Boolean Decomposition of Binary Matrices Using a Post-Nonlinear Mixture Approach ⋆

Sebastian MIRON[a,*], Mamadou DIOP[a,b], Anthony LARUE[c], Eddy ROBIN[a], David BRIE[a]

[a] *Université de Lorraine, CNRS, CRAN, F-54000 Nancy, France*
[b] *CEA Tech Grand-Est, F-57070 Metz Technopôle, France*
[c] *CEA LIST, F-91191 Gif-sur-Yvette, France.*

Abstract

We introduce a novel binary matrix factorization (BMF) approach based on a *post-nonlinear mixture model*. Unlike the existing BMF methods, which are based on the classical matrix product, the proposed mixture model is equivalent to the *Boolean* matrix factorization model when the entries of the factor matrices are exactly binary. Consequently, our approach yields interpretable results in the case of overlapping sources and more accurate low-rank binary matrix approximations compared to the state-of-the-art. We propose a simple yet efficient algorithm for solving the proposed BMF problem based on multiplicative update rules. In addition, we provide for the first time in the binary data literature, a necessary and sufficient condition for the uniqueness of the *Boolean* matrix factorization, as well as several other uniqueness results. The interest of this new approach is illustrated in numerical simulation and on real datasets.

Keywords: Binary source separation, Boolean factorization, post-nonlinear mixture, Boolean rank, uniqueness

1. Introduction

Categorical data in general, and *binary* data in particular, are increasingly present in various domains. Binary data can be either generated by *two state processes/phenomena* (and naturally encoded on the binary $\{0, 1\}$ values), *e.g.* proximity sensors, push buttons, electric switches, yes/no answers in surveys, *etc.*, or artificially obtained by *high (1-bit) quantization* of real-valued data. *1-bit quantization* and *1-bit compressive sensing* techniques have been initially developed to speed up the digital converters (see *e.g.,* [1, 2]) but they represent nowadays a highly interesting solution to the *data deluge* problem encountered in the last years in many application fields. In spectroscopy, for example, an expert may be more interested in the presence/absence of a peak in a recorded (or unmixed) spectrum at a given wavelength, rather than in the actual magnitude of this peak. Therefore, *binary matrix factorization (BMF)* (or binary matrix decomposition)

⋆This work was supported by the operational program *FEDER-FSE Lorraine et Massif des Vosges 2014-2020.*
*Corresponding author
Email address: sebastian.miron@univ-lorraine.fr (Sebastian MIRON)
URL: http://w3.cran.univ-lorraine.fr/perso/sebastian.miron/ (Sebastian MIRON)

gained a lot of interest lately and have already been successfully used in many domains such as digital telecommunications [3, 4], role mining [5], phenotype identification from gene expression data [6, 7], protein complex prediction [8], document classification [9], feature reconstruction in images [6], pattern discovery for gene expression images [10] or recommendation systems [11].

Over the last two decades, the majority of results and algorithms for BMF came from statistics, data science and machine learning communities. Meanwhile, surprisingly, two of the first algorithms for matrix factorization in binary factors are originating from signal processing [3, 4], and were developed for source separation in digital telecommunications. However, these algorithms consider the case where only one of the two factors (the source matrix) has binary $\{-1, +1\}$ entries, while the data and the other factor are complex-valued. Another early method for the decomposition of binary-valued matrices is the *Logistic Principal Components Analysis (LPCA)* [12] which uses a *probabilistic* model for the data. However, this method factorizes an element-wise probability distribution and does not produce binary factors. This algorithm served as inspiration to several other similar methods, such as the ones proposed in [13, 14, 15, 16, 17]. A problem closely related to LPCA, that concentrated some research efforts lately, is the *1-bit matrix completion*, where only a subset of the entries of the observed binary matrix is supposed known ([18, 19, 20, 21]). Using the same probabilistic framework, variants of these algorithms, with different specificities, have been proposed (see *e.g.,* [22, 23, 24, 25]).

Parallelly to the up-mentioned methods, a class of *greedy*-like approaches ([26, 27, 28, 29, 30, 31]) has been introduced for solving the BMF problem. As the binary matrix factorization problem is NP-hard [29], these methods employ various strategies to iteratively estimate and extract the binary rank-1 terms (sources) from the observation matrix.

Another way to tackle the NP-hardness of the BMF problem is its *relaxation over the real nonnegative orthant* ([32, 7, 33]). This way, the discrete problem is casted into a continuous optimization problem, with constraints forcing the "binarity" of the results.

A family of methods related to BMF is formed by the *co-clustering* (or *block clustering*) algorithms for binary data. The co-clustering aims at partitioning an objects-attributes matrix in low-rank homogenous structures; these low-rank binary structures are sought using BMF-like approaches (see *e.g.,* [34, 35, 36, 37]).

While the BMF methods mentioned above use fundamentally different algorithmic approaches, they are all based on a mixture model defined using the classical real-matrix product. This implies (as explained in the following section) that these decompositions have the tendency to generate "decorrelated" factors (sources), as they strongly penalize the factors having overlapping supports. This represents a major drawback in a large class of applications where common behaviors of different "populations" are sought. Moreover, by forcing the non-overlapping of the factors, these methods yield an increased rank of the BMF decomposition, which is an undesired feature in low-rank approximation and data compression applications.

Contributions: In this paper we take a fresh look at the *binary matrix factorization* from a signal processing (*source separation*) perspective. We propose a BMF approach based on a relaxation over the real field of the *Boolean matrix product*. Unlike the approach in [32], by using a *post-nonlinear mixture model*, the binary matrix product used by our method is equivalent to the *Boolean matrix product* in the exact binary case, and therefore, the correlation of the underlaying factors does no longer affect the algorithm performance. Another major contribution of this work is represented by the results on the uniqueness of the *Boolean* decomposition of a binary matrix; we provide a necessary condition, a sufficient condition and a necessary and sufficient condition for the uniqueness of the *Boolean* decomposition of a binary matrix. As far as we know, it is the first that this type of conditions are provided in the binary data literature. The performance of the proposed algorithm is illustrated on synthetic and real data, and compared with similar state-of-the-art algorithms.

The paper is organized as follows. In Section 2 we illustrate and discuss the binary matrix decompositions models and their features. Section 3 introduces the uniqueness conditions for the *Boolean* matrix decomposition.The proposed *post-nonlinear mixture BMF approach* is introduced in Section 4 and is tested on synthetic and real data in Section 5. The last section, *i.e.,* Section 6, presents the conclusions of this work.

Notation: A bold-face capital letter $\boldsymbol{X}$ denotes a matrix, vectors are written in bold-face lower-case $\boldsymbol{x}$ and $\boldsymbol{x}_k$ indicates the kth column of matrix $\boldsymbol{X}$. Scalars are lower-case x or upper-case X. The element on the ith column and the jth row of matrix $\boldsymbol{X}$ is denoted $\boldsymbol{X}_{ij}$. $\boldsymbol{X}^{(k)}$ denotes the kth rank-1 term in the decomposition of $\boldsymbol{X}$. The all-ones matrix of size $N \times M$ is symbolized by $\mathbf{1}_{N\times M}$. The Frobenius norm of a matrix is denoted $\|\cdot\|_{\mathsf{F}}$, $(.)^{\mathsf{T}}$ is the matrix transposition operator and "$*$" denotes the Hadamard (element-wise) product of two matrices; $|\cdot|$ symbolizes the cardinality of a set.

2. Binary Matrix Decompositions (Factorizations)

The decomposition of a $N \times M$ matrix $\boldsymbol{X}$ with binary entries ($\boldsymbol{X}_{ij} \in \{0, 1\}$), into a sum of K (generally $K \ll \min\{M, N\}$) rank-1 terms:

$$\boldsymbol{X} = \boldsymbol{X}^{(1)} + \cdots + \boldsymbol{X}^{(K)}, \tag{1}$$

with $\boldsymbol{X}^{(k)} = \boldsymbol{w}_k \boldsymbol{h}_k{}^{\mathsf{T}}$, $(k = 1, \ldots, K)$ is generally known as *Binary Matrix Decomposition* or *Binary Matrix Factorization (BMF)*. The vectors $\boldsymbol{w}_k$ and $\boldsymbol{h}_k$ are oftenly termed as *factors*. The smallest K for which (1) holds exactly is called the *rank* of the decomposition.

If we adopt a *source separation* point of view, $\boldsymbol{X}$ represents the observed source mixtures (the measurements), while the rank-1 matrix $\boldsymbol{X}^{(k)}$ is the kth source, with the two dimensions corresponding to the two observed diversities; K is the number of sources that we seek to separate from the mixture.

Depending on the set over which this factorization is performed, several BMFs and decomposition ranks can be defined:

- If the entries of $\boldsymbol{w}_k$ and $\boldsymbol{h}_k$ are *real* values, the rank of the decomposition (1) is the familiar *real rank*, denoted by $\text{rank}_{\mathbb{R}}\{\boldsymbol{X}\}$. The *real BMF* of $\boldsymbol{X}$ as well as the *real rank* can be easily obtained by well-known algebraic methods such as the *Singular Value Decomposition (SVD)*. While this decomposition may seem algorithmically very appealing, it does not preserve binarity and therefore its results are difficult to interpret in binary data applications.

- If the entries of $\boldsymbol{w}_k$ and $\boldsymbol{h}_k$ are constraint to be *real nonnegative*, the BMF (1) boils down to the *Nonnegative Matrix Factorization (NMF)* [38], and its rank is the *nonnegative rank*, $\text{rank}_{\mathbb{R}_+}\{\boldsymbol{X}\}$. Many algorithms exist for the computation of the NMF decomposition of $\boldsymbol{X}$, *e.g.,* [39]. This decomposition is not well-adapted either to analyze binary data (although it has already been used to this end), as it does not preserve binarity of the results.

- If the entries of $\boldsymbol{w}_k$ and $\boldsymbol{h}_k$ are restricted to the $\{0,1\}$ set, the decomposition of (1) is generally known as the *binary* decomposition of $\boldsymbol{X}$ and the associated rank is the *binary rank* (see *e.g.*, [40]), denoted $\text{rank}_{\{0,1\}}\{\boldsymbol{X}\}$. The *binary rank* can be interpreted as the minimum number of *disjoint* all-ones (rank-1) rectangles needed to cover all the 1's of $\boldsymbol{X}$ (after some row and column permutations).

- If we keep the restriction of $\boldsymbol{w}_k$ and $\boldsymbol{h}_k$ to the $\{0,1\}$ set and replace the "+" in (1) by the logical OR operator "$\vee$", than eq. (1) can be seen as a decomposition of $\boldsymbol{X}$ over the *Boolean semiring* $(\mathbb{B}, \vee, \wedge)$. The symbol "$\wedge$" denotes the AND logical operator, which is the same as the real multiplication for binary values. This decomposition is oftenly termed as the *Boolean* factorization of the binary matrix $\boldsymbol{X}$, and the rank of the decomposition is known as the *Boolean rank* [41] or the *Schein rank* [42], and denoted $\text{rank}_{\mathbb{B}}\{\boldsymbol{X}\}$. The *Boolean rank* of $\boldsymbol{X}$ can be understood as the minimum number of all-ones (*not necessarily disjoint*) rectangles needed to cover all the 1's of $\boldsymbol{X}$ (after some row and column permutations). The problem of determining the *Boolean rank* of a binary matrix is NP-complete [43].

Table 1 summarizes the different decompositions of a binary matrix presented above.

Set of values for $\boldsymbol{w}_k$, $\boldsymbol{h}_k$	*sum* operator	Decomposition	rank
real field ($\mathbb{R}$)	real sum ("+")	*Real BMF*	$\text{rank}_{\mathbb{R}}$
real nonnegative ($\mathbb{R}_+$)	real sum ("+")	*Nonnegative Matrix Factorization (NMF)*	$\text{rank}_{\mathbb{R}_+}$
{0,1}	real sum ("+")	*Binary Decomposition*	$\text{rank}_{\{0,1\}}$
{0,1}	logical OR ("$\vee$")	*Boolean Decomposition*	$\text{rank}_{\mathbb{B}}$

Table 1: Binary matrix decompositions

It is known that the *Boolean rank* of a binary matrix is a lower bound for the other nonnegative ranks [40], but to the best our knowledge there is no proven relationship between the *real rank* and the *Boolean rank*:

$$\text{rank}_{\mathbb{B}}\{\boldsymbol{X}\} \leq \text{rank}_{\mathbb{R}_+}\{\boldsymbol{X}\} \leq \text{rank}_{\{0,1\}}\{\boldsymbol{X}\}, \tag{2}$$

$$\text{rank}_{\mathbb{R}}\{\boldsymbol{X}\} \leq \text{rank}_{\mathbb{R}_+}\{\boldsymbol{X}\}. \tag{3}$$

We propose next two toy examples to illustrate the the difference between the *binary* and the *Boolean* ranks.

Consider the 3×3 binary matrix of $\text{rank}_{\mathbb{R}}$ equal to 3:

$$\boldsymbol{X} = \begin{pmatrix} \mathbf{1} & \mathbf{1} & 0 \\ \mathbf{1} & \mathbf{1} & \mathbf{1} \\ 0 & \mathbf{1} & \mathbf{1} \end{pmatrix}. \tag{4}$$

The *binary rank* of $\boldsymbol{X}$ is also $\text{rank}_{\{0,1\}}\{\boldsymbol{X}\} = 3$ and two possible *binary* decompositions are:

$$\boldsymbol{X} = \underbrace{\begin{pmatrix} \mathbf{1} & 0 & 0 \\ \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}}_{\boldsymbol{X}^{(1)}} + \underbrace{\begin{pmatrix} 0 & \mathbf{1} & 0 \\ 0 & \mathbf{1} & 0 \\ 0 & \mathbf{1} & 0 \end{pmatrix}}_{\boldsymbol{X}^{(2)}} + \underbrace{\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} \\ 0 & 0 & \mathbf{1} \end{pmatrix}}_{\boldsymbol{X}^{(3)}} = \underbrace{\begin{pmatrix} \mathbf{1} & \mathbf{1} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}}_{\boldsymbol{X}^{(1)}} + \underbrace{\begin{pmatrix} 0 & 0 & 0 \\ \mathbf{1} & \mathbf{1} & \mathbf{1} \\ 0 & 0 & 0 \end{pmatrix}}_{\boldsymbol{X}^{(2)}} + \underbrace{\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & \mathbf{1} & \mathbf{1} \end{pmatrix}}_{\boldsymbol{X}^{(3)}}.$$

This decomposition preserves the binary nature of the results, but it suffers from *non-uniquness*.

By replacing the arithmetical sum by the logical disjunction "$\vee$" the following *Boolean* decomposition of $\boldsymbol{X}$ is obtained:

$$\boldsymbol{X} = \underbrace{\begin{pmatrix} \mathbf{1} & \mathbf{1} & 0 \\ \mathbf{1} & \mathbf{1} & 0 \\ 0 & 0 & 0 \end{pmatrix}}_{\boldsymbol{X}^{(1)}} \vee \underbrace{\begin{pmatrix} 0 & 0 & 0 \\ 0 & \mathbf{1} & \mathbf{1} \\ 0 & \mathbf{1} & \mathbf{1} \end{pmatrix}}_{\boldsymbol{X}^{(2)}}.$$

The *Boolean rank* of this decomposition is $\text{rank}_{\mathbb{B}}\{\boldsymbol{X}\} = 2$. Moreover, this decomposition seems to be unique *unique*, *i.e.* we could not find another rank-2 decomposition that allows to recover $\boldsymbol{X}$ exactly.

