algorithms in order to estimate the parameters of Skellam-SNMF, the EM and the VBEM algorithms, and we have also seen how to estimate the hyperparameters. It has been showed in the experiments that VBEM did not seems to give better estimates than EM given fixed hyperparameters, but could improve itself as it

analyzed new data due to online estimation of the parameters latent prior distribution. This feature can be interpreted as a way to conjugate blind data processing methods (as matrix factorizations are often considered) and automatic learning. We consider it as our second main contribution.

Finally, we have shown that Skellam-SNMF could compete with the original SNMF model with Euclidean distance in a simple task of automatic clustering. This gives then a alternative algorithm to be tested for all applications that needs SNMF.

7.2 Forthcoming work

Two features have been put forward in Skellam-SNMF without being tested due to the lack of space. The first one is the ability to deal with missing data, which could be used for data restoration applications such as inpainting or for prediction problems like movie or music recommendation. The second one is the automatic estimation of model order K which is possible by comparing the ELBO values of two competing models. This will be done in future work.

Furthermore, a generalization of Skellam-SNMF for any semi-nonnegative tensor factorization model, the same way Generalized Coupled Tensor Factorization [25] is a generalization of Poisson-NMF, is currently under publication. This generalization has already been developed and implemented in the Wonterfact package [35]. A technical report containing the underlying theory can be found in the repository of this package.

Finally, as we have already mentioned, the ability of self-improvement via the online estimation of parameters prior distribution seems very promising. We wish to further study this feature, find strategies in order to not let the process degenerate, and test it in real applications.

APPENDIX A
EXPRESSION OF ELBO

Supposing that $\mathcal{Q}$ is defined as in equations (40) to (43), the normalized ELBO can be computed as:

$$g^M\left(\mathcal{Q}, \boldsymbol{X}_{\mathcal{O}}\right) = g^M_{\boldsymbol{Z}}\left(\mathcal{Q}, \boldsymbol{X}_{\mathcal{O}}\right) + \sum_{kj} g_{\lambda_{kj}}\left(\mathcal{Q}\right) + \sum_{k} g_{\boldsymbol{\theta}_k}\left(\mathcal{Q}\right) \tag{59}$$

with (the definitions of $\bar{\ell}_{sij}$ and σ_{ij} can be found in Algorithm 2)

$$\begin{aligned} g^1_{\boldsymbol{Z}}\left(\mathcal{Q}\right) = &-\sum_{kj} \frac{\hat{\alpha}_{kj}}{\hat{\beta}_{kj}} + \sum_{ij\in\mathcal{O}} \left(\ln \frac{{}_0F_1\left(|X_{ij}|+1, \sigma_{ij}\right)}{\Gamma\left(|X_{ij}|+1\right)} \right. \\ &\left. + \sum_s \max\left(\left(-1\right)^s X_{ij}, 0\right) \ln \bar{\ell}_{sij} \right) + \sum_{ij\notin\mathcal{O}} \sum_s \bar{\ell}_{sij}, \end{aligned} \tag{60}$$

$$\begin{aligned} g^\infty_{\boldsymbol{Z}}\left(\mathcal{Q}\right) = &-\sum_{kj} \frac{\hat{\alpha}_{kj}}{\hat{\beta}_{kj}} + \sum_{ij\in\mathcal{O}} (\\ &+ |X_{ij}| \ln \left(\frac{|X_{ij}| + \sqrt{X_{ij}^2 + 4\sigma_{ij}}}{2} \right) - \sqrt{X_{ij}^2 + 4\sigma_{ij}} \\ &+ \sum_s \max\left(\left(-1\right)^s X_{ij}, 0\right) \ln \bar{\ell}_{sij} \Bigg) + \sum_{ij\notin\mathcal{O}} \sum_s \bar{\ell}_{sij}, \end{aligned} \tag{61}$$

$$\begin{aligned} g_{\lambda_{kj}}\left(\mathcal{Q}\right) = &\ \alpha_{\upsilon(k,j)} \ln \frac{\beta_{\omega(k,j)}}{\hat{\beta}_{kj}} + \hat{\alpha}_{kj} \left(1 - \frac{\beta_{\omega(k,j)}}{\hat{\beta}_{kj}}\right) \\ &- \psi\left(\hat{\alpha}_{kj}\right)\left(-\alpha_{\upsilon(k,j)}\right) - \ln \frac{\Gamma\left(\alpha_{\upsilon(k,j)}\right)}{\Gamma\left(\hat{\alpha}_{kj}\right)}, \end{aligned} \tag{62}$$

