ORTHOGONAL SYMMETRIC NONNEGATIVE MATRIX TRI-FACTORIZATION

Alexandra Dache *Arnaud Vandaele* *Nicolas Gillis*

University of Mons

ABSTRACT

Symmetric nonnegative matrix tri-factorization (trisymNMF) factorizes a symmetric input n-by-n matrix, X, using two matrices, a nonnegative n-by-r matrix W and a nonnegative symmetric r-by-r matrix S, such that $X \approx WSW^\top$. trisymNMF has been used in many applications, including topic modeling and community detection. In the latter application, each column of W corresponds to a community, and the entries of S indicate the interactions between the communities. In this paper, we focus on the particular case of an orthogonal matrix W, corresponding to the case of disjoint communities. We first analyse this problem, and in particular provide an identifiability result. Then we propose an efficient coordinate descent method that exploits the properties of the problem to solve it efficiently. We also propose a new initialization strategy, provably correct in the noiseless case and robust to noise, relying on smooth separable NMF. Empirically, we show that our method outperforms the state of the art on synthetic and real data.

***Index Terms*—** nonnegative matrix factorization (NMF), symmetry, orthogonality, tri-factorization, clustering

1. INTRODUCTION

Given a nonnegative symmetric matrix $X \in \mathbb{R}^{n\times n}_+$ and a factorization rank $r \in \mathbb{N}$, symmetric non-negative matrix tri-factorization (trisymNMF) looks for a nonnegative matrix $W \in \mathbb{R}^{n\times r}$ and a nonnegative symmetric matrix $S \in \mathbb{R}^{r\times r}$ such that $X \approx WSW^\top$. trisymNMF is a generalization of symmetric nonnegative matrix factorization (symNMF) where S is the identity matrix, which has been used successfully in various applications; see [1] for a recent survey. However, symNMF has several drawbacks; for example it can only factorize positive semidefinite matrices (since $WW^\top$ is positive semidefinite), e.g., the matrix

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{1}$$

cannot be factorized with symNMF for any r because it has a negative eigenvalue; this is related to the notion of the cp rank of symmetric nonnegative matrices [2]. Hence, trisymNMF has the advantage to be more expressive, and be able to factorize any matrix, in the worst case with the trivial decomposition $X = IXI$ of rank n. Since the input matrix X is symmetric, it can be interpreted as the adjacency matrix of a graph, where the entry $X(i,j) = X(j,i)$ indicates the weight of the connection between nodes i and j. Consequently, trisymNMF can be interpreted as a community detection model. For each element of the matrix X, we have:

$$X(i,j) \approx W(i,:)SW(:,j)^\top = \sum_{k=1}^{r}\sum_{l=1}^{r} W(i,k)S(k,l)W(j,l).$$

The columns of W identify the communities, i.e., strongly correlated elements in the dataset. If $W(j,k)$ is non-zero, element j belongs to community k. The matrix S enables interactions between communities, where the entry $S(k,l)$ represents the strength of the connection between communities k and l. trisymNMF is closely related to the stochastic block model; see, e.g., the discussion in [3].

In this work, we focus on finding disjoint communities, meaning that each node belongs to only one community, while communities can interact together. Mathematically, this constraint translates to ensuring that there is at most one non-zero element per row of W. To satisfy this constraint, W can be imposed to be column-wise orthogonal: together with nonnegativity, orthogonality implies that each row of W has at most a single positive entry. We measure the error using the Frobenius norm, and consider the following problem

$$\min_{W\geq 0, S\geq 0} \|X - WSW^\top\|_F^2 \quad \text{such that} \quad W^\top W = I, \tag{2}$$

referred to as orthogonal trisymNMF (OtrisymNMF). This model was introduced in [4] and used for clustering.