In the example considered above it can be seen that the *Boolean decomposition* of a binary matrix seems to have more interesting features compared to the *binary* decompositions, such as *lower rank* and *uniqueness*. This makes it an interesting tool for binary data analysis and binary data compression applications. Nevertheless, at this point it is not clear whether these advantages hold in general or only in some particular cases. The next example brings more insights into these aspects.

Consider the 4×4 binary matrix:

$$\boldsymbol{X} = \begin{pmatrix} \mathbf{1} & \mathbf{1} & 0 & 0 \\ \mathbf{1} & \mathbf{1} & 0 & 0 \\ 0 & 0 & \mathbf{1} & \mathbf{1} \\ 0 & 0 & \mathbf{1} & \mathbf{1} \end{pmatrix}. \tag{5}$$

The decomposition of $\boldsymbol{X}$ in a minimum number of rank-1 terms can be expressed as:

$$\boldsymbol{X} = \underbrace{\begin{pmatrix} \mathbf{1} & \mathbf{1} & 0 & 0 \\ \mathbf{1} & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}}_{\boldsymbol{X}^{(1)}} + \underbrace{\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & \mathbf{1} \\ 0 & 0 & \mathbf{1} & \mathbf{1} \end{pmatrix}}_{\boldsymbol{X}^{(2)}} = \underbrace{\begin{pmatrix} \mathbf{1} & \mathbf{1} & 0 & 0 \\ \mathbf{1} & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}}_{\boldsymbol{X}^{(1)}} \vee \underbrace{\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & \mathbf{1} \\ 0 & 0 & \mathbf{1} & \mathbf{1} \end{pmatrix}}_{\boldsymbol{X}^{(2)}}$$

In this case, all the decompositions are equivalent and we have $\text{rank}_{\mathbb{B}}\{\boldsymbol{X}\} = \text{rank}_{\mathbb{R}}\{\boldsymbol{X}\} = \text{rank}_{\mathbb{R}_+}\{\boldsymbol{X}\} = \text{rank}_{\{0,1\}}\{\boldsymbol{X}\} = 2$. Moreover, the two decompositions that preserve the binary nature of the data, *i.e.,* the *binary* and the *Boolean* decompositions, are unique (according to uniqueness results presented in Section 3).

If we take a closer look at these two examples, we see that the *Boolean* decomposition is advantageous over the *binary* decomposition when the rank-1 terms $\boldsymbol{X}^{(k)}$ have overlapping supports[1], *i.e.,* in the case of *correlated* sources[2], as illustrated by the first example. When the sources are *uncorrelated*, *i.e.,* the supports of the rank-1 terms are disjoint (this is the case for the second example), the *binary* and the *Boolean* decompositions are equivalent.

We illustrate next, on a practical example, the utility of the *Boolean* factorization in data analysis. We consider a subset of the `UCI zoo dataset`[3], composed of a list of five animal species and five *Boolean*-valued attributes, as shown in Table 2.

	airborne	backbone	toothed	aquatic	milk
crow	**1**	**1**	0	0	0
hawk	**1**	**1**	0	0	0
carp	0	**1**	**1**	**1**	0
dolphin	0	**1**	**1**	**1**	**1**
elephant	0	**1**	**1**	0	**1**

Table 2: The considered subset of the UCI zoo dataset. Each row corresponds to a species and each column to an attribute.

The presence/absence of an attribute for a species is encoded by "1"/ "0", respectively. This results in a 5×5 binary-valued matrix $\boldsymbol{X}$, as shown in Table 2. The objective is to find classes of similar species, *i.e.,* species sharing the same attributes. This comes down to decomposing the binary attributes matrix into a sum of rank-1 binary terms $\boldsymbol{X}^{(k)} = \boldsymbol{w}_k \boldsymbol{h}_k^{\mathsf{T}}$. For each rank-1 term, $\boldsymbol{w}_k$ will have "1"s in the positions corresponding to the similar species and $\boldsymbol{h}_i$ will have "1"s in the positions corresponding to the shared attributes. For the data considered here, we obtain a

[1]We define the support of a vector $\boldsymbol{x}$ as $\text{supp}\{\boldsymbol{x}\} = \{i, \boldsymbol{x}_i \neq 0\}$ and the support of matrix $\boldsymbol{X}$ as $\text{supp}\{\boldsymbol{X}\} = \{(i,j), \boldsymbol{X}_{ij} \neq 0\}$.

[2]The correlation of two binary sources $\boldsymbol{X}^{(i)}$ and $\boldsymbol{X}^{(j)}$ is defined as the cardinality of the intersection of their supports, *i.e.*, $\text{corr}\{\boldsymbol{X}^{(i)}, \boldsymbol{X}^{(j)}\} = \mid \text{supp}\{\boldsymbol{X}^{(i)}\} \cap \text{supp}\{\boldsymbol{X}^{(j)}\} \mid$.

[3]Available on the UCI Machine learning repository website: `https://archive.ics.uci.edu/ml/datasets/Zoo` .

rank-3 *Boolean* decomposition; the three rank-1 terms are highlighted in Table 2. Interestingly, one can see that, according to the decomposition, the dolphin belongs to two classes, as it has common features both with the `carp` and the `elephant`. This a reasonable result, as the dolphin is an aquatic mammal. This result is possible thanks to the fact that the *Boolean* decomposition allows overlapping supports of the rank-1 terms, as illustrated in Table 2.

The examples above illustrate the fact that the *Boolean* factorization of a binary matrix has interesting properties in terms of *rank* and *uniqueness*, which makes it an interesting tool for binary data analysis and signal processing applications. In this paper we focus on this particular decomposition, study its uniqueness and propose an efficient algorithm to achieve the *Boolean* BMF.

3. Uniqueness of the *Boolean* BMF

In the previous section we have seen that *Boolean* decomposition seems to present interesting uniqueness features. We prove in this section a *necessary and sufficient* condition for the unique *Boolean* decomposition of a binary matrix, and some other uniqueness results. Consider the *Boolean decomposition* of a binary matrix $\boldsymbol{X}$ given by:

$$\boldsymbol{X} = \boldsymbol{X}^{(1)} \vee \cdots \vee \boldsymbol{X}^{(K)} = \bigvee_{k=1}^{K} \boldsymbol{X}^{(k)}, \tag{6}$$

with $\boldsymbol{X}^{(k)} = \boldsymbol{w}_k \boldsymbol{h}_k^{\mathsf{T}}$, and the entries of $\boldsymbol{w}_k$ and $\boldsymbol{h}_k$ restricted to the set $\{0, 1\}$.

Definition 3.1 (Full uniqueness)**.** *We say that the decomposition* (6) *is* fully unique *if, for any other set of K rank-1 binary matrices $\left\{\bar{\boldsymbol{X}}^{(1)}, \cdots, \bar{\boldsymbol{X}}^{(K)}\right\}$ satisfying* (6)*, we have $\boldsymbol{X}^{(k)} = \bar{\boldsymbol{X}}^{(k)}$, for all $k = 1, \cdots, K$, up to some permutation of the superscripts $(1, \cdots, K)$.*

This permutation of the superscripts is also known in the literature as the *order* indeterminacy. It is worth noting that, for the decomposition in binary terms there is no *scaling* indeterminacy, contrary to the real-value decompositions.

The condition for the full uniqueness of (6) as well as its proof, is based on an intermediary and more relaxed notion which is the *partial uniqueness*. Let us consider the *Boolean* decomposition of the rank-K binary matrix $\boldsymbol{X}$ given by (6), and re-write it as:

$$\boldsymbol{X} = \bigvee_{k \neq i} \boldsymbol{X}^{(k)} \vee \boldsymbol{X}^{(i)} = \boldsymbol{X}^{(\setminus i)} \vee \boldsymbol{X}^{(i)}, \tag{7}$$

where $\boldsymbol{X}^{(\setminus i)} = \bigvee_{k \neq i} \boldsymbol{X}^{(k)}$ is a binary matrix of *Boolean* rank $K - 1$.

Definition 3.2 (Partial uniqueness)**.** *We say that the decomposition* (7) *is* partially unique *with respect to $\boldsymbol{X}^{(i)}$ if, for a given $\boldsymbol{X}^{(\setminus i)}$, the only rank-1 binary matrix satisfying* (7) *is $\boldsymbol{X}^{(i)}$.*

We study next under which conditions the partial uniqueness of a *Boolean* decomposition can be achieved. First we prove a simple yet insightful *necessary* condition for partial uniqueness.

Theorem 3.1 (Partial uniqueness: a necessary condition)**.** *The decomposition* (7) *is not partially unique with respect to* $\boldsymbol{X}^{(i)} = \boldsymbol{w}_i\boldsymbol{h}_i^\mathsf{T}$ *if there exists* $j \in \{1, \cdots, K\} \setminus \{i\}$ *such that* $\text{supp}\{\boldsymbol{w}_i\} \subseteq \text{supp}\{\boldsymbol{w}_j\}$ *or* $\text{supp}\{\boldsymbol{h}_i\} \subseteq \text{supp}\{\boldsymbol{h}_j\}$.

Proof. Let us consider the case $\text{supp}\{\boldsymbol{w}_i\} \subseteq \text{supp}\{\boldsymbol{w}_j\}$ (the proof for the case $\text{supp}\{\boldsymbol{h}_i\} \subseteq \text{supp}\{\boldsymbol{h}_j\}$ is similar). If, at the same time, $\text{supp}\{\boldsymbol{h}_i\} \subseteq \text{supp}\{\boldsymbol{h}_j\}$, then $\text{supp}\{\boldsymbol{X}^{(i)}\} \subseteq \text{supp}\{\boldsymbol{X}^{(j)}\}$, meaning that $\boldsymbol{X}^{(i)} \vee \boldsymbol{X}^{(j)} = \boldsymbol{X}^{(j)}$ and consequently $\boldsymbol{X}$ is of rank $K-1$, which contradicts the assumption. If $\text{supp}\{\boldsymbol{h}_i\} \nsubseteq \text{supp}\{\boldsymbol{h}_j\}$, then $\boldsymbol{h}_i$ can be expressed as $\boldsymbol{h}_i = \boldsymbol{h}_i^{\subseteq} \vee \boldsymbol{h}_i^{\nsubseteq}$, with $\text{supp}\{\boldsymbol{h}_i^{\subseteq}\} \subseteq \text{supp}\{\boldsymbol{h}_j\}$, $\text{supp}\{\boldsymbol{h}_i^{\nsubseteq}\} \nsubseteq \text{supp}\{\boldsymbol{h}_j\}$ and $\text{supp}\{\boldsymbol{h}_i^{\nsubseteq}\} \neq \emptyset$. Let $\boldsymbol{h}$ be a $M \times 1$ binary vector such that $\text{supp}\{\boldsymbol{h}\} \subseteq \text{supp}\{\boldsymbol{h}_j\}$ and $\boldsymbol{h} \neq \boldsymbol{h}_i^{\subseteq}$. Decomposition (7) can then be expressed as $\boldsymbol{X} = \bigvee_{k\neq i,j} \boldsymbol{X}^{(k)} \vee \boldsymbol{X}^{(j)} \vee \boldsymbol{X}^{(i)} = \bigvee_{k\neq i,j} \boldsymbol{X}^{(k)} \vee \boldsymbol{w}_j\boldsymbol{h}_j^\mathsf{T} \vee \boldsymbol{w}_i(\boldsymbol{h}_i^{\subseteq} \vee \boldsymbol{h}_i^{\nsubseteq})^\mathsf{T} = \bigvee_{k\neq i,j} \boldsymbol{X}^{(k)} \vee \boldsymbol{w}_j\boldsymbol{h}_j^\mathsf{T} \vee \boldsymbol{w}_i(\boldsymbol{h}_i^{\subseteq})^\mathsf{T} \vee \boldsymbol{w}_i(\boldsymbol{h}_i^{\nsubseteq})^\mathsf{T}$. Under the given assumptions, $\boldsymbol{w}_j\boldsymbol{h}_j^\mathsf{T} \vee \boldsymbol{w}_i(\boldsymbol{h}_i^{\subseteq})^\mathsf{T} = \boldsymbol{w}_j\boldsymbol{h}_j^\mathsf{T}$ and $\boldsymbol{w}_j\boldsymbol{h}_j^\mathsf{T} = \boldsymbol{w}_j\boldsymbol{h}_j^\mathsf{T} \vee \boldsymbol{w}_i\boldsymbol{h}^\mathsf{T}$, which, after substitution in the considered decomposition and some simple algebraic manipulations, yields $\boldsymbol{X} = \bigvee_{k\neq i} \boldsymbol{X}^{(k)} \vee \boldsymbol{w}_i(\boldsymbol{h}_i^{\nsubseteq} \vee \boldsymbol{h})^\mathsf{T} = \bigvee_{k\neq i} \boldsymbol{X}^{(k)} \vee \bar{\boldsymbol{X}}^{(i)}$, with $\bar{\boldsymbol{X}}^{(i)} = \boldsymbol{w}_i(\boldsymbol{h}_i^{\nsubseteq} \vee \boldsymbol{h})^\mathsf{T}$. As $\boldsymbol{h} \neq \boldsymbol{h}_i^{\subseteq}$ and $\text{supp}\{\boldsymbol{h}_i^{\nsubseteq}\} \cap \text{supp}\{\boldsymbol{h}\} = \emptyset$, it means that $\boldsymbol{X}^{(i)} \neq \bar{\boldsymbol{X}}^{(i)}$, *i.e,* decomposition is not partially unique.

□

An illustration of this necessary condition is given on Figure 1. The gray and the dark gray rectangles represent the supports of $\boldsymbol{X}^{(j)}$ and $\boldsymbol{X}^{(i)}$, respectively, as defined in Theorem 3.1. One can observe that $\text{supp}\{\boldsymbol{w}_i\} \subset \text{supp}\{\boldsymbol{w}_j\}$. The dashed lines represent the bounds of three different possible support configurations for the rank-1 term $\boldsymbol{X}^{(i)}$, illustrating the non-uniqueness of this term in the decomposition. A key point for understanding the uniqueness results presented in this section is the fact that all these admissible configurations of $\boldsymbol{X}^{(i)}$ include the rank-1 term $\boldsymbol{w}_i(\boldsymbol{h}_i^{\nsubseteq})^\mathsf{T}$, which is outside the support of $\boldsymbol{X}^{(j)}$. This term represents the $\boldsymbol{X}^{(i)}$ with the minimum support, satisfying (7).

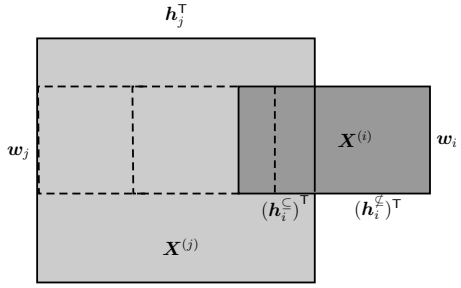


Figure 1: Illustration of the necessary condition for partial uniqueness given by Theorem 3.1

Based on these observations, we prove next a necessary and sufficient condition for partial uniqueness.

Theorem 3.2 (Partial uniqueness: a necessary and sufficient condition)**.** *Consider the decomposition* (7) *and define the following subsets:*

- $\Omega_{\boldsymbol{w}_i}$ *: the smallest subset of* $\{1, \cdots, K\} \setminus \{i\}$ *such that* $\text{supp}\{\boldsymbol{w}_i\} \subseteq \bigcup_{k \in \Omega_{\boldsymbol{w}_i}} \text{supp}\{\boldsymbol{w}_k\}$
- $\Omega_{\boldsymbol{h}_i}$ *: the smallest subset of* $\{1, \cdots, K\} \setminus \{i\}$ *such that* $\text{supp}\{\boldsymbol{h}_i\} \subseteq \bigcup_{k \in \Omega_{\boldsymbol{h}_i}} \text{supp}\{\boldsymbol{h}_k\}$

The decomposition is partially unique with respect to $\boldsymbol{X}^{(i)} = \boldsymbol{w}_i \boldsymbol{h}_i^\mathsf{T}$ *iff:* $\bigcap_{k \in \Omega_{\boldsymbol{h}_i}} \text{supp}\{\boldsymbol{w}_k\} = \emptyset$ *and* $\bigcap_{k \in \Omega_{\boldsymbol{w}_i}} \text{supp}\{\boldsymbol{h}_k\} = \emptyset$.