$$\begin{aligned} g_{\boldsymbol{\theta}_k}\left(\mathcal{Q}\right) = &\ \ln \frac{\Gamma\left(\sum_{si} \alpha_{\varphi(s,i,k)}\right)}{\Gamma\left(\sum_{si} \hat{\alpha}_{sik}\right)} - \sum_{si} \ln \frac{\Gamma\left(\alpha_{\varphi(s,i,k)}\right)}{\Gamma\left(\hat{\alpha}_{sik}\right)} \\ &- \sum_{si} \left(\hat{\alpha}_{sik} - \alpha_{\varphi(s,i,k)}\right) \left(\psi\left(\hat{\alpha}_{sik}\right) - \psi\left(\sum_{s'i'} \hat{\alpha}_{s'i'k}\right) \right). \end{aligned} \tag{63}$$

Detailed calculations can be found in the supplementary material.

REFERENCES

[1] P. Comon and C. Jutten, *Handbook of Blind Source Separation: Independent Component Analysis and Applications*, 1st ed. USA: Academic Press, Inc., 2010.

[2] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 11 2006.

[3] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in Neural Information Processing Systems*, no. 13, 2001, pp. 556–562.

[4] B. Rén, L. Pueyo, G. B. Zhu, J. Debes, and G. Duchêne, "Non-negative matrix factorization: Robust extraction of extended structures," p. 104, 12 2017. [Online]. Available: https://doi.org/10.3847/1538-4357/aaa1f2

[5] C. Févotte, N. Bertin, and J.-L. Durrieu, "Nonnegative matrix factorization with the Itakura-Saito divergence. With application to music analysis," *Neural Computation*, vol. 21, no. 3, pp. 793–830, 2009.

[6] L. Taslaman and B. Nilsson, "A Framework for Regularized Non-Negative Matrix Factorization, with Application to the Analysis of Gene Expression Data," *PLoS ONE*, vol. 7, no. 11, p. 46331, 11 2012. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3487913/

[7] K. Huang, X. Fu, and N. D. Sidiropoulos, "Anchor-Free Correlated Topic Modeling: Identifiability and Algorithm," in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29. Curran Associates, Inc., 2016. [Online]. Available: https://proceedings.neurips.cc/paper/2016/file/d707329bece455a462b58ce00d1194c9-Paper.pdf

[8] X. Fu, K. Huang, N. D. Sidiropoulos, and W. K. Ma, "Nonnegative Matrix Factorization for Signal and Data Analytics: Identifiability, Algorithms, and Applications," *IEEE Signal Processing Magazine*, vol. 36, no. 2, pp. 59–80, 2019.

[9] C. Févotte and J. Idier, "Algorithms for nonnegative matrix factorization with the beta-divergence," *Neural Computation*, vol. 23, no. 9, pp. 2421–2456, 2011.

[10] A. Cichocki, R. Zdunek, and S. Amari, "Csiszar's divergences for non-negative matrix factorization : Family of new algorithms," in *Proc. of LVA/ICA*, Charleston, SC, USA, 2006, pp. 32–39.

[11] F. Wang and P. Li, "Efficient nonnegative matrix factorization with random projections," in *Proceedings of the 10th SIAM International Conference on Data Mining, SDM 2010*. Society for Industrial and Applied Mathematics Publications, 2010, pp. 281–292.

[12] A. T. Cemgil, "Bayesian Inference for Nonnegative Matrix Factorisation Models," *Computational Intelligence and Neuroscience*, vol. 2009, p. 17 pages, 2009.

[13] T. Hofmann, "Unsupervised Learning by Probabilistic Latent Semantic Analysis," *Machine Learning*, vol. 42, pp. 177–196, 2001.

[14] M. Shashanka, B. Raj, and P. Smaragdis, "Probabilistic Latent Variable Models as Nonnegative Factorizations," *Computational intelligence and neuroscience*, vol. 2008, no. 4, pp. 1–8, 2008. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/18509481

[15] S. Henriet, U. Simsekli, S. D. Santos, B. Fuentes, and G. Richard, "Independent-Variation Matrix Factorization with Application to Energy Disaggregation," *IEEE Signal Processing Letters*, vol. 26, no. 11, pp. 1643–1647, 2019.

[16] Q. Qi, Y. Zhao, M. Li, and R. Simon, "Non-negative matrix factorization of gene expression profiles: A plug-in for BRB-ArrayTools," *Bioinformatics*, vol. 25, no. 4, pp. 545–547, 2 2009. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/19131367/

[17] J. Le Roux, A. de Cheveigné, and L. C. Parra, "Adaptive Template Matching with Shift-Invariant Semi-NMF," *Advances in Neural Information Processing Systems*, vol. 21, 2008.