Outline and contribution of the paper. In this paper, we revisit OtrisymNMF. In Section 2, we discuss the uniqueness/identifiability of OtrisymNMF. In Section 3, we propose a new algorithm based on coordinate descent. In Section 4, we provide experiments on synthetic data sets to show the superiority of the new proposed algorithm compared to the state of the art, and experiments on real data sets to show the effectiveness of OtrisymNMF for clustering, and also show it superiority over orthogonal NMF, symNMF and trisymNMF in terms of expressiveness, that is, OtrisymNMF can obtain smaller errors for the same number of parameters.

Authors acknowledge the support by the European Union (ERC consolidator, eLinoR, no 101085607).

2. INTERPRETATION AND UNIQUENESS

Let us first interpret OtrisymNMF (OtrisymNMF), $X = WSW^\top$. For W nonnegative ($W \geq 0$) and W orthogonal ($W^\top W = I$), each row of W has at most a single positive entry, because nonnegative orthogonal vector (here, the columns of W) must have disjoint supports. Let us denote $k_i \in \{1, 2, 3, \dots, r\}$ the index of the non-zero element in the ith row of W. In other words, the kth columns of W corresponds to the community $\mathcal{C}_k = \{i \mid k_i = k\}$. For all i, j,

$$X(i,j) = W(i,:)\, S\, W(j,:)^\top = W(i,k_i)\, S(k_i,k_j)\, W(j,k_j).$$

This means that the entry $X(i,j)$ is positive if and only if the ith and jth nodes belong to two communities that interact. Note that nothing prevents the model from having $S(k,k) = 0$ which would mean that nodes in the community k do not interact, although they behave similarly. For example, this could represent the dating activities of heterosexual males and females, or the number of fights between people from different karate clubs (assuming people within the same club cannot fight each other). The simplest example where this happens is for the matrix from (1) with $X = IXI$.

Let us now prove the uniqueness of factors in OtrisymNMF. Although this result is relatively straightforward, we did not find it in the literature. The proof is closely related to that of orthogonal NMF [5, Theorem 4.40, p. 136].

Theorem 1. *Let $X = WSW^\top$ where $W \in \mathbb{R}^{n \times r}_+$ satisfies $W^\top W = I$, and $S \in \mathbb{R}^{r \times r}_+$ satisfies $\operatorname{rank}(S) = r$. Then the exact OtrisymNMF X of size r is unique up to permutation, that is, any other OtrisymNMF of X must be obtained by permutations of the columns of W and the corresponding permutations of rows and columns of S, which is nothing more than a change in cluster indexing.*

Proof. Since $X = (WS)W^\top$ and $W^\top$ has a single non-zero entry in each column, each column of X is a multiple of one of the r columns of WS. Moreover, the rank of WS and of X is equal to r, since $W^\top WS = S$ and $W^\top XW = S$ while $\operatorname{rank}(S) = r$, because $\operatorname{rank}(AB) \leq \min(\operatorname{rank}(A), \operatorname{rank}(B))$ for any matrices A and B. This implies that X has, up to scaling, r distinct columns. Let us now consider another orthogonal try-symNMF of $X = \hat{W}\hat{S}\hat{W}^\top$. Since $\operatorname{rank}(X) = r$, we must have $\operatorname{rank}(\hat{S}) = r$. For the same reasons as above, the columns of $\hat{W}\hat{S}$ are a multiple of the columns of X. Since the rank of $\hat{W}\hat{S}$ is r, these r columns must be a multiple of the distinct columns of X, that is, of the r columns of WS. This implies $\hat{W}\hat{S} = WSD\Pi$ where Π is a permutation matrix and D is a diagonal matrix. Now, we have

$$X = WSW^\top = \hat{W}\hat{S}\hat{W}^\top = WSD\Pi\hat{W}^\top.$$

Multiplying on the left by $S^{-1}W^\top$, we obtain $W^\top = D\Pi\hat{W}^\top$ while by orthogonality of W and $\hat{W}$, D is the identity matrix (since $W^\top W = DD = I$), leading to $\hat{S} = \Pi^\top S \Pi$, which concludes the proof. $\square$

3. BLOCK COORDINATE DESCENT FOR OTRISYMNMF

In most factorization algorithms, it is common to optimize over the different factors. We adopt this framework for our problem by updating alternately W and S while the other factor is fixed. The article [4] proposes an iterative algorithm by updating W and then S. The formulas do not exploit the specific structure of W. Another article [6] proposes a method that adds two penalty terms based on α-divergence to incorporate the orthogonality of W.