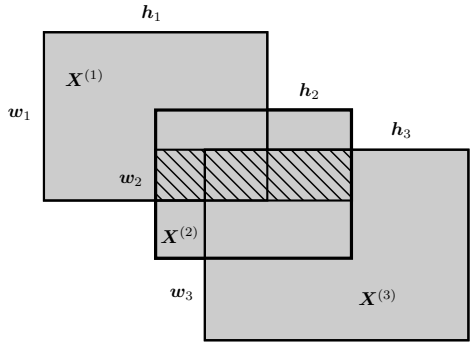
Proof. The sufficient condition. Suppose that $\bigcap_{k \in \Omega_{\boldsymbol{h}_i}} \text{supp}\{\boldsymbol{w}_k\} \neq \emptyset$ and let $\boldsymbol{w}^\cap$ be a $N \times 1$ binary vector such that: $\text{supp}\{\boldsymbol{w}^\cap\} = \bigcap_{k \in \Omega_{\boldsymbol{h}_i}} \text{supp}\{\boldsymbol{w}_k\} \cap \text{supp}\{\boldsymbol{w}_i\}$. Then, $\boldsymbol{w}_i$ can be expressed as $\boldsymbol{w}_i = \boldsymbol{w}_i^\cap \vee \boldsymbol{w}_i^*$, where $\boldsymbol{w}_i^\cap$ and $\boldsymbol{w}_i^*$ have disjoint supports. As $\text{supp}\{\boldsymbol{h}_i\} \subseteq \bigcup_{k \in \Omega_{\boldsymbol{h}_i}} \text{supp}\{\boldsymbol{h}_k\}$ it implies $\text{supp}\{\boldsymbol{w}_i^*\} \neq \emptyset$ (otherwise $\text{supp}\{\boldsymbol{w}_i \boldsymbol{h}_i^\mathsf{T}\} \subseteq \bigcup_{k \in \Omega_{\boldsymbol{h}_i}} \text{supp}\{\boldsymbol{w}_k \boldsymbol{h}_k^\mathsf{T}\}$, resulting in a rank loss of $\boldsymbol{X}$). Now, let us write $\boldsymbol{X} = \boldsymbol{X}^{(\setminus i)} \vee \boldsymbol{X}^{(i)} = \bigvee_{k \notin \Omega_{\boldsymbol{h}_i}} \boldsymbol{X}^{(k)} \vee \bigvee_{k \in \Omega_{\boldsymbol{h}_i}} \boldsymbol{X}^{(k)} \vee \boldsymbol{X}^{(i)} = \bigvee_{k \notin \Omega_{\boldsymbol{h}_i}} \boldsymbol{X}^{(k)} \vee \bigvee_{k \in \Omega_{\boldsymbol{h}_i}} \boldsymbol{w}_k \boldsymbol{h}_k^\mathsf{T} \vee (\boldsymbol{w}_i^\cap \vee \boldsymbol{w}_i^*) \boldsymbol{h}_i^\mathsf{T}$. Let $\boldsymbol{w}$ be a $N \times 1$ binary vector such that $\text{supp}\{\boldsymbol{w}\} \subset \text{supp}\{\boldsymbol{w}_i^\cap\}$, which implies $\boldsymbol{w}_i^\cap \vee \boldsymbol{w} = \boldsymbol{w}_i^\cap$. Then we have $\boldsymbol{X} = \bigvee_{k \notin \Omega_{\boldsymbol{h}_i}} \boldsymbol{X}^{(k)} \vee \bigvee_{k \in \Omega_{\boldsymbol{h}_i}} \boldsymbol{w}_k \boldsymbol{h}_k^\mathsf{T} \vee (\boldsymbol{w}_i^\cap \vee \boldsymbol{w} \vee \boldsymbol{w}_i^*) \boldsymbol{h}_i^\mathsf{T} = \bigvee_{k \notin \Omega_{\boldsymbol{h}_i}} \boldsymbol{X}^{(k)} \vee \bigvee_{k \in \Omega_{\boldsymbol{h}_i}} \boldsymbol{w}_k \boldsymbol{h}_k^\mathsf{T} \vee \boldsymbol{w}_i^\cap \boldsymbol{h}_i^\mathsf{T} \vee (\boldsymbol{w} \vee \boldsymbol{w}_i^*) \boldsymbol{h}_i^\mathsf{T} = \bigvee_{k \notin \Omega_{\boldsymbol{h}_i}} \boldsymbol{X}^{(k)} \vee \bigvee_{k \in \Omega_{\boldsymbol{h}_i}} \boldsymbol{w}_k \boldsymbol{h}_k^\mathsf{T} \vee (\boldsymbol{w} \vee \boldsymbol{w}_i^*) \boldsymbol{h}_i^\mathsf{T} = \bigvee_{k \neq i} \boldsymbol{X}^{(k)} \vee \bar{\boldsymbol{X}}^{(i)}$, with $\bar{\boldsymbol{X}}^{(i)} = (\boldsymbol{w} \vee \boldsymbol{w}_i^*) \boldsymbol{h}_i^\mathsf{T}$. As $\text{supp}\{\boldsymbol{w}\} \subset \text{supp}\{\boldsymbol{w}_i^\cap\}$, it means that $\boldsymbol{X}^{(i)} \neq \bar{\boldsymbol{X}}^{(i)}$, *i.e,* the decomposition is not partially unique, which contradicts the assumption.

The necessary condition. Suppose that the decomposition $\boldsymbol{X} = \boldsymbol{X}^{(\setminus i)} \vee \boldsymbol{X}^{(i)}$ is not partially unique, *i.e.,* there exists a rank-1 binary matrix $\bar{\boldsymbol{X}}^{(i)} = \bar{\boldsymbol{w}}_i \bar{\boldsymbol{h}}_i^\mathsf{T}$, $\bar{\boldsymbol{X}}^{(i)} \neq \boldsymbol{X}^{(i)}$ such that: $\boldsymbol{X} = \boldsymbol{X}^{(\setminus i)} \vee \boldsymbol{X}^{(i)} = \boldsymbol{X}^{(\setminus i)} \vee \bar{\boldsymbol{X}}^{(i)}$. $\bar{\boldsymbol{X}}^{(i)} \neq \boldsymbol{X}^{(i)}$ implies $\bar{\boldsymbol{w}}_i \neq \boldsymbol{w}_i$ or $\bar{\boldsymbol{h}}_i \neq \boldsymbol{h}_i$. To preserve the rank K of $\boldsymbol{X}$, the following non-inclusion relations must be satisfied: $\boldsymbol{X}^{(i)} \nsubseteq \boldsymbol{X}$ and $\bar{\boldsymbol{X}}^{(i)} \nsubseteq \boldsymbol{X}$. This means that we can find a rank-1 term $\boldsymbol{w}_i^* \boldsymbol{h}_i^{*\mathsf{T}}$ with $\text{supp}\{\boldsymbol{w}_i^* \boldsymbol{h}_i^{*\mathsf{T}}\} \nsubseteq \text{supp}\{\boldsymbol{X}^{(\setminus i)}\}$, whose support is included in both $\boldsymbol{X}^{(i)}$ and $\bar{\boldsymbol{X}}^{(i)}$. Thus, we can write $\boldsymbol{w}_i = \boldsymbol{w}_i^* \vee \boldsymbol{w}_i^\star$, $\bar{\boldsymbol{w}}_i = \boldsymbol{w}_i^* \vee \bar{\boldsymbol{w}}_i^\star$, $\boldsymbol{h}_i = \boldsymbol{h}_i^* \vee \boldsymbol{h}_i^\star$ and $\bar{\boldsymbol{h}}_i = \boldsymbol{h}_i^* \vee \bar{\boldsymbol{h}}_i^\star$, with $\text{supp}\{\boldsymbol{w}_i^* \boldsymbol{h}_i^{*\mathsf{T}}\} \nsubseteq \text{supp}\{\boldsymbol{X}^{(\setminus i)}\}$ and $\left\{\text{supp}\{\boldsymbol{w}_i^* \boldsymbol{h}_i^{\star\mathsf{T}}\}, \text{supp}\{\boldsymbol{w}_i^\star \boldsymbol{h}_i^{*\mathsf{T}}\}, \text{supp}\{\boldsymbol{w}_i^\star \boldsymbol{h}_i^{\star\mathsf{T}}\}, \text{supp}\{\boldsymbol{w}_i^* \bar{\boldsymbol{h}}_i^{\star\mathsf{T}}\}, \text{supp}\{\bar{\boldsymbol{w}}_i^\star \boldsymbol{h}_i^{*\mathsf{T}}\}, \text{supp}\{\bar{\boldsymbol{w}}_i^\star \bar{\boldsymbol{h}}_i^{\star\mathsf{T}}\}\right\} \subseteq \text{supp}\{\boldsymbol{X}^{(\setminus i)}\}$. As $\bar{\boldsymbol{w}}_i \neq \boldsymbol{w}_i$ or $\bar{\boldsymbol{h}}_i \neq \boldsymbol{h}_i$, it means that at least one of the supports of $\boldsymbol{w}_i^\star, \boldsymbol{h}_i^\star, \bar{\boldsymbol{w}}_i^\star, \bar{\boldsymbol{h}}_i^\star$ is non-empty. Suppose, without loss of generality, that $\text{supp}\{\boldsymbol{w}_i^\star\} \neq \emptyset$. As $\text{supp}\{\boldsymbol{w}_i^\star\} \subseteq \text{supp}\{\boldsymbol{w}_i\}$ and $\text{supp}\{\boldsymbol{w}_i^\star \boldsymbol{h}_i^\mathsf{T}\} \subseteq \text{supp}\{\boldsymbol{X}^{(\setminus i)}\}$ it means that $\text{supp}\{\boldsymbol{w}_i^\star\} \subseteq \bigcap_{k \in \Omega_{\boldsymbol{h}_i}} \text{supp}\{\boldsymbol{w}_k\} \neq \emptyset$, which contradicts the assumption and ends the proof.

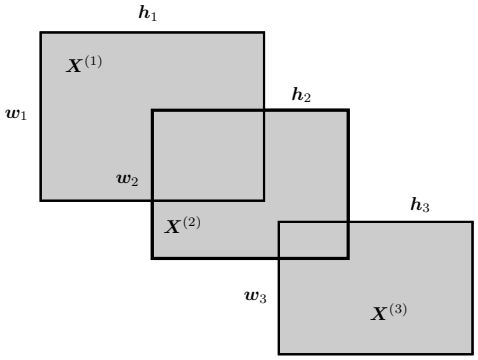
□

Figure 2 illustrates the partial uniqueness conditions in the case of a rank-3 decomposition; the grey rectangles represent the supports of the three sources. In Figure 2 (a) it can be seen that the partial uniqueness condition with respect to $\boldsymbol{X}^{(2)}$ is not satisfied. Indeed, $\text{supp}\{\boldsymbol{h}_2\} \subseteq (\text{supp}\{\boldsymbol{h}_1\} \cup \text{supp}\{\boldsymbol{h}_3\})$ and $\text{supp}\{\boldsymbol{w}_1\} \cap \text{supp}\{\boldsymbol{w}_3\} \neq \emptyset$. Thus, by subtracting, for example, the

hatched part from the support of $\boldsymbol{X}^{(2)}$, another admissible rank-1 term $\bar{\boldsymbol{X}}^{(2)} \neq \boldsymbol{X}^{(2)}$ can be obtained. On Figure 2 (b), $\text{supp}\{\boldsymbol{w}_1\} \cap \text{supp}\{\boldsymbol{w}_3\} = \emptyset$, and therefore, given $\boldsymbol{X}^{(1)}$ and $\boldsymbol{X}^{(3)}$, the only possible rank-1 term satisfying the decomposition is $\boldsymbol{X}^{(2)}$.



(a) *Partial non-uniqueness* with respect to $\boldsymbol{X}^{(2)}$: by subtracting the hatched part from the support of $\boldsymbol{X}^{(2)}$ we obtain another rank-1 term $\bar{\boldsymbol{X}}^{(2)}$, that satisfies $\boldsymbol{X} = \boldsymbol{X}^{(1)} \vee \bar{\boldsymbol{X}}^{(2)} \vee \boldsymbol{X}^{(3)}$.



(b) *Partial uniqueness* with respect to $\boldsymbol{X}^{(2)}$: for fixed $\boldsymbol{X}^{(1)}$ and $\boldsymbol{X}^{(3)}$, $\boldsymbol{X}^{(2)}$ is the only rank-1 binary matrix satisfying $\boldsymbol{X} = \boldsymbol{X}^{(1)} \vee \boldsymbol{X}^{(2)} \vee \boldsymbol{X}^{(3)}$.

Figure 2: An illustration of partial uniqueness condition of Theorem 3.2, for a rank-3 decomposition. The grey rectangles represent the supports of the three rank-1 terms.