[18] G. Trigeorgis, K. Bousmalis, S. Zafeiriou, and B. W. Schuller, "A Deep Semi-NMF model for learning hidden representations," in *31st International Conference on Machine Learning, ICML 2014*, vol. 5. PMLR, 6 2014, pp. 3677–3688. [Online]. Available: http://proceedings.mlr.press/v32/trigeorgis14.html

[19] F. Rousset, F. Peyrin, and N. Ducros, "A Semi Nonnegative Matrix Factorization Technique for Pattern Generalization in Single-Pixel Imaging," *IEEE Transactions on Computational Imaging*, vol. 4, no. 2, pp. 284–294, 3 2018.

[20] C. Ding, T. Li, and M. Jordan, "Convex and semi-nonnegative matrix factorizations." *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 1, pp. 45–55, 1 2010. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/19926898

[21] M. Chouh, M. Hanafi, and K. Boukhetala, "Semi-nonnegative rank for real matrices and its connection to the usual rank," *Linear Algebra and Its Applications*, vol. 466, pp. 27–37, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.laa.2014.09.046

[22] N. Gillis and A. Kumar, "Exact and heuristic algorithms for semi-nonnegative matrix factorization," *SIAM Journal on Matrix Analysis and Applications*, vol. 36, no. 4, pp. 1404–1424, 2015.

[23] D. W. Dreisigmeyer, "Tight Semi-nonnegative Matrix Factorization," *Pattern Recognition and Image Analysis*, vol. 30, no. 4, pp. 632–637, 2020.

[24] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, no. 4-5, pp. 993–1022, 2003.

[25] K. Y. Yilmaz, A. T. Cemgil, and U. Simsekli, "Generalized Coupled Tensor Factorization," in *NIPS*, Granada, Spain, 2011.

[26] G. J. J. Mysore and M. Sahani, "Variational Inference in Non-negative Factorial Hidden Markov Models for Efficient Audio Source Separation," in *Proc. of ICML*, Édimbourg, Écosse, 2012.

[27] B. Fuentes, R. Badeau, and G. Richard, "Harmonic Adaptive Latent Component Analysis of Audio and Application to Music Transcription," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 21, no. 9, pp. 1854–1866, 2013.

[28] A. A. Alzaid and M. A. Omair, "On the Poisson difference distribution inference and applications," *Bulletin of the Malaysian Mathematical Sciences Society*, vol. 33, no. 1, pp. 17–45, 2010.

[29] W. Gautschi and J. Slavik, "On the Computation of Modified Bessel Function Ratios," *Mathematics Of Computation*, vol. 32, no. 143, pp. 865–875, 1978.

[30] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society.*, vol. 39, no. 1, pp. 1–38, 1977. [Online]. Available: http://www.jstor.org/stable/10.2307/2984875

[31] M. Beal, "Variational Algorithms for Approximate Bayesian Inference," Ph.D. dissertation, Univ. College of London, 2003.

[32] B. E. Chérief-Abdellatif, "Consistency of ELBO maximization for model selection," *Proc. of MLR*, vol. 96, pp. 11–31, 2019.

[33] C. A. McGrory and D. M. Titterington, "Variational approximations in Bayesian model selection for finite mixture distributions," *Computational Statistics & Data Analysis*, vol. 51, no. 11, pp. 5352–5367, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167947306002362

[34] T. P. Minka, "Estimating a Dirichlet distribution," Microsoft Research Lab, Tech. Rep., 2000. [Online]. Available: http://research.microsoft.com/en-us/um/people/minka/papers/dirichlet/minka-dirichlet.pdf

[35] B. Fuentes, "wonterfact (WONderful TEnsoR FACTorization)." [Online]. Available: https://github.com/SmartImpulse/Wonterfact

[36] C. Thurau, "Python Matrix Factorization Module." [Online]. Available: https://github.com/cthurau/pymf

[37] A. F. Berlinet and C. Roland, "Acceleration of the em algorithm: P-EM versus epsilon algorithm," *Computational Statistics and Data Analysis*, vol. 56, no. 12, pp. 4122–4137, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.csda.2012.03.005

[38] D. Dua and C. Graff, "{UCI} Machine Learning Repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[39] D. FitzGerald, M. Cranitch, and E. Coyle, "On the use of the beta divergence for musical source separation," in *IET Irish Signals and Systems Conference (ISSC 2009)*, 5 2009, pp. 1–6.