In this paper, we propose an iterative method leveraging the disjoint support of the rows of W by updating W row-wise. The calculations are simplified by working only with the non-zero elements of W. To do so, we resort to a block coordinate descent method. We update alternatively the rows of W, normalize W, and then update S.

Update of W. First note that there is a scaling degree of freedom in OtrisymNMF: if we have a factorization $WSW^\top$ where $WW^\top = D$ is diagonal, but not equal to the identity matrix, we can rescale W and S to obtain orthogonality: $WSW^\top = WD^{-1}(DSD)(WD^{-1})^\top$ where WD^{-1} is orthogonal. Hence what is crucial in the algorithm is that the columns of W have disjoint support, not that they have unit norm. Normalization can be done a posteriori.

The matrix W is updated row-wise. For each row, among the r possible positions for the non-zero entry, we need to identify the one leading to the smallest error. To do this, for each possibility $k = 1, ..., r$, we compute the best value of $W(i,k)$ while the other entries of the row are set to zero, and the best possibility is kept to update $W(i,:)$. By expanding the Frobenius norm in the objective function, the one-variable subproblem in variable $W(i,k)$ is:

$$\begin{aligned} \min_{W(i,k) \geq 0} \; & (X(i,i) - (W(i,k)S(k,k)W(i,k))^2 \\ & + 2\sum_{i \neq j} (X(i,j) - W(i,k)SW(j,:))^2 + \text{constants}. \end{aligned} \tag{3}$$

Solving (3) can be achieved by finding the minimizer of the fourth-order polynomial $az^4 + bz^2 + cz$, where $a = S(k,k)^2$, $b = 2\left(\sum_{j \neq i}^n (W(j,:)S(:,k))^2 - S(k,k)X(i,i)\right)$, and $c = -4\sum_{j \neq i}^n X(i,j)W(j,:)S(:,k)$. If a, b and c are available, it is possible to solve (3) in $\mathcal{O}(1)$ operations, by computing the different extrema of (3) with Cardano's method and hence identify the best value for $W(i,k)$. The computation of the coefficient b costs $\mathcal{O}(nr)$ which leads to a total of $\mathcal{O}(n^2r^2)$ operations for a run over the nr entries of W. However, this complexity can be significantly improved. In our implementation, W is stored as two vectors w and v of length n where $v(i) \in \{1, ..., r\}$ stores the position of the non-zero element of $W(i,:)$ and $w(i) = W(i, v(i))$. Moreover, by precomputing the quantity $\sum_{j \neq i}^n (W(j,:)S(:,k))^2$ for all k and by taking into account that the data matrix X has $\operatorname{nnz}(X)$ non-zero entries, the update of W can be done in $\mathcal{O}(\text{nnz}(X)r + nr)$ operations; see Algorithm 1 for the details.

Update of S Let us consider the unconstrained problem $\min_S f(S)$ where $f(S) := 1/2\|X - WSW^\top\|_F^2$. The optimality conditions are given by

$$\nabla_S f(S) = W^\top(WSW^\top - X)W = 0.$$

Since $W^\top W = I$, the optimal solution for S actually has a closed form: $S^* = W^\top XW$, where S^* automatically satisfies the constraints: S^* is nonnegative since X and W are, and is symmetric since X is. By using the sparse storage detailed above for W, the quantity $W^\top XW$ can be computed in $\mathcal{O}(\text{nnz}(X))$ operations; see Algorithm 1, which we will refer as CD for coordinate descent.