Theorem 3.3 (Full uniqueness : necessary and sufficient condition)**.** *Let $\boldsymbol{X}$ be a binary matrix of* Boolean *rank K, following the* Boolean *decomposition given by* (6)*. The decomposition is* fully unique *(as defined in Definition 3.1) iff the partial uniqueness with respect to $\boldsymbol{X}^{(i)}$ is satisfied for all $i = 1, \ldots, K$.*

Proof. We only give hereafter the proof for the sufficient condition; the proof of the necessary condition is obvious. Let us suppose that the decomposition is not fully unique, meaning that there exists another rank-K decomposition $\boldsymbol{X} = \bar{\boldsymbol{X}}^{(1)} \vee \cdots \vee \bar{\boldsymbol{X}}^{(K)}$, such that at least one rank-1 term, say $\bar{\boldsymbol{X}}^{(i)}$, is such that $\boldsymbol{X}^{(i)} \neq \bar{\boldsymbol{X}}^{(i)}$. Let us re-write $\boldsymbol{X}^{(i)}$ as $\boldsymbol{X}^{(i)} = \boldsymbol{w}_i \boldsymbol{h}_i^\mathsf{T} = (\boldsymbol{w}_i^* \vee \boldsymbol{w}_i^\star)(\boldsymbol{h}_i^* \vee \boldsymbol{h}_i^\star)^\mathsf{T} = \boldsymbol{w}_i^* \boldsymbol{h}_i^{*\mathsf{T}} \vee \boldsymbol{w}_i^* \boldsymbol{h}_i^{\star\mathsf{T}} \vee \boldsymbol{w}_i^\star \boldsymbol{h}_i^{*\mathsf{T}} \vee \boldsymbol{w}_i^\star \boldsymbol{h}_i^{\star\mathsf{T}}$, with $\text{supp}\{\boldsymbol{w}_i^*\} \cap \text{supp}\{\boldsymbol{w}_i^\star\} = \emptyset$ and $\text{supp}\{\boldsymbol{h}_i^*\} \cap \text{supp}\{\boldsymbol{h}_i^\star\} = \emptyset$. Suppose that this decomposition is such that $\boldsymbol{w}_i^* \boldsymbol{h}_i^{*\mathsf{T}}$ is the rank-1 term with the smallest support such that: $\text{supp}\{\boldsymbol{w}_i^* \boldsymbol{h}_i^{*\mathsf{T}}\} \nsubseteq \bigcup_{k \neq i} \text{supp}\{\boldsymbol{X}^{(k)}\}$ and $\left\{\text{supp}\{\boldsymbol{w}_i^* \boldsymbol{h}_i^{\star\mathsf{T}}\}, \text{supp}\{\boldsymbol{w}_i^\star \boldsymbol{h}_i^{*\mathsf{T}}\}, \text{supp}\{\boldsymbol{w}_i^\star \boldsymbol{h}_i^{\star\mathsf{T}}\}\right\} \subseteq \bigcup_{k \neq i} \text{supp}\{\boldsymbol{X}^{(k)}\}$. This decomposition is always possible as one can always decompose a *Boolean* rank-1 term as a sum of four rank-1 term with disjoint supports. The support of $\boldsymbol{w}_i^* \boldsymbol{h}_i^{*\mathsf{T}}$ must be non-empty, because otherwise the *Boolean* rank of $\boldsymbol{X}$ would be less than K. For the same reason, any rank-K decomposition of $\boldsymbol{X}$ must have an ith term containing $\boldsymbol{w}_i^* \boldsymbol{h}_i^{*\mathsf{T}}$. Therefore, $\bar{\boldsymbol{X}}^{(i)}$ can be expressed as $\bar{\boldsymbol{X}}^{(i)} = \bar{\boldsymbol{w}}_i \bar{\boldsymbol{h}}_i^\mathsf{T} = (\boldsymbol{w}_i^* \vee \bar{\boldsymbol{w}}_i^\star)(\boldsymbol{h}_i^* \vee \bar{\boldsymbol{h}}_i^\star)^\mathsf{T} = \boldsymbol{w}_i^* \boldsymbol{h}_i^{*\mathsf{T}} \vee \boldsymbol{w}_i^* \bar{\boldsymbol{h}}_i^{\star\mathsf{T}} \vee \bar{\boldsymbol{w}}_i^\star \boldsymbol{h}_i^{*\mathsf{T}} \vee \bar{\boldsymbol{w}}_i^\star \bar{\boldsymbol{h}}_i^{\star\mathsf{T}}$, where the last three rank-1 terms are included in $\bigcup_{k \neq i} \text{supp}\{\bar{\boldsymbol{X}}^{(k)}\}$. Thus, we can write $\boldsymbol{X} = \bigvee_{k \neq i} \boldsymbol{X}^{(k)} \vee \boldsymbol{w}_i^* \boldsymbol{h}_i^{*\mathsf{T}} = \bigvee_{k \neq i} \bar{\boldsymbol{X}}^{(k)} \vee \boldsymbol{w}_i^* \boldsymbol{h}_i^{*\mathsf{T}}$. As both $\bigvee_{k \neq i} \boldsymbol{X}^{(k)}$ and $\bigvee_{k \neq i} \bar{\boldsymbol{X}}^{(k)}$ have a *Boolean* rank equal to $K - 1$ and the support of $\boldsymbol{w}_i^* \boldsymbol{h}_i^{*\mathsf{T}}$ is not included in the support of either of them, it means that $\bigvee_{k \neq i} \boldsymbol{X}^{(k)} = \bigvee_{k \neq i} \bar{\boldsymbol{X}}^{(k)}$. Using this result, we have: $\boldsymbol{X} = \bigvee_{k \neq i} \boldsymbol{X}^{(k)} \vee \boldsymbol{X}^{(i)} = \bigvee_{k \neq i} \boldsymbol{X}^{(k)} \vee \bar{\boldsymbol{X}}^{(i)}$,

meaning that the decomposition is not partially unique with respect to $\boldsymbol{X}^{(i)}$, which contradicts the hypothesis and ends the proof. □

A direct consequence of Theorem 3.2 and Theorem 3.3 is the following sufficient condition which is of practical interest.

Corollary 3.1 (Full uniqueness: a sufficient condition)**.** *The decomposition* (6) *is* fully unique *if its rank-1 terms* $\boldsymbol{X}^{(i)} = \boldsymbol{w}_i \boldsymbol{h}_i^{\mathsf{T}}$ *satisfy* $\text{supp}\{\boldsymbol{w}_i\} \nsubseteq \bigcup_{k \neq i} \text{supp}\{\boldsymbol{w}_k\}$ *and* $\text{supp}\{\boldsymbol{h}_i\} \nsubseteq \bigcup_{k \neq i} \text{supp}\{\boldsymbol{w}_k\}$, *for all* $i = 1, \ldots, K$.

Proof. The proof is directly obtained by applying the results of theorems 3.2 and 3.3. □

Roughly speaking, the uniqueness results presented in this section express the fact that a *Boolean* decomposition with rank-1 terms having overlapping supports (correlated sources) has less chances of being unique than a decomposition with non-overlapping supports (non-correlated sources), and the "probability of non-uniqueness" increases with the correlation degree of the sources. This will be illustrated on randomly generated binary matrices in section 5.1.

An interesting question is how are these uniqueness results related to the well-known *separability* and *sufficiently scattered* uniqueness conditions for the NMF model [44] ? The *Boolean* BMF model is in general non-linear in its parameters $\boldsymbol{W}$ and $\boldsymbol{H}$. Thus, the rationale that leads to NMF uniqueness conditions, which is based on linearity considerations, cannot even be applied to the *Boolean* BMF case. As shown in Section 2, the ranks of the two decompositions are generally not even the same, which makes impossible the comparison between the corresponding uniqueness results. However, the two models are equivalent is the case where both the columns of $\boldsymbol{W}$ and of $\boldsymbol{H}$ have disjoint supports, respectively. In this case, both models are identifiable, have the same rank, and the two decompositions yield exactly the same rank-1 terms $\boldsymbol{X}^{(k)}$. Nevertheless, even in this case, the columns of $\boldsymbol{W}$ and $\boldsymbol{H}$ for NMF, are subject to scaling/counter-scaling ambiguities, which is not the case for the *Boolean* BMF. In that respect, one could claim that the identifiability conditions for the *Boolean* BMF model are less restrictive those for NMF. Moreover, we have a necessary and sufficient condition for the identifiability of the *Boolean* BMF model, while such condition does not exist for NMF, in the general case.

4. A post-nonlinear mixture model approach to *Boolean* BMF

Consider the *Boolean* decomposition of a binary matrix given by (6). By arranging the factors $\mathbf{w}_k$ and $\mathbf{h}_k$, $k = 1, \ldots, K$ of the rank-1 terms on the columns of matrices $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_K]$ and $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_K]$, of respective sizes $N \times K$ and $M \times K$, the *Boolean* BMF can be re-written as:

$$\mathbf{X} = \mathbf{W} \diamond \mathbf{H}^T. \tag{8}$$

"$\diamond$" is called *Boolean matrix product* and is defined as $\mathbf{X}_{ij} = \bigvee_{k=1}^{K} (\mathbf{W}_{ik} \wedge \mathbf{H}_{jk})$, where "$\vee$" and "$\wedge$" are OR and AND logical operators, respectively. Until now we only considered the *exact* BMF, *i.e.,* $\boldsymbol{X}$ can be exactly recovered from the K rank-1 terms. In the sequel we suppose that the rank-K model can be corrupted by noise/errors. Depending on the set over which the decomposition of the binary matrix is performed, this noise term can take different forms. For the decomposition of $\boldsymbol{X}$ in binary terms, and for the *Boolean* BMF in particular, it is natural to assume that the entries of the "noise" matrix take values in the $\{0, 1\}$ set, and that we have

$$\boldsymbol{X} = \boldsymbol{W} \diamond \boldsymbol{H}^{\mathsf{T}} \oplus \boldsymbol{N}, \tag{9}$$

with "$\oplus$", the element-wise XOR logical operator. Intuitively, this means that the 1's of $\boldsymbol{N}$ change the corresponding entries of $\boldsymbol{W} \diamond \boldsymbol{H}^{\mathsf{T}}$ regardless of their respective values ("0" or "1").

Thus, the inverse problem to be solved in order to achieve the *Boolean* BMF is the following:

$$\{\boldsymbol{W}, \boldsymbol{H}\} = \arg \min_{\boldsymbol{W}, \boldsymbol{H} \in \{0,1\}} \left\| \boldsymbol{X} - \boldsymbol{W} \diamond \boldsymbol{H}^{\mathsf{T}} \right\|_{\mathsf{F}}^2 = \arg \min_{\boldsymbol{w}_k, \boldsymbol{h}_k \in \{0,1\}} \left\| \boldsymbol{X} - \bigvee_{k=1}^{K} \boldsymbol{w}_k \boldsymbol{h}_k^{\mathsf{T}} \right\|_{\mathsf{F}}^2. \tag{10}$$

It is worth noting that, for a binary-valued matrix, the *Frobenius* norm and the "entry-wise" norms ℓ_0 and ℓ_1 are equivalent.

As we mentioned earlier in this paper, the problem (10) is NP-complete [29], and two main strategies have been proposed to tackle this issue.

4.1. The relaxation of Boolean *BMF problem over the* nonnegative real *orthant.*

Several authors (*e.g.,* [32, 7, 33]) proposed to compute the BMF by relaxing it over the *nonnegative real* orthant. This comes down at replacing (10) by the inverse problem:

$$\{\boldsymbol{W}, \boldsymbol{H}\} = \arg \min_{\boldsymbol{W}, \boldsymbol{H} \in \{0,1\}} \left\| \boldsymbol{X} - \boldsymbol{W}\boldsymbol{H}^{\mathsf{T}} \right\|_{\mathsf{F}}^2 = \arg \min_{\boldsymbol{w}_k, \boldsymbol{h}_k \in \{0,1\}} \left\| \boldsymbol{X} - \sum_{k=1}^{K} \boldsymbol{w}_k \boldsymbol{h}_k^{\mathsf{T}} \right\|_{\mathsf{F}}^2, \tag{11}$$

where $\boldsymbol{W}\boldsymbol{H}^{\mathsf{T}}$ is the classical real-matrix product. By replacing the *Boolean* BMF problem (10) by its relaxation (11), the NP-hard problem is reduced to a classical real-valued optimization problem, easier to solve. However, one can easily see that (11) is equivalent to (10) only in the case where the rank-1 terms have disjoint supports. In fact, (11) solves the *binary* BMF problem instead of the *Boolean* BMF, as illustrated in Section 2. Therefore, all the algorithms based on (11) *assume implicitly that the sources are uncorrelated.* Nevertheless, the decomposition (11) may still yield good results if the source correlation is not "too strong", as illustrated by the following example.

Consider the *Boolean* BMF given by:

$$\underbrace{\begin{pmatrix}\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&\mathbf{1}\\0&0&\mathbf{1}&\mathbf{1}\end{pmatrix}}_{\boldsymbol{X}} = \underbrace{\begin{pmatrix}\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\0&0&0&0\end{pmatrix}}_{\boldsymbol{w}_1\boldsymbol{h}_1^\mathsf{T}} \vee \underbrace{\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&\mathbf{1}&\mathbf{1}\\0&0&\mathbf{1}&\mathbf{1}\end{pmatrix}}_{\boldsymbol{w}_2\boldsymbol{h}_2^\mathsf{T}}$$

This decomposition is unique, according to Corollary 3.1. Suppose now that we perform the rank-2 decomposition of $\boldsymbol{X}$ using (11) and obtain:

$$\underbrace{\begin{pmatrix}\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&\mathbf{1}\\0&0&\mathbf{1}&\mathbf{1}\end{pmatrix}}_{\boldsymbol{X}} \approx \underbrace{\begin{pmatrix}\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\0&0&0&0\end{pmatrix}}_{\boldsymbol{w}_1\boldsymbol{h}_1^\mathsf{T}} + \underbrace{\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&\mathbf{1}&\mathbf{1}\\0&0&\mathbf{1}&\mathbf{1}\end{pmatrix}}_{\boldsymbol{w}_2\boldsymbol{h}_2^\mathsf{T}} \approx \underbrace{\begin{pmatrix}\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\0&0&0&0\end{pmatrix}}_{\boldsymbol{w}_1\boldsymbol{h}_1^\mathsf{T}} + \underbrace{\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&\mathbf{1}\\0&0&0&\mathbf{1}\end{pmatrix}}_{\boldsymbol{w}_2\boldsymbol{h}_2^\mathsf{T}}.$$

It can be observed that in this case, the "true" decomposition produces the same minimum cost value $\|\boldsymbol{X}-\boldsymbol{W}\boldsymbol{H}^\mathsf{T}\|_\mathsf{F}^2$ =1 as the second binary decomposition. This means that a "good" algorithm based on model (11) may produce indistinctly the "true" solution or another equivalent one (with respect to the data fitting term). However, as the correlation of the two sources increases, the "true" solution of the *Boolean* decomposition of $\boldsymbol{X}$ corresponds no longer to a minimum of the cost function (11), as illustrated by the following example:

$$\underbrace{\begin{pmatrix}\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&\mathbf{1}\\\mathbf{1}&\mathbf{1}&\mathbf{1}&\mathbf{1}\\0&\mathbf{1}&\mathbf{1}&\mathbf{1}\end{pmatrix}}_{\boldsymbol{X}} = \underbrace{\begin{pmatrix}\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\0&0&0&0\end{pmatrix}}_{\boldsymbol{w}_1\boldsymbol{h}_1^\mathsf{T}} \vee \underbrace{\begin{pmatrix}0&0&0&0\\0&\mathbf{1}&\mathbf{1}&\mathbf{1}\\0&\mathbf{1}&\mathbf{1}&\mathbf{1}\\0&\mathbf{1}&\mathbf{1}&\mathbf{1}\end{pmatrix}}_{\boldsymbol{w}_2\boldsymbol{h}_2^\mathsf{T}}.$$

In this case, the "true" solution yields a cost $\|\boldsymbol{X}-\boldsymbol{W}\boldsymbol{H}^\mathsf{T}\|_\mathsf{F}^2 = 4$, while the decomposition:

$$\underbrace{\begin{pmatrix}\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&\mathbf{1}\\\mathbf{1}&\mathbf{1}&\mathbf{1}&\mathbf{1}\\0&\mathbf{1}&\mathbf{1}&\mathbf{1}\end{pmatrix}}_{\boldsymbol{X}} \approx \underbrace{\begin{pmatrix}\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\\mathbf{1}&\mathbf{1}&\mathbf{1}&0\\0&0&0&0\end{pmatrix}}_{\boldsymbol{w}_1\boldsymbol{h}_1^\mathsf{T}} + \underbrace{\begin{pmatrix}0&0&0&0\\0&0&0&\mathbf{1}\\0&0&0&\mathbf{1}\\0&0&0&\mathbf{1}\end{pmatrix}}_{\boldsymbol{w}_2\boldsymbol{h}_2^\mathsf{T}}$$

costs only $\|\boldsymbol{X}-\boldsymbol{W}\boldsymbol{H}^\mathsf{T}\|_\mathsf{F}^2 = 2$.

Two BMF algorithms based on the *real nonnegative* relaxation approach presented were proposed in [32], namely the Penalty Function algorithm (PF) and the THresholding algorithm (TH). The TH algorithm is based on a thresholding procedure of the entries of matrices $\boldsymbol{W}$ and $\boldsymbol{H}$, while the PF algorithm implements (11) by minimizing the cost function:

$$\sum_{i,j}\left(\boldsymbol{X}_{ij}-\left(\boldsymbol{W}\boldsymbol{H}^\mathsf{T}\right)_{ij}\right)^2 + \frac{1}{2}\lambda\sum_{i,k}\left(\boldsymbol{W}_{ik}^2-\boldsymbol{W}_{ik}\right)^2 + \frac{1}{2}\lambda\sum_{j,k}\left(\boldsymbol{H}_{jk}^2-\boldsymbol{H}_{jk}\right)^2 \tag{12}$$

The binary constraint on $\boldsymbol{H}$ and $\boldsymbol{W}$ is imposed by using the penalty terms $\boldsymbol{H}_{jk}^2-\boldsymbol{H}_{jk}$ and $\boldsymbol{W}_{ik}^2-\boldsymbol{W}_{ik}$. To minimize (12), a gradient descent algorithm with multiplicative update rule, similar to NMF [38, 39] is used. This algorithm was used as inspiration for the approach proposed in this paper.

4.2. Greedy *algorithms for the BMF problem*

The second main strategy used to tackle the NP-hardness of *Boolean* BMF is to iteratively estimate and subtract the rank-1 terms from $\boldsymbol{X}$, in a *greedy* manner. Several approaches have been proposed using different *greedy* strategies ([26, 28, 29, 30, 31]). For illustration and comparison purposes, we briefly present hereafter two such algorithms.

4.2.1. The ASSOciation rules algorithm (ASSO)

The ASSO algorithm for solving the *Discrete Basis Problem* was introduced in [29]. ASSO proceeds by deflation and seeks at each iteration the rank-1 term that minimizes the residual. The implementation of the algorithm exploits the correlation between the columns of $\boldsymbol{X}$ through the notion of *association* between the columns i and j, defined as $c(i \Rightarrow j, \boldsymbol{X}) = \frac{\boldsymbol{x}_i^\top \boldsymbol{x}_j}{\boldsymbol{x}_i^\top \boldsymbol{x}_i}$. These associations are used to form candidate basis vectors that will form the columns of $\boldsymbol{H}$, by a greedy selection.

4.2.2. The Formal Concept analysis algorithm (FC)

Another *greedy*-type BMF algorithm that will be used in this paper for comparison, was proposed in [30]. It is based on the *formal concept analysis* (*FCA*); a *formal concept* defines a binary relation between a set of objects and a set of attributes. Here, $\boldsymbol{X}$ is interpreted as an *object-attribute* matrix, $\boldsymbol{W}$ and $\boldsymbol{H}$ are interpreted as *object-factor* and *factor-attribute* matrices, respectively. The proposed algorithm is based on a theorem saying that the optimal decompositions of $\boldsymbol{X}$ (*i.e.,* those having a minimum rank K) are those where the factors are *formal concepts* in the sense of formal concept analysis. Roughly speaking, the approach consists in iteratively estimating and subtracting from $\boldsymbol{X}$, rank-1 terms $\boldsymbol{X}^{(k)}$ having maximum overlapping with $\boldsymbol{X}$. Two algorithms have been proposed in [30]. The first one starts by computing all the formal concepts and then, during each iteration, selects the one maximizing the intersection with $\boldsymbol{X}$. This algorithm produces good results but it is inefficient for large datasets. The second algorithm (that will be used for comparison in this paper), uses a more efficient strategy and constructs the factor concepts by adding sequentially "promising columns".

All *greedy* approaches mentioned in this section use deflation procedures to iteratively estimate the sources. This technique guarantees, similarly to the *real nonnegative* relaxation strategy, the exact recovery of the "true" sources only in the uncorrelated case. When the sources are correlated, their ability to recover the "true" sources will be highly dependent on their correlation level.

4.3. The proposed Post NonLinear Penalty Function algorithm (PNL-PF)

As explained in the previous subsection, most existing methods for the *Boolean* BMF yield good results only in the case of uncorrelated sources, because they are solving approximate, relaxed versions of the inverse problem (10). We propose in this section a BMF problem formulation based on a *post-nonlinear* mixture model which is equivalent to (10) when the matrices $\boldsymbol{W}$ and

$\boldsymbol{H}$ are exactly binary. An algorithm for estimating $\boldsymbol{W}$ and $\boldsymbol{H}$ based on this formulation is also introduced. Therefore, this new approach is expected to provide good results even in the case of highly correlated sources.

We preserve the idea of [32] to replace the *Boolean* matrix product by the real matrix product with binary constraints on $\boldsymbol{W}$ and $\boldsymbol{H}$, but we introduce a nonlinear function that guarantees the binarity of the reconstructed data. In theory, we desire a peace-wise linear function such that:

$$\Phi_{th}(x) = \begin{cases} 1 & \text{for } x > (1-\delta) \\ x & \text{for } \delta \leq \frac{x-\delta}{1-2\delta} \leq 1-\delta \\ 0 & \text{for } x < \delta \end{cases} \text{, with } 0 \leq \delta < 0.5. \tag{13}$$

Thus, the inverse problem we seek to solve, can be formulated as:

$$\{\boldsymbol{W}, \boldsymbol{H}\} = \arg \min_{\boldsymbol{W},\boldsymbol{H} \in \{0,1\}} \|\boldsymbol{X} - \Phi_{th}(\boldsymbol{W}\boldsymbol{H}^{\mathsf{T}})\|_{\mathsf{F}}^2. \tag{14}$$

However, for algorithmic convenience, in this paper we approximate $\Phi_{th}(x)$ by the sigmoid function $\Phi(x) = \frac{1}{1+e^{-\gamma(x-0.5)}}$ as it is continuously differentiable; the parameter γ allows to adjust the slope of Φ.

Figure 3 plots $\Phi_{th}(x)$ for $\delta = 0.1$ and $\Phi(x)$ for $\gamma = 1, 5, 10, 100$. One can see that, as the value of γ increases, $\Phi(x)$ fits better $\Phi_{th}(x)$. However, for algorithmic reasons, the value of γ must not be big in order to facilitate the parameters' variations during the iterations. In practice, a good choice for γ is a value between 1 and 10, depending on the application.

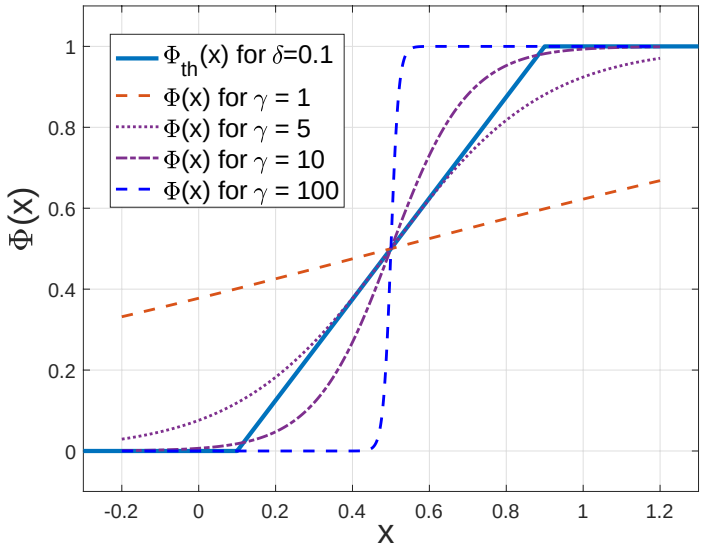


Figure 3: Function $\Phi_{th}(x)$ and its sigmoid approximations for various values of γ

To solve (14), we propose a gradient descent algorithm with a multiplicative update rule. The proposed algorithm minimizes the objective function (15); the fundamental difference compared

to (12) is the integration of the non-linear function Φ to guarantee $\Phi(\boldsymbol{W}\boldsymbol{H}^\mathsf{T}) \approx \boldsymbol{W} \diamond \boldsymbol{H}^\mathsf{T}$:

$$G(\boldsymbol{W},\boldsymbol{H}) = \frac{1}{2}\sum_{i,j}\left(\boldsymbol{X}_{ij} - \Phi(\boldsymbol{W}\boldsymbol{H}^\mathsf{T})_{ij}\right)^2 + \frac{1}{2}\lambda\sum_{i,k}\left(\boldsymbol{W}_{ik}^2 - \boldsymbol{W}_{ik}\right)^2 + \frac{1}{2}\lambda\sum_{j,k}\left(\boldsymbol{H}_{jk}^2 - \boldsymbol{H}_{jk}\right)^2. \quad (15)$$

Algorithm 4.1 : Post NonLinear Penalty Function algorithm (PNL-PF)

Input: $\boldsymbol{X}$, K, Nb_{iter}, λ, ϵ, γ

Output: $\boldsymbol{W}$, $\boldsymbol{H}$

STEP 1: Initializations: $\boldsymbol{W} \leftarrow rand(N,K)$, $\boldsymbol{H} \leftarrow rand(N,K)$, $iter = 0$

STEP 2: Normalization

$$\boldsymbol{W} \leftarrow \boldsymbol{W}\boldsymbol{D}_W^{-1/2}\boldsymbol{D}_H^{1/2}, \qquad \text{with } \boldsymbol{D}_W = diag\left(\max(\boldsymbol{w}_1), \max(\boldsymbol{w}_2), \cdots, \max(\boldsymbol{w}_K)\right)$$

$$\boldsymbol{H} \leftarrow \boldsymbol{H}\boldsymbol{D}_H^{-1/2}\boldsymbol{D}_W^{1/2}, \qquad \text{with } \boldsymbol{D}_H = diag\left(\max(\boldsymbol{h}_1), \max(\boldsymbol{h}_2), \cdots, \max(\boldsymbol{h}_K)\right)$$

STEP 3: Updates of $\boldsymbol{W}$, $\boldsymbol{H}$

$iter \leftarrow iter + 1$

$$\boldsymbol{H} \leftarrow \boldsymbol{H} * \frac{\gamma((\boldsymbol{X} * \Omega(\boldsymbol{W}\boldsymbol{H}^\mathsf{T}))^\mathsf{T} \cdot \boldsymbol{W}) + 3\lambda\boldsymbol{H}^2}{\gamma(\Psi(\boldsymbol{W}\boldsymbol{H}^\mathsf{T})^\mathsf{T} \cdot \boldsymbol{W}) + 2\lambda\boldsymbol{H}^3 + \lambda\boldsymbol{H}}$$

$$\boldsymbol{W} \leftarrow \boldsymbol{W} * \frac{\gamma((\boldsymbol{X} * \Omega(\boldsymbol{W}\boldsymbol{H}^\mathsf{T})) \cdot \boldsymbol{H}) + 3\lambda\boldsymbol{W}^2}{\gamma(\Psi(\boldsymbol{W}\boldsymbol{H}^\mathsf{T}) \cdot \boldsymbol{H}) + 2\lambda\boldsymbol{W}^3 + \lambda\boldsymbol{W}}$$

STEP 4: Stop criterion

if $iter \geq Nb_{iter}$ or $G(\boldsymbol{W},\boldsymbol{H}) < \epsilon$ **then**

  break

else

  return to *STEP 3*

end if

The steps of the proposed algorithm are presented in Algorithm 4.1. In practice, for a faster convergence, $\boldsymbol{W}$ and $\boldsymbol{H}$ are initialized with the result of the NMF algorithm [38]. In *STEP 3*, $\boldsymbol{W}$ and $\boldsymbol{H}$ are normalized in order to confine the values of $\boldsymbol{W}_{ij}$ and $\boldsymbol{H}_{ij}$ within the interval $[0,1]$. Ω and Ψ are two element-wise functions that associate to each entry $\boldsymbol{Z}_{ij}$ of a matrix $\boldsymbol{Z}$ (given in argument) the real values $\frac{e^{-\gamma\cdot(\boldsymbol{Z}_{ij}-0.5)}}{\left(1+e^{-\gamma\cdot(\boldsymbol{Z}_{ij}-0.5)}\right)^2}$ and $\frac{e^{-\gamma\cdot(\boldsymbol{Z}_{ij}-0.5)}}{\left(1+e^{-\gamma\cdot(\boldsymbol{Z}_{ij}-0.5)}\right)^3}$, respectively. In Algorithm 4.1, the power operation for a matrix $\boldsymbol{Z}$ is defined as $\boldsymbol{Z}^2 = \boldsymbol{Z} * \boldsymbol{Z}$, and the matrix division is performed element-wise. The derivation of the update rules for $\boldsymbol{W}$ and $\boldsymbol{H}$ are detailed in appendix Appendix A.

Convergence issues. The algorithm PNL-PF proposed in this section is based on multiplicative update rules, similar to the ones introduced in [38, 39] for NMF. It was proven in [39] that the NMF cost function is non-increasing after each update. However, several papers such as [45, 46] pointed out that such properties do not imply the convergence to a stationary point, and that

multiplicative NMF lacks sound optimization properties. While the cost function used in [38] is convex with respect to each of the two factor matrices, the PNL-PF optimization problem (15) is highly non-linear and non-convex because of the nonlinear function Φ and the binarity penalties. Therefore, providing sound theoretical convergence guarantees for the proposed algorithm is a highly difficult task and is an open problem in the current state of knowledge. Algorithms based on more advanced optimization procedures should be developed for convergence claims. However, this is beyond the scope of this paper and is part of ongoing work. Meanwhile, we study numerically in Section 5.1 the convergence of the proposed algorithm and show that it exhibits good convergence behavior in realistic scenarios.

5. Performance evaluation on synthetic and real data

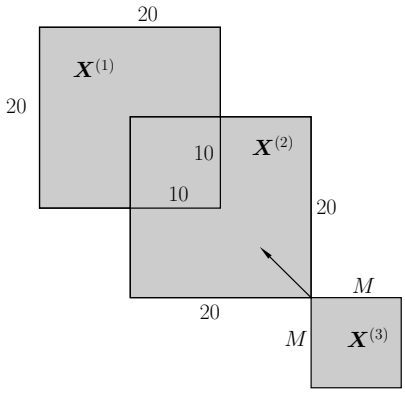
In this section we provide a numerical performance analysis of the proposed PNL-PF algorithm, compared to the state-of-the-art BMF approaches.

5.1. Evaluation on synthetic data

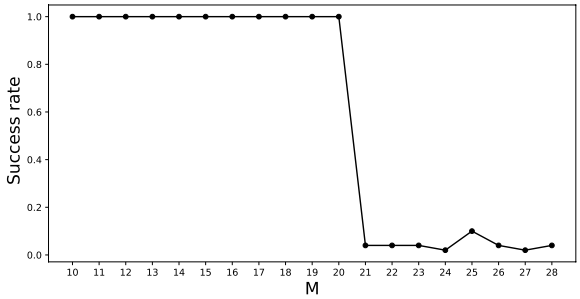
The first experiment aims at validating the identifiability results derived in Section 3. To this end, we designed a scenario with a *Boolean* mixture $\boldsymbol{X}$ (40×40) of three binary sources $\boldsymbol{X}^{(1)}$, $\boldsymbol{X}^{(2)}$ and $\boldsymbol{X}^{(3)}$. The supports of these sources are represented on Figure 4a: the first two sources have overlapping supports of size 20×20. The support of $\boldsymbol{X}^{(3)}$ is of size $M \times M$ and is touching the bottom right corner of $\boldsymbol{X}^{(3)}$, as illustrated. We have increased gradually the value of M by moving the top left corner of $\boldsymbol{X}^{(3)}$ towards $\boldsymbol{X}^{(1)}$, as indicated by the arrow, while fixing its bottom right corner. For each value of M, we performed the BMF using PNL-PF. On Figure 4b we plotted the rate of successful recovery of $\boldsymbol{X}^{(2)}$ from the mixture in 100 trials, for different values of M. It can be observed that we have a success rate of 100% for the values of $M \leq 20$, which corresponds to a range where the decomposition is unique, according to the results of Section 3. For $M > 20$, the support of $\boldsymbol{X}^{(3)}$ overlaps with the support of $\boldsymbol{X}^{(1)}$, and thus violates the uniqueness condition of Theorem 3.2. In this case, as expected, the success rate curve drops down to zero, which validates the uniqueness condition.