Initialization In the noiseless case, we have $X = WSW^\top$, where $W^\top$ is separable, that is, there exists an index set $\mathcal{K}$ of cardinality r such that $W^\top(\mathcal{K}, :) = \text{diag}(z)$, with a positive vector $z \in \mathbb{R}^r_+$. and $\text{diag}(\cdot)$ is a diagonal matrix with z on its diagonal [7]. This follows from the fact that W is orthogonal. Therefore, separable NMF algorithms (see [5, Chapter 7]) will identify $\mathcal{K}$ such that

$$X(:, \mathcal{K}) = (WS)W(\mathcal{K}, :)^\top = (WS)\,\text{diag}(z).$$

Hence $X(:, \mathcal{K})$ is equal to WS, up to scaling. To recover W, we solve $\min_{W, W^\top W = I} \|X - X(:, \mathcal{K})W\|_F^2$ which can be done in closed form, as in orthogonal NMF [8]. Finally, we compute $S = W^\top XW$.

The successive projection algorithm (SPA) is one of the most popular separable NMF algorithm [9]. However, in the presence of noise, it is not very robust as it only extracts r columns of X that contain noise. To address this issue, smoothed SPA (SSPA) [10] relies on the assumption that there are multiple data points close to each column of WS. Hence it uses p of columns of X to be averaged to better approximate WS. In OtrisymNMF, we are in a very favorable situation since all columns of X are close to some of WS, up to scaling, since W is orthogonal, and we can therefore use a large value for p, up to n/r, we will use $p = \lfloor 0.2\frac{n}{r} \rfloor$.

Algorithm 1: CD for OtrisymNMF (2)

```
Input: $X \in \mathbb{R}^{n\times n}_+$, $r \in \mathbb{N}$, $w \in \mathbb{R}^n_+$ and
       $v \in \{1, \dots, r\}^n$ for $W \in \mathbb{R}^{n\times r}$, $S \in \mathbb{R}^{r\times r}$.
Output: $W \in \mathbb{R}^{n\times r}_+$ as two vectors $w \in \mathbb{R}^n_+$,
       $v \in \{1, ..., r\}^n$ and $S \in \mathbb{R}^{r\times r}$ s.t. $S = S^\top$.
1: % Precomput. of $\sum_{j=1}^n (W(j,:)S(:,k))^2$ $\forall k$ in $\mathcal{O}(nr)$
2: $p \leftarrow 0^r$
3: for $k = 1$ to $r$ and $j = 1$ to $n$ do
4:     $p_k \leftarrow p_k + w_j^2 S(v_j, k)^2$
5: end for
6: % Update of W (given S) in $\mathcal{O}(nnz(X)r + nr)$
7: for $i = 1$ to $n$ do
8:     $f^*, x^*, k^* \leftarrow +\infty$, None, None
9:     for $k = 1$ to $r$ do
10:        $a \leftarrow S(k,k)^2$; $c \leftarrow 0$
11:        $b \leftarrow 2\left(p_k - w_i^2 S(v_i,k)^2 - S(k,k)X(i,i)\right)$
12:        for $j = 1$ to $n$, $j \neq i$ and $X(i,j) \neq 0$ do
13:            $c \leftarrow c - 4\left(w_p X(i,j) S(v_j, k)\right)$
14:        end for
15:        $x \leftarrow \arg\min ax^4 + bx^2 + cx$ %Cardano in $\mathcal{O}(1)$
16:        if $ax^4 + bx^2 + cx < f^*$ then
17:            $f^*, x^*, k^* \leftarrow ax^4 + bx^2 + cx,\ x,\ k$
18:        end if
19:    end for
20:    % Update of p before the update of the ith row of W
21:    for $k = 1$ to $r$ do
22:        $p_k \leftarrow p_k - w_i^2 S(v_i,k)^2 + x^{*2} S(k^*,k)^2$
23:    end for
24:    % Update of the ith row of W
25:    $w_i, v_i \leftarrow x^*, k^*$
26: end for
27: % Normalization of the columns of W in $\mathcal{O}(n)$
28: for $k = 1, \dots, r$ do $q_k = \sum_{i, v_i = k} w_i^2$
29: for $i = 1, \dots, n$ do $w_i \leftarrow \frac{w_i}{\sqrt{q_{v_i}}}$
30: % Update of S (given W) in $\mathcal{O}(nnz(X))$
31: $S \leftarrow 0^{r\times r}$
32: for $(i,j) \in \{1, ..., n\} \times \{1, ..., n\}$ s.t. $X(i,j) \neq 0$ do
33:     $S(v_i, v_j) \leftarrow S(v_i, v_j) + w_i w_j X(i,j)$
34: end for
```