We study next the behavior of our approach in the presence of XOR binary noise, as defined in Section 4. We simulated the following model: $\boldsymbol{X} \oplus \boldsymbol{N}$, where the noise matrix $\boldsymbol{N}$ is a random binary matrix that follows a Bernouilli distribution of parameter b, and $\boldsymbol{X} = \boldsymbol{W} \diamond \boldsymbol{H}^{\mathsf{T}}$, is a rank-$K$ binary matrix. We computed the average reconstruction error for $\boldsymbol{X}$, $\mathrm{Err}_{\boldsymbol{X}} = \frac{\|\boldsymbol{X} - \hat{\boldsymbol{X}}\|_{\mathrm{F}}^2}{MN}$ (where $\hat{\boldsymbol{X}}$ is the reconstructed matrix) over 40 trials, as a function of the added noise rate b. $\boldsymbol{W}$ and $\boldsymbol{H}$ were generated according to a Bernoulli distribution of parameter p.

Thus, we designed a second experiment to compare the performance of the proposed PNL-PF algorithm with state-of-the-art methods, in the case of $K = 3$ "sparse" sources with $p = 0.25$ (which corresponds to a 1's to 0's ratio of ≈ 0.0625 for each rank-1 term). We plotted on Figure



(a) The source supports for the simulation set-up of the first experiment.



(b) The plot of success rate of the *Boolean* decomposition with respect to the values of M.

Figure 4: Numerical validation of the uniqueness condition derived in Section 3 using PNL-PF with $\gamma = 4$, $\lambda = 10$.

5 the reconstruction error $\text{Err}_{\boldsymbol{X}}$ with respect to the noise rate b, for PNL-PF, NMF [38], PF [32], FC [30], and ASSO [29]. One can see that, in the noiseless case ($b = 0$), all algorithms

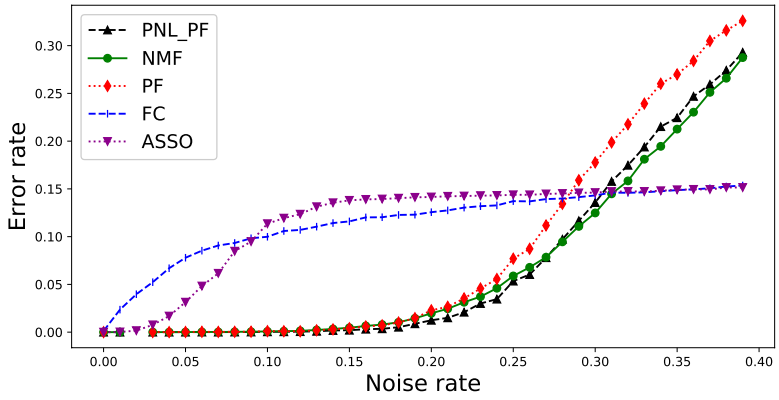


Figure 5: Comparison of the reconstruction error for PNL-PF with state-of -the-art algorithms, in the case of a rank-3 "sparse" binary matrix $\boldsymbol{X}$, for increasing additive XOR noise rates.

estimate $\boldsymbol{X}$ exactly. When the noise power increases, PNL-PF, PF and NMF still yield accurate estimations, while the two greedy algorithms (FC and ASSO) completely fail to reconstruct $\boldsymbol{X}$. This is an expected behavior as the greedy algorithms were designed for exact binary factorizations, and do not perform well for low-rank approximation problems. Moreover, one can observe that the proposed PNL-PF algorithm exhibits performances very similar to NMF; this is somewhat surprising, as NMF is not constraint to binary values, and therefore one may expect NMF to better fit the low-rank decomposition of $\boldsymbol{X}$. However, for high noise rates (> 0.3), all algorithms fail, as the low-rank structure of $\boldsymbol{X}$ is destroyed by the non-linear addition of the XOR noise.

The third experiment is very similar to the second one, with the difference that matrices $\boldsymbol{W}$ and $\boldsymbol{H}$ were generated according to a Bernoulli distribution of parameter $p = 0.6$, which implies much more "dense" rank-1 terms (with a 1's to 0's ratio of ≈ 0.36). It can be seen on Figure 6 that,

not only the performances of ASSO et FC degrade further, but also the gap between PNL-PF and PF increases. The reason for this is the strong overlapping of the rank-1 term supports (*i.e.,* the strong correlation of the sources). As explained in Section 4.1, in this case, the algorithms based on the classical matrix product of binary matrices, such as PF, no longer produce good results. Meanwhile, the proposed PNL-PF method still achieves good performances, close to NMF, thanks to its underlying post-nonlinear mixture model.

The results of these experiments also illustrates the fact that, for a binary-valued matrix, its binary decomposition (associated with the PF, FC and ASSO algorithms) requires a higher rank to achieve the same approximation error as compared to its *Boolean* decomposition (performed by PNL-PF).

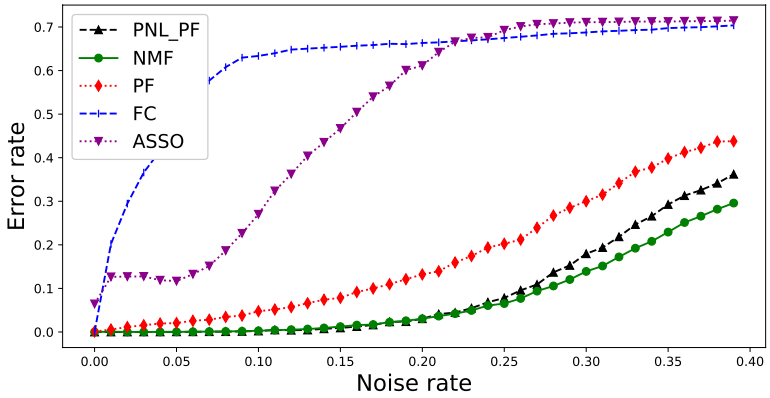


Figure 6: Comparison of the reconstruction error for PNL-PF with state-of -the-art algorithms, in the case a rank-3 "dense" binary matrix $\boldsymbol{X}$, for increasing additive XOR noise rates.

The fourth experiment on synthetic data (Figure 7) was designed to numerically study the convergence properties of the proposed algorithm PNL-PF. We plotted the cost functions for our algorithm in the case of 4 sources with $N = M = 200$ for in the noiseless case, and in the presence of 10% and 20% of XOR noise. The parameters used in the simulations are $\lambda = 10$ and $\gamma = 1$. The ratio "ones to zeros" for the simulated sources was set to 15% for figures A and B and to 85% for figures C and D. It can be seen from Figure 7 that our algorithm achieves fast convergence in all the considered scenarios. As expected, in the "non-correlated" case (figure A) the convergence is faster (after about 3 iterations) compared to the "correlated" case where the algorithm needs twice more iterations to converge. This numerical study shows (in the absence of an analytical convergence proof), that the proposed algorithm behaves well in most practical situations.

5.2. Application to real/ realistic data

We study in this section the compression capability of our method compared to the other state-of-the-art methods on real /realistic binary data extracted from the Causality Workbench database (`http://www.causality.inf.ethz.ch`).

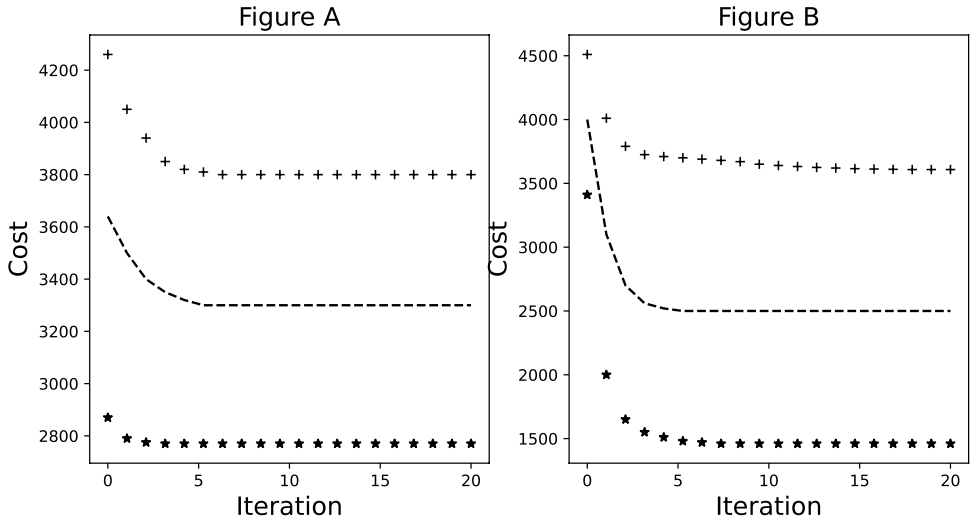


Figure 7: Illustration of the convergence speed of the proposed algorithms for three different levels of XOR noise ($K = 4$ sources, $N = M = 200$, $\lambda = 10$ and $\gamma = 1$). The ratio "ones to zeros" for the simulated sources is of 15% for figure A and 85% for figure B

.

The first experiment was performed on the `sido0_train` dataset. SIDO (SImple Drug Operation mechanisms) contains descriptors of molecules, which have been tested against the AIDS HIV virus. The binary values indicate the molecular activity: 1 - active, 0 - inactive (please refer to Causality Workbench website for further details on `sido0` dataset generation). The resulting binary matrix is of size 12678×4932 and could be qualified as "sparse", as only 9.8% of its entries are non-zero. We plotted in Figure 8 the reconstruction error for the algorithms mentioned in the previous section, for different ranks of the data matrix decomposition, going from 0 to 100. Once again, as expected, the "non-binary" NMF algorithm presents the best rank-compression performance. However, in terms of memory requirements, PNL-PF outperforms largely NMF. As one can see in Figure 8, for a rank 20, NMF gives the same reconstruction error as PNL-PF with a rank 25. In terms of storage memory, for the same compression error, NMF requires $(12678 + 4932) \times 20 \times 16 = 5635200$ bits (real-values are generally encoded on 2 bytes), while PNL-PF needs only $(12678 + 4932) \times 25 = 440250$ bites, that is 12 times less ! Moreover, our PNL-PF algorithm and the PF algorithm perform better than the greedy algorithms, especially for low ranks. This behavior can be explained by the fact that, for this "sparse" dataset, the correlation (overlapping) of the rank-1 terms is low, in which case the performances of PF and PNL-PF are similar (as explained in Section 4).

In the second experiment we used another dataset (`lucap0_test`) from the Causality Workbench database. LUCAP (LUng CAncer set with Probes) contains toy data generated artificially by causal Bayesian networks with binary variables and are modeling a medical application for the diagnosis, prevention, and cure of lung cancer (please refer to Causality Workbench website for

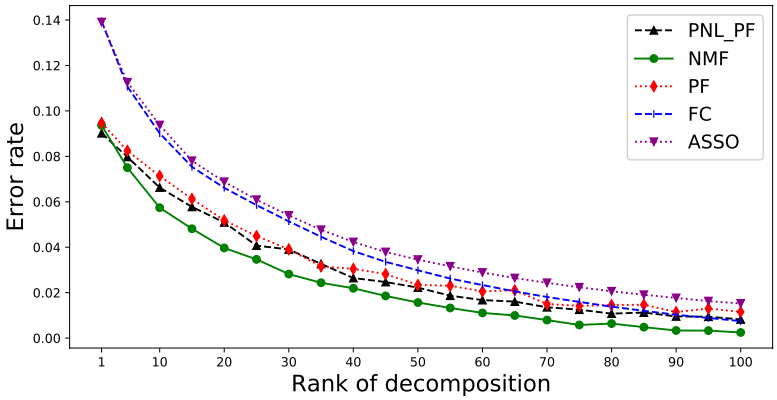


Figure 8: Comparison of the reconstruction error for PNL-PF with state-of-the-art algorithms, for the "sparse" `sido0_train` dataset, for different decomposition ranks.

further details). In this case the binary data matrix, of size 10000×143 is much more dense, with 51% of non-zero entries. For `lucap0_test` dataset, the overlapping of the rank-1 terms is much

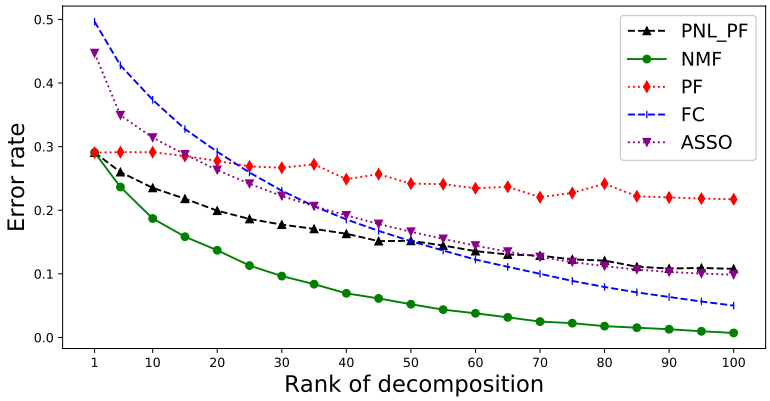


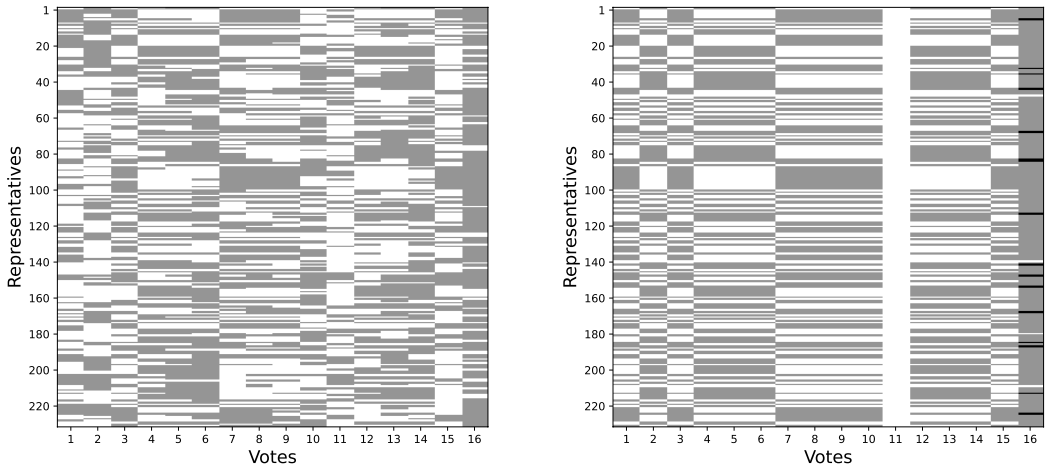
Figure 9: Comparison of the reconstruction error for PNL-PF with state-of-the-art algorithms, for the "dense" `lucap0_test` dataset, for different decomposition ranks.

more important and, consequently, the PF algorithm performs poorly as one can see on Figure 9. Meanwhile, the PNL-PF algorithm still gives accurate approximations for decomposition ranks between 1 and 50, and requires about 6 times less memory compared to NMF, which recommends it as a very interesting tool for low-rank approximation of binary-valued matrices.

The third experiment considers the `Congressional Voting Records Data Set` from the `UCI Machine Learning Repository.`[4] This dataset includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes identified by the CQA. After suppressing the lines containing missing values, we obtain a 235×16 binary-valued matrix ("1"= yes and "0"=no).

[4] `https://archive.ics.uci.edu/ml/datasets/Congressional+Voting+Records`

Each row corresponds to the votes of a representative (democrat/republican), and each column corresponds to a voted law, as one can see on Figure 10a. The ground truth (democrat/republican row labels) is available. For the images presented in this section, the white color encodes the "0" values, gray color the "1"s and black color is used to represent values equal to "2" resulting from the superposition of two rank-1 binary terms.



(a) The initial 235×16 dataset

(b) The arithmetic sum of the two rank-1 terms resulting from the decomposition of the dataset using PNL-PF. The black color indicates the overlapping values.

Figure 10: Congressional voting dataset

The goal of this experiment is to classify the representatives in one of the two classes: democrat or republican, based only on the way they voted. To this end, we applied the PNL-PF algorithm with a rank equal to 2; the result of this decomposition is presented on Figure 10b. Two remarks can be made:

- as expected, there is little overlapping between the supports of the two rank-1 terms, meaning that, globaly, the democrats and the republicans voted differently;
- the democrats and the republicans voted in a contradictory manner for law no.11 ("synfuels-corporation-cutback"), as the corresponding column is not discriminant for the classification.

Table 3 presents a comparison between the different binary decomposition algorithms in terms of classification performance. A representative is classified as democrat or as republican respectively, if its corresponding entry in the $\boldsymbol{w}_i$ vector of the rank-1 term corresponding to the democrat or republican camp respectively, is equal to "1". The column "Successful classification" corresponds to the representatives that were correctly classified as democrat or republican, "Ambiguous classification", to the ones that were labeled at the same time as democrat and republican, and the last column accounts for the non-classified representatives.

Algorithm	Successful classification	Ambiguous classification	Non-classified
PNL-PF	212	16	4
PF	212	16	4
FC	177	46	9
ASSO	131	97	4

Table 3: Performance comparison of representatives classification

As for this dataset the overlapping of the two rank-1 terms is very limited, the proposed PNL-PF algorithm gives the same decomposition as PF; however both surpass in classification performance the other two greedy methods, as it can be seen in Table 3. We do not have access to their identities but the ambiguously classified congressmen are 12 republicans (rows: 8, 35, 70, 85, 86, 115, 143, 155, 166, 186, 188, 225) and 4 democrats (rows: 38, 46, 149, 214).

In the fourth experiment we analyze the `UCI zoo dataset`[5] that was partially seen in Section 2. This dataset consists of 101 animal species from a zoo. There are 16 variables with various traits to describe the animals. We considered only the 15 *Boolean* valued variables: `airborne, feathers, breathes, backbone, tail, eggs, predator, aquatic, fins, toothed, milk, catsize, hair, venomous, domestic`, in this order. This results in a 101×15 binary-valued matrix that can be seen in Figure 11. We applied our PNL-PF algorithm to the data matrix from Figure 11

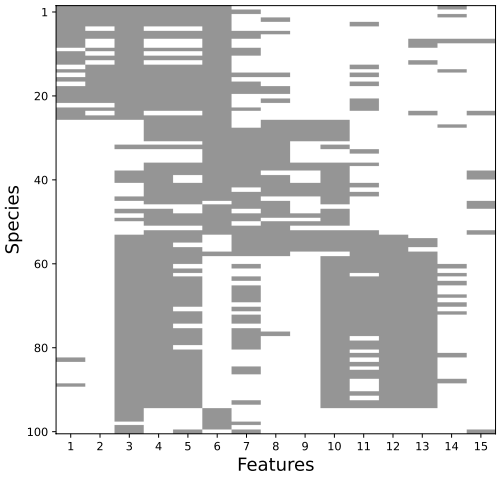


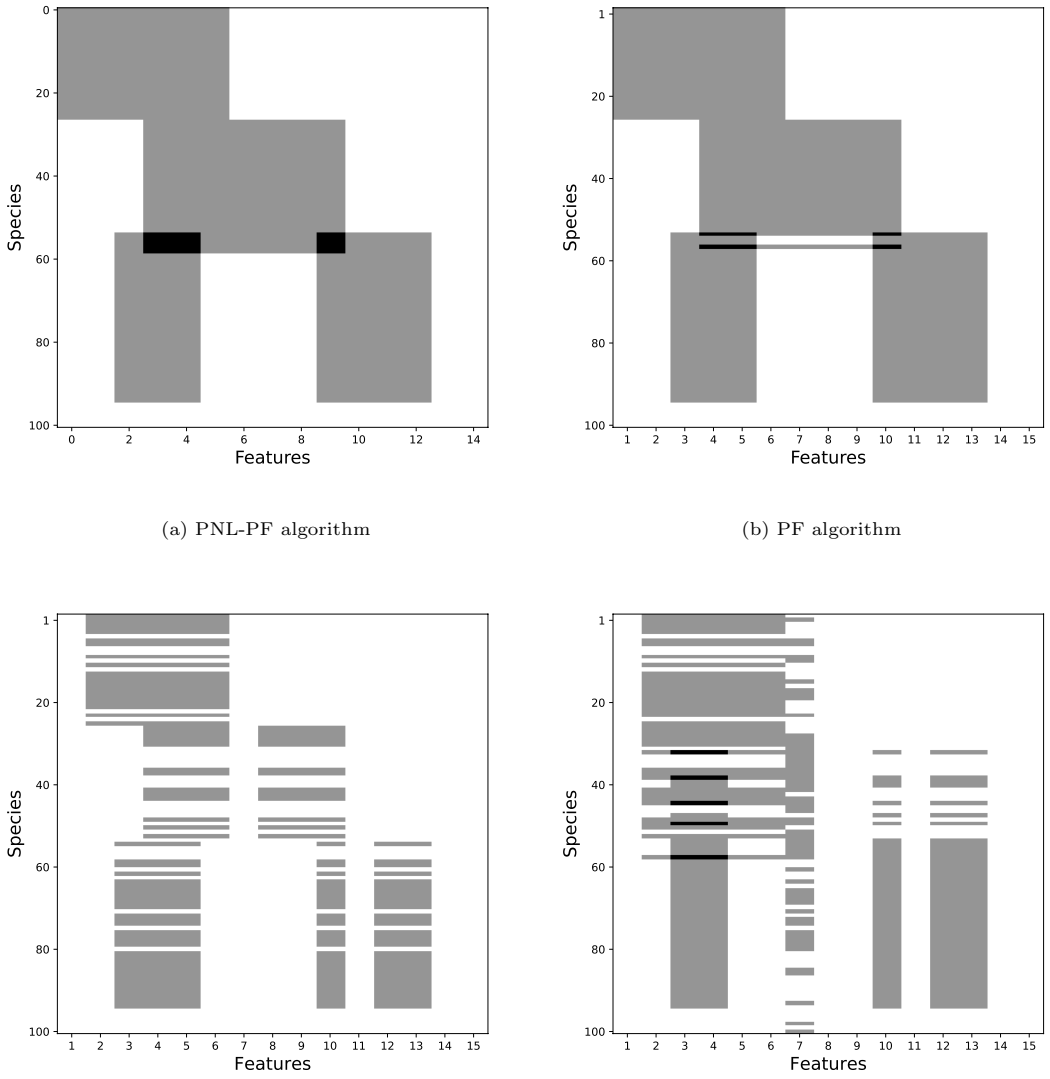
Figure 11: The initial zoo dataset

for different decomposition ranks. The highest rank for which the decomposition result was stable (the same result was obtained each time over 10 consecutive runs, with random initializations) is

[5] `https://archive.ics.uci.edu/ml/datasets/Zoo.`

rank 3. The result of the rank-3 *Boolean* decomposition by PNL-PF is plotted in Figure 12a. We obtain three animal classes sharing the following common features:

- features class #1: `airborne, feathers, breathes, backbone, tail, eggs`;
- features class #2: `backbone, tail, eggs, predator, aquatic, fins, toothed`;
- features class #3: `breathes, backbone, tail, toothed, milk, catsize, hair`.



(a) PNL-PF algorithm

(b) PF algorithm

(c) FC algorithm

(d) ASSO algorithm

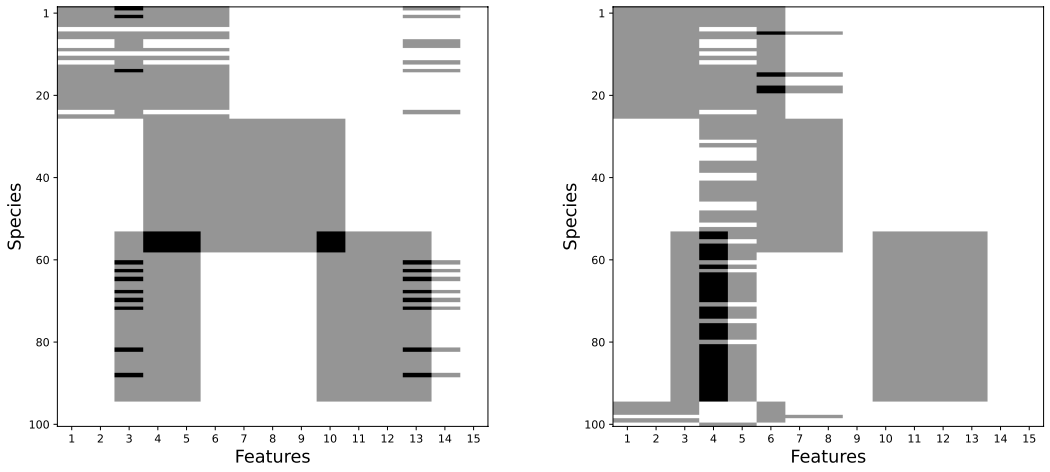
Figure 12: Results of the rank-3 decomposition of the zoo dataset from Figure 11

An animal from an identified class has most of the class features, but not necessarily all of them. Generally speaking, the first class represents the "birds", the second one, the "fishes" and the third one, the "mammals":

- class #1 ("birds"): `chicken, crow, dove, duck, flamingo, gnat, gull, hawk, honeybee, housefly, kiwi, ladybird, lark, moth, ostrich, parakeet, penguin, pheasant, rhea, skimmer, skua, sparrow, swan, tortoise, vulture, wasp, wren`;
- class #2 ("fishes"): `bass, carp, catfish, chub, crab, crayfish, dogfish,` **dolphin**, `frog, haddock, herring, lobster, newt, octopus, pike, piranha, pit viper,` **platypus**, **porpoise**, `seahorse,` **sea lion**, **seal**, `sea snake, sea wasp, slowworm, sole, starfish, stingray, toad, tuatara, tuna`;
- class #3 ("mammals"): `aardvark, antelope, bear, boar, buffalo, calf, cavy, cheetah, deer,` **dolphin**, `elephant, fruitbat, giraffe, girl, goat, gorilla, hamster, hare, leopard, lion, lynx, mink, mole, mongoose, opossum, oryx,` **platypus**, `polecat, pony,` **porpoise**, `puma, pussycat, raccoon, reindeer,` **seal**, **sea lion**, `squirrel, vampire, vole, wallaby, wolf.`

It can be seen that there is an overlapping between class #2 and class #3; they have the following animals in common: `dolphin, sea lion, seal, platypus, porpoise`. This is a reasonable result as these particular animals present feature belonging to both classes. Six other animals (`clam, flea, scorpion, slug, termite, worm`) were not included in any of these three classes because they do not present sufficient characteristics of any of them. Also, one can see that the algorithm did not take into account the last two attributes (`venomous, domestic`), as they were not discriminant for this rank-3 decomposition.The results for the three other binary algorithms are also given in Figure 12. As expected, PF yields a similar result, but with smaller overlapping between the last two terms. The results for FC and ASSO are more difficult to interpret; moreover, they exclude additional animals from the classification.

An important aspect in data analysis is the uniqueness of the decomposition, as it guarantees the unambiguous interpretation of the decomposition result. In the case of an *exact* rank-K *Boolean* decomposition of a binary matrix $\boldsymbol{X}$ $\left(\boldsymbol{X} = \bigvee_{k=1}^{K} \boldsymbol{w}_k \boldsymbol{h}_k^\mathsf{T}\right)$, the conditions proved in Section 3 can be directly used to asses uniqueness. However, in the case of a rank-K *approximation*, *i.e.,* $\boldsymbol{X} \approx \bigvee_{k=1}^{K} \boldsymbol{w}_k \boldsymbol{h}_k^\mathsf{T}$, one must first check that the approximation is stable, *i.e.*, the approximation error matrix does not change if we run the decomposition several times. If the decomposition is stable, the uniqueness conditions from Section 3 can then be applied to check the uniqueness of the rank-1 terms. This procedure is in no way different from the NMF case, when one needs to asses uniqueness of a non-negative low-rank approximation of a non-negative data matrix. In our case, the rank-3 PNL-PF decomposition seems stable over 10 runs. According to Corollary 3.1, the result in Figure 12a is unique, which guarantees an interpretable classification. The rank-4 decomposition, however, is not stable. For illustration, we plotted in Figure 13 two possible rank-4 approximation results of the zoo dataset. Two different classification (at least) can thus be obtained, each one with its own interpretation. Therefore, the uniqueness conditions derived



(a) (b)

Figure 13: Illustration of the non-uniqueness of the rank-4 decomposition of the dataset from Figure 11. Figures (a) and (b) represent two different rank-4 *Boolean* decompositions obtained by PNL-PF.

in Section 3 are highly useful in analyzing the stability and interpretability of the decomposition, and determining the adequate rank (number of classes).

6. Conclusions

In this paper we introduced a new approach to the *Boolean* factorization of binary-valued matrices, within the blind source separation framework. The proposed method is based on a post-nonlinear mixture model which is equivalent to the *Boolean* mixture model when the factors are exactly binary. This model explicitly accounts for the correlation of the mixed sources and therefore it yields more interpretable results in the case of "overlapping" factors, compared to the state-of-the-art algorithms. A simple, yet efficient algorithm to estimate the parameters of this new mixture model was proposed, based on NMF-like multiplicative update rules. In addition to this new approach, we provided, for the first time in the binary data literature, uniqueness conditions for the *Boolean* factorization of binary matrices. The proposed algorithms were compared to similar state-of-the-art methods on simulated and real data. We showed that, in the case of "overlapping" factors, these methods achieve accurate low-rank approximations of binary matrices, which is a highly desirable property in many signal processing and data compression applications. The approach introduced in this paper constitutes a promising tool for binary data/signal analysis especially in the case of highly correlated sources. Ongoing work aims at developing algorithms for the considered approach based on more advanced optimization procedures, allowing to provide theoretical convergence guarantees. A symmetric version of PNL-PF is also in development; it will allow to apply our method in network science to the analysis of no-oriented graphs.