4. NUMERICAL EXPERIMENTS

All experiments were performed on an i7 processor with a clock frequency of 2.80GHz. The code will be made available from `https://github.com/Alexia1305/SymTriNMF`.

To evaluate the results, we use accuracy, which is calculated as the percentage of nodes correctly classified. Let π_k be a boolean vector such that $\pi_k[i]$ = true if and only if node i belongs to cluster k. Furthermore, by defining π'_k as the cluster k found by an algorithm, the accuracy is defined as:

$$\textit{Accuracy} = \max_{P \in \text{Permutations}} \frac{1}{n} \sum_{k=1:r} (\pi'_k[P] \wedge \pi_k).$$

We will calculate the average accuracy. We will report the score rate defined as the number of instances when the algorithm perfectly clusters all nodes. We will also use the relative error $\|X - WSW^\top\|_F / \|X\|_F$.

4.1. Synthetic data

To generate a noisy synthetic X, we proceed as follows. A random matrix W is created such that $W^\top W = I$. For that, a non-zero element is randomly chosen for each row (using `rand`), and it is enforced that no cluster is empty. Then, the columns are orthogonalized. We then create a sparse random

S sparse (using `sprand`, density=0.3) with $S(k,k) = 1$ for all k. For the noise, we generate N with the Gaussian distribution (using `randn`) and let

$$X = WSW^\top + \epsilon N \frac{\|WSW^\top\|_F}{\|N\|_F},$$

where ϵ is the noise level.

Comparing initializations Before comparing Algorithm 1 to existing algorithms for OtrisymNMF, let us show that SSPA is in fact a good initialization strategy . For each value of ϵ, we generate 100 matrices X as above, with $n = 200$ and $r = 8$, and run the algorithms. Four initializations are compared: random, k-means, SPA and SSPA initializations. K-means is a clustering method (we use [11]). K-means is used to find the non-zero elements of W, which are then normalized. We stop the algorithms after 1000 iterations or when the relative error has not decreased by at least 10^{-5} between two iterations.

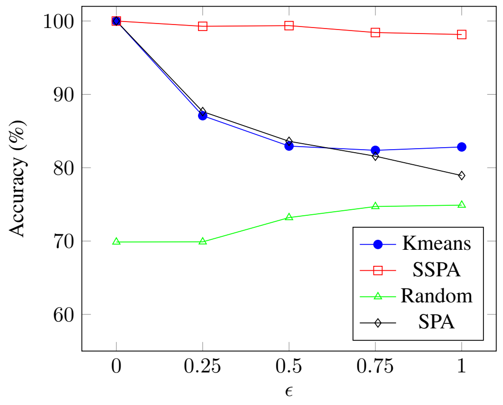

Fig. 1: Average accuracy for 100 random matrices.

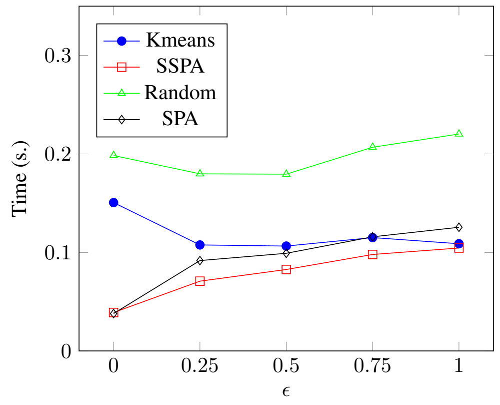

Fig. 2: Average computing time for 100 random matrices.