Appendix A. Derivation of the update rules for the algorithm PNL-PF

Computation of the partial derivative of $G(\boldsymbol{W},\boldsymbol{H})$ with respect to $\boldsymbol{H}$:

$$\frac{\partial G(\boldsymbol{W},\boldsymbol{H})}{\partial \boldsymbol{H}_{pn}} = -\sum_i \left(\boldsymbol{X}_{ip} - \frac{1}{1+e^{-\gamma((\boldsymbol{W}\cdot\boldsymbol{H}^\mathsf{T})_{ip}-0.5)}}\right) \frac{\partial\left(\frac{1}{1+e^{-\gamma\cdot((\boldsymbol{W}\boldsymbol{H}^\mathsf{T})_{ip}-0.5)}}\right)}{\partial \boldsymbol{H}_{pn}} + \lambda(\boldsymbol{H}_{pn}^2 - \boldsymbol{H}_{pn})\cdot(2\boldsymbol{H}_{pn}-1).$$

Or $\frac{\partial\left(\frac{1}{1+e^{-\gamma\cdot((\boldsymbol{W}\boldsymbol{H}^\mathsf{T})_{ip}-0.5)}}\right)}{\partial \boldsymbol{H}_{pn}} = -\frac{\frac{\partial}{\partial \boldsymbol{H}_{pn}}\left(1+e^{-\gamma\cdot((\boldsymbol{W}\boldsymbol{H}^\mathsf{T})_{ip}-0.5)}\right)}{\left(1+e^{-\gamma\cdot((\boldsymbol{W}\boldsymbol{H}^\mathsf{T})_{ip}-0.5)}\right)^2}$ and $\frac{\partial}{\partial \boldsymbol{H}_{pn}}\left(1+e^{-\gamma\cdot((\boldsymbol{W}\boldsymbol{H}^\mathsf{T})_{ip}-0.5)}\right) = -\gamma \boldsymbol{W}_{in} e^{-\gamma\cdot\left((\boldsymbol{W}\boldsymbol{H}^\mathsf{T})_{ip}-0.5\right)}$ Then $\frac{\partial\left(\frac{1}{1+e^{-\gamma\cdot\left((\boldsymbol{W}\boldsymbol{H}^\mathsf{T})_{ip}-0.5\right)}}\right)}{\partial \boldsymbol{H}_{pn}} = \frac{\gamma \boldsymbol{W}_{in} e^{-\gamma\cdot\left((\boldsymbol{W}\boldsymbol{H}^\mathsf{T})_{ip}-0.5\right)}}{\left(1+e^{-\gamma\cdot((\boldsymbol{W}\boldsymbol{H}^\mathsf{T})_{ip}-0.5)}\right)^2}$, and

$$\frac{\partial G(\boldsymbol{W},\boldsymbol{H})}{\partial \boldsymbol{H}_{pn}} = -\sum_i \left(\boldsymbol{X}_{ip} - \frac{1}{1+e^{-\gamma((\boldsymbol{W}\cdot\boldsymbol{H}^\mathsf{T})_{ip}-0.5)}}\right)\cdot\frac{\gamma(\boldsymbol{W}^\mathsf{T})_{ni} e^{-\gamma\cdot\left((\boldsymbol{W}\boldsymbol{H}^\mathsf{T})_{ip}-0.5\right)}}{\left(1+e^{-\gamma\cdot((\boldsymbol{W}\boldsymbol{H}^\mathsf{T})_{ip}-0.5)}\right)^2} + 2\lambda {\boldsymbol{H}_{pn}}^3 - 3\lambda \boldsymbol{H}_{pn}^2 + \lambda \boldsymbol{H}_{pn}.$$

By introducing the Ω and Ψ functions defined in the main text, we obtain:

$$\frac{\partial G(\boldsymbol{W},\boldsymbol{H})}{\partial \boldsymbol{H}_{pn}} = -\gamma((\boldsymbol{X} * \Omega(\boldsymbol{W}\boldsymbol{H}^\mathsf{T}))^\mathsf{T}\cdot\boldsymbol{W})_{pn} + \gamma(\Psi(\boldsymbol{W}\boldsymbol{H}^\mathsf{T})^\mathsf{T}\cdot\boldsymbol{W})_{pn} + 2\lambda {\boldsymbol{H}_{pn}}^3 - 3\lambda \boldsymbol{H}_{pn}^2 + \lambda \boldsymbol{H}_{pn}.$$

The gradient descent for $\boldsymbol{H}$ can be expressed as:

$$\boldsymbol{H}_{pn} \leftarrow \boldsymbol{H}_{pn} - \alpha_{\boldsymbol{H}_{pn}} \frac{\partial}{\boldsymbol{H}_{pn}} G(\boldsymbol{W},\boldsymbol{H}).$$

By choosing $\alpha_{\boldsymbol{H}_{pn}}$ as:

$$\alpha_{\boldsymbol{H}_{pn}} = \frac{\boldsymbol{H}_{pn}}{\gamma(\Psi(\boldsymbol{W}\boldsymbol{H}^\mathsf{T})^\mathsf{T}\cdot\boldsymbol{W})_{pn} + 2\lambda {\boldsymbol{H}_{pn}}^3 + \lambda \boldsymbol{H}_{pn}},$$

the following update rule for $\boldsymbol{H}$ is obtained:

$$\boldsymbol{H}_{pn} \leftarrow \boldsymbol{H}_{pn} \frac{\gamma((\boldsymbol{X} * \Omega(\boldsymbol{W}\boldsymbol{H}^\mathsf{T}))^\mathsf{T}\cdot\boldsymbol{W})_{pn} + 3\lambda \boldsymbol{H}_{pn}^2}{\gamma(\Psi(\boldsymbol{W}\boldsymbol{H}^\mathsf{T})^\mathsf{T}\cdot\boldsymbol{W})_{pn} + 2\lambda {\boldsymbol{H}_{pn}}^3 + \lambda \boldsymbol{H}_{pn}}$$

$$\boldsymbol{H} \leftarrow \boldsymbol{H} * \frac{\gamma((\boldsymbol{X} * \Omega(\boldsymbol{W}\boldsymbol{H}^\mathsf{T}))^\mathsf{T}\cdot\boldsymbol{W}) + 3\lambda \boldsymbol{H}^2}{\gamma(\Psi(\boldsymbol{W}\boldsymbol{H}^\mathsf{T})^\mathsf{T}\cdot\boldsymbol{W}) + 2\lambda \boldsymbol{H}^3 + \lambda \boldsymbol{H}},$$

where the matrix division is taken element-wise and matrix power is defined as $\boldsymbol{Z}^2 = \boldsymbol{Z} * \boldsymbol{Z}$.

By switching the roles of $\boldsymbol{W}$ and $\boldsymbol{H}$, one can easily obtain the update rules for $\boldsymbol{W}$ as:

$$\boldsymbol{W}_{mn} \leftarrow \boldsymbol{W}_{mn} \frac{\gamma((\boldsymbol{X} * \Omega(\boldsymbol{W}\boldsymbol{H}^\mathsf{T}))\cdot\boldsymbol{H})_{mn} + 3\lambda \boldsymbol{W}_{mn}^2}{\gamma(\Psi(\boldsymbol{W}\boldsymbol{H}^\mathsf{T})\cdot\boldsymbol{W})_{mn} + 2\lambda {\boldsymbol{W}_{mn}}^3 + \lambda \boldsymbol{W}_{mn}}$$

$$\boldsymbol{W} \leftarrow \boldsymbol{W} * \frac{\gamma((\boldsymbol{X} * \Omega(\boldsymbol{W}\boldsymbol{H}^\mathsf{T}))\cdot\boldsymbol{H}) + 3\lambda \boldsymbol{W}^2}{\gamma(\Psi(\boldsymbol{W}\boldsymbol{H}^\mathsf{T})\cdot\boldsymbol{W}) + 2\lambda \boldsymbol{W}^3 + \lambda \boldsymbol{W}}.$$

References

[1] P. T. Boufounos, R. G. Baraniuk, 1-bit compressive sensing, in: 42nd Annual Conference on Information Sciences and Systems, (CISS 2008), IEEE, 2008, pp. 16–21.

[2] L. Jacques, J. N. Laska, P. T. Boufounos, R. G. Baraniuk, Robust 1-bit compressive sensing via binary stable embeddings of sparse vectors, IEEE Transactions on Information Theory 59 (4) (2013) 2082–2102.

[3] S. Talwar, M. Viberg, A. Paulraj, Blind separation of synchronous co-channel digital signals using an antenna array. Part I. Algorithms, IEEE Transactions on Signal Processing 44 (5) (1996) 1184–1197.

[4] A.-J. Van der Veen, Analytical method for blind binary signal separation, IEEE Transactions on Signal Processing 45 (4) (1997) 1078–1082.

[5] H. Lu, J. Vaidya, V. Atluri, Y. Hong, Constraint-aware role mining via extended boolean matrix decomposition, IEEE Transactions on Dependable and Secure Computing 9 (5) (2012) 655–669.

[6] E. Meeds, Z. Ghahramani, R. M. Neal, S. T. Roweis, Modeling dyadic data with binary latent factors, in: Advances in neural information processing systems, 2006, pp. 977–984.

[7] Z.-Y. Zhang, T. Li, C. Ding, X.-W. Ren, X.-S. Zhang, Binary matrix factorization for analyzing gene expression data, Data Mining and Knowledge Discovery 20 (1) (2010) 28–52.

[8] S. Tu, R. Chen, L. Xu, A binary matrix factorization algorithm for protein complex prediction, Proteome Science 9 (1) (2011) 1.

[9] T. Li, A general model for clustering binary data, in: The 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, ACM, 2005, pp. 188–197.

[10] B.-H. Shen, S. Ji, J. Ye, Mining discrete patterns via binary matrix factorization, in: The 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2009, pp. 757–766.

[11] E. Nenova, D. I. Ignatov, A. V. Konstantinov, An FCA-based boolean matrix factorisation for collaborative filtering, arXiv preprint arXiv:1310.4366.

[12] A. I. Schein, L. K. Saul, L. H. Ungar, A generalized linear model for principal component analysis of binary data., in: AISTATS, Vol. 3, 2003, p. 10.

[13] J. De Leeuw, Principal component analysis of binary data by iterated singular value decomposition, Computational statistics & data analysis 50 (1) (2006) 21–39.

[14] L. Kozma, A. Ilin, T. Raiko, Binary principal component analysis in the Netflix collaborative filtering task, in: IEEE International Workshop on Machine Learning for Signal Processing, (MLSP 2009), IEEE, 2009, pp. 1–6.

[15] S. Lee, J. Z. Huang, J. Hu, Sparse logistic principal components analysis for binary data, The annals of applied statistics 4 (3) (2010) 1579.

[16] Z. Kang, C. J. Spanos, Sequential logistic principal component analysis (SLPCA): Dimensional reduction in streaming multivariate binary-state system, in: 13th International Conference on Machine Learning and Applications, (ICMLA 2014), IEEE, 2014, pp. 171–177.

[17] Y. Shen, M. Mardani, G. B. Giannakis, Online categorical subspace learning for sketching big data with misses, IEEE Transactions on Signal Processing 65 (15) (2017) 4004–4018.

[18] T. Cai, W.-X. Zhou, A max-norm constrained minimization approach to 1-bit matrix completion, The Journal of Machine Learning Research 14 (1) (2013) 3619–3647.

[19] M. A. Davenport, Y. Plan, E. Van Den Berg, M. Wootters, 1-bit matrix completion, Information and Inference: A Journal of the IMA 3 (3) (2014) 189–223.

[20] S. A. Bhaskar, A. Javanmard, 1-bit matrix completion under exact low-rank constraint, in: 49th Annual Conference on Information Sciences and Systems, (CISS 2015), IEEE, 2015.

[21] V. Cottet, P. Alquier, 1-bit matrix completion: PAC-Bayesian analysis of a variational approximation, Machine Learning 107 (3) (2018) 579–603.

[22] P. Pajunen, Blind separation of binary sources with less sensors than sources, in: IEEE International Conference on Neural Networks, (ICNN1997), Vol. 3, IEEE, 1997, pp. 1994–1997.

[23] A. A. Frolov, A. Abraham, P. Y. Polyakov, D. Húsek, H. Řezanková, BFA and BMF: What is the difference, in: 12th International Conference on Intelligent Systems Design and Applications (ISDA 2012), IEEE, 2012, pp. 890–896.

[24] R. Khanna, L. Zhang, D. Agarwal, B.-C. Chen, Parallel matrix factorization for binary response, in: IEEE International Conference on Big Data (ICBD 2013), IEEE, 2013, pp. 430–438.

[25] S. Ravanbakhsh, B. Póczos, R. Greiner, Boolean matrix factorization and noisy completion via message passing., in: ICML, 2016, pp. 945–954.

[26] M. Koyutürk, A. Grama, N. Ramakrishnan, Algebraic techniques for analysis of large discrete-valued datasets, in: European Conference on Principles of Data Mining and Knowledge Discovery, Springer, 2002, pp. 311–324.

[27] M. Koyutürk, A. Grama, PROXIMUS: a framework for analyzing very high dimensional discrete-attributed datasets, in: The ninth ACM SIGKD International Conference on Knowledge Discovery and Data Mining, ACM, 2003, pp. 147–156.

[28] H. Lu, J. Vaidya, V. Atluri, Optimal boolean matrix decomposition: Application to role engineering, in: 24th IEEE International Conference on Data Engineering (ICDE 2008), IEEE, 2008, pp. 297–306.

[29] P. Miettinen, T. Mielikainen, A. Gionis, G. Das, H. Mannila, The discrete basis problem, IEEE Transactions on Knowledge and Data Engineering 20 (10) (2008) 1348–1362.

[30] R. Belohlavek, V. Vychodil, Discovery of optimal factors in binary data via a novel method of matrix decomposition, Journal of Computer and System Sciences 76 (1) (2010) 3–20.

[31] H. Nguyen, R. Zheng, Binary independent component analysis with OR mixtures, IEEE Transactions on Signal Processing 59 (7) (2011) 3168–3181.

[32] Z. Zhang, C. Ding, T. Li, X. Zhang, Binary matrix factorization with applications, in: IEEE Seventh International Conference on Data Mining (ICDM 2007), IEEE, 2007, pp. 391–400.

[33] L. Mukherjee, S. N. Ravi, V. K. Ithapu, T. Holmes, V. Singh, An NMF perspective on binary hashing, in: IEEE International Conference on Computer Vision, (ICCV 2015), 2015, pp. 4184–4192.

[34] G. Govaert, M. Nadif, Block clustering with Bernoulli mixture models: Comparison of different approaches, Computational Statistics & Data Analysis 52 (6) (2008) 3233–3245.

[35] A. P. Streich, M. Frank, D. Basin, J. M. Buhmann, Multi-assignment clustering for Boolean data, in: 26th Annual International Conference on Machine Learning, ACM, 2009, pp. 969–976.

[36] L. Labiod, M. Nadif, Co-clustering for binary and categorical data with maximum modularity, in: 2011 IEEE 11th International Conference on Data Mining (ICDM 2011), IEEE, 2011, pp. 1140–1145.

[37] S. Sukhanov, C. Debes, A. M. Zoubir, Interpretable clustering ensembles using binary matrix factorization, in: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2018), IEEE, 2018.

[38] D. D. Lee, H. S. Seung, Learning the parts of objects by non-negative matrix factorization, Nature 401 (6755) (1999) 788–791.

[39] D. D. Lee, H. S. Seung, Algorithms for non-negative matrix factorization, in: Advances in neural information processing systems, 2001, pp. 556–562.

[40] T. Watson, Nonnegative rank vs. binary rank, Chicago Journal of Theoretical Computer Science 2 (2016) 1–13.

[41] V. L. Watts, Boolean rank of Kronecker products, Linear Algebra and its Applications 336 (1-3) (2001) 261–264.

[42] K. H. Kim, Boolean matrix theory and applications, Vol. 70, Dekker, 1982.

[43] J. Orlin, Contentment in graph theory: covering graphs with cliques, in: Indagationes Mathematicae (Proceedings), Vol. 80, Elsevier, 1977, pp. 406–424.

[44] X. Fu, K. Huang, N. D. Sidiropoulos, W.-K. Ma, Nonnegative matrix factorization for signal and data analytics: Identifiability, algorithms, and applications., IEEE Signal Processing Magazine 36 (2) (2019) 59–80.

[45] E. F. Gonzalez, Y. Zhang, Accelerating the Lee-Seung algorithm for nonnegative matrix factorization, Tech. rep., Department of Computational and Applied Mathematics, Rice University (2005).

[46] C.-J. Lin, Projected gradient methods for nonnegative matrix factorization, Neural computation 19 (10) (2007) 2756–2779.