Figures 1, 2, 3 and 4 show the average accuracy, average computing time, average relative error, and average score rate.

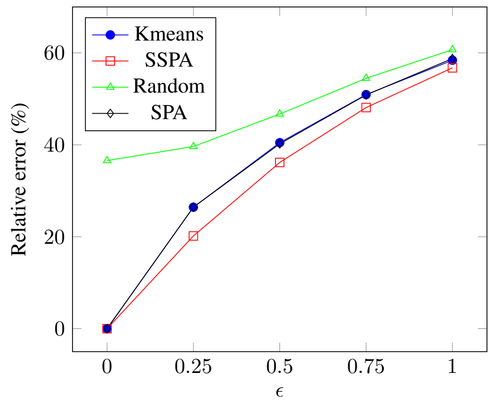

Fig. 3: Average relative error for different initializations as a function of noise.

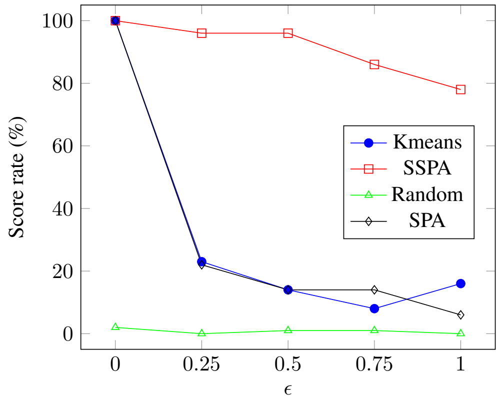

Fig. 4: Average success rate for 100 random matrices.

The results show that SPA, SSPA, and kmeans provide the exact solution in the noiseless case ($\epsilon = 0$). However, SSPA yields the best results and proves to be much more robust to noise with a success rate above 75% with $\epsilon = 1$. SSPA also enables faster convergence towards the solution.

Comparison with the MU from [4] Next, we compare our algorithm for OtrisymNMF with the multiplicative updates (MU) proposed in [4]. Algorithm 1 is initialized with SSPA. For the MU method, we also use SSPA, for a fair comparison. However, the zero elements of W are replaced by a small non-zero value, otherwise the MU cannot modify them. At the end of the algorithm, entries of W that are very close to 0 are replaced by 0. Both methods are stopped if they have converged (relative error decreased by less than 10^{-5}) or if they exceed a maximum time of 2 seconds.

The results are provided in Table 1. The MU converge faster than Algorithm 1, one hundredth of a second versus one-tenth of a second on average. However, Algorithm 1 more frequently identifies the groundtrue cluster. With $\epsilon = 1$, Algorithm 1 finds the exact classification 70 times out of 100, versus 47 for the MU.

Table 1: MU vs Algorithm 1 on noisy data.

Noise	ϵ	0	0.25	0.5	0.75	1
Score rate	MU	99%	96%	82%	77%	47%
	CD	**100%**	**96%**	**94%**	**90%**	**70%**
Average accuracy	MU	99.91%	99.57%	98.72%	98.53%	95.98%
	CD	**100%**	**99.43%**	**99.27%**	**99.03%**	**96.6%**
Time (s.)	MU	0.025	0.006	0.007	0.006	0.006
	CD	0.066	0.093	0.109	0.129	0.127

4.2. Real data: Classification of facial images

Let us compare four clustering algorithms: our CD algorithm for OtrisymNMF (Algorithm 1), OtrisymNMF with MU [4], the alternating optimization algorithm for ONMF [8], and K-means. We use a subset of the CBCL data set of facial images (260 images with 19×19 pixels each). ONMF factorizes $X \approx WH$, where $W \geq 0$, $H \geq 0$, and $HH^\top = I$.

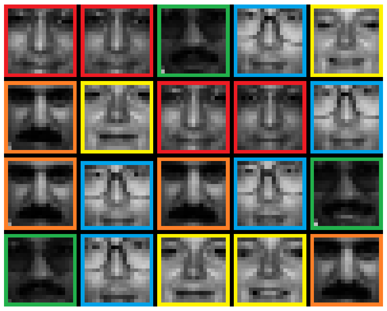
Fig. 5: Example of a data set with $r = 5$.

For different values of r, we first construct the matrix A by randomly picking the images of r selected persons (note that the number of images per person is not necessarily the same). Each column of A contains one vectorized facial image. We randomly shuffle the order of the images. An example of a dataset with $r = 5$ is shown on Figure 5. We then run the algorithms initialized with SSPA on the symmetric matrix[1] $X = A^\top A$ to find the r clusters of faces, i.e., faces of the same individuals, with a time limit of 40 s., and a minimum decrease in the relative error of 10^{-7} between two iterations. We repeat the test 10 times for each value of r (each time, selecting a different subset of r persons).

Table 2 provides the mean and standard deviation of the accuracies for $r = 2, 5, 10, 20, 30, 40, 50$, along with the percentage of tests where the algorithm provided the best solution in terms of accuracy. We observe that Algorithm 1 has the best average ranking, performing better than the MU for OtrisymNMF, ONMF and K-means.

Comparison with other matrix factorizations In this section, we compare OTriSymONMF with two classic factorization models: ONMF and SymNMF. We are interested in the expressivity of these models, that is, which models reconstruct the data the best for a fixed number of parameters. ONMF was described in the previous section, while SymNMF is the factorization $X \approx WW^\top$ with $W \geq 0$. For SymNMF, we use a coordinate descent algorithm from [12].

[1]We could compute such a similarity matrix in different ways and analyze the effect, but this is out of the scope of this paper.

Table 2: Accuracy of the clustering on 50 random selections with r groups of the CBCL dataset.

Average and standard deviation of the accuracy (%)				
r	CD	MU	ONMF	K-means
2	$\mathbf{94.73} \pm 11.40$	94.54 ± 11.17	$\mathbf{94.73} \pm 11.40$	93.41 ± 12.83
5	$\mathbf{76.63} \pm 14.92$	74.33 ± 14.98	75.93 ± 14.78	73.59 ± 15.20
10	70.70 ± 8.45	69.49 ± 7.97	$\mathbf{70.93} \pm 8.22$	66.39 ± 8.26
20	$\mathbf{63.52} \pm 4.85$	59.55 ± 4.48	62.49 ± 4.68	60.82 ± 6.10
30	$\mathbf{60.18} \pm 3.56$	55.79 ± 3.20	59.41 ± 3.40	56.8 ± 4.43
40	$\mathbf{55.29} \pm 2.81$	52.49 ± 2.62	54.78 ± 2.51	55.24 ± 3.65
50	$\mathbf{53.56} \pm 0.69$	51.11 ± 0	52.59 ± 0	53.19 ± 3.00
Percentage of runs where the algorithm achieves the highest accuracy compared to other algorithms (%)				
r	CD	MU	ONMF	K-means
2	**96**	94	**96**	94
5	**64**	36	54	57
10	**60**	32	54	32
20	**62**	4	30	30
30	**62**	0	29	24
40	46	2	18	**48**
50	**58**	0	8	46

The number of parameters of each model is:
OTriSymNMF: $r(r-1)/2 + 2n$, namely $r(r-1)/2$ for the r-by-r symmetric S, and $2n$ for W as it requires the column index and the value of the non-zero element for each row.
ONMF: $nr + 2n$, namely nr elements in W and $2n$ in H as for W in OtrisymNMF.
SymNMF: nr, namely nr elements in W.

For different values of r, we compute the error of the factorizations for two datasets: CBCL and TDT2. The CBCL dataset A consists of 2429 images with 19×19 pixels each, and we represent the interaction between the pixels as before, $X = AA^\top$. Figure 6 provides the visualization of W (that is,

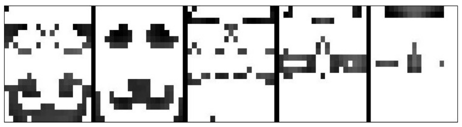
Fig. 6: The 5 columns of the W obtained with OtrisymNMF CD are reshaped as images and displayed in a 19×19 grid.

reshaped columns of W as images) obtained with OtrisymNMF (Algorithm 1) with $r = 5$. It shows 5 communities of pixels, that correspond to a coherent spatial clustering.

TDT2 is a sparse dataset $B \in \mathbb{R}^{209 \times 909}$, with 99.37% of the entries equal to zero (we used a subsampled version for computational reasons). The entry $B(i, j)$ represents the number of occurrences of the word i in the document j. We construct $X = BB^\top \in \mathbb{R}^{209 \times 209}$, which contains the interaction between the words.

Figures 7 and 8 show the relative error as a function of the number of parameters for CBCL and TDT2. For CBCL, OtrisymNMF with Algorithm 1 has the lowest error between 700 and 1400 parameters. For TDT2, it has the lowest error between 1000 and 5500 parameters. ONMF is the least expressive model, regardless of the number of parameters, while symNMF tends to outperform OtrisymNMF when the number of parameters is large. The reason is that the orthogonality constraint makes OtrisymNMF have a hard time reducing the error below the variability present in each cluster/community.

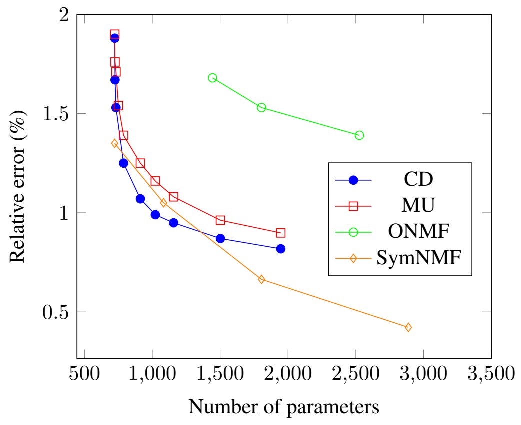

Fig. 7: CBCL: error vs number of parameters for OtrisymNMF ($r = 2, 3, 5, 8, 12, 20, 30, 40, 50$), ONMF ($r = 2, 3, 5$) and SymNMF ($r = 2, 3, 5, 8$).

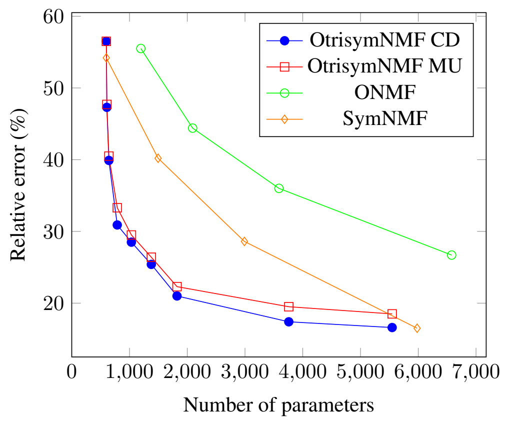

Fig. 8: TDT2: relative error vs number of parameters for OtrisymNMF ($r = 2, 5, 10, 20, 30, 40, 50, 80, 100$), ONMF and SymNMF ($r = 2, 5, 10, 20$).

5. CONCLUSION

In this paper, we investigated OtrisymNMF (2). We proved the uniqueness of the factorization and proposed a new coordinate descent method that exploits the properties of this problem to solve it efficiently; see Algorithm 1. We also proposed a new effective initialization strategy based on smooth SPA. We showed on synthetic and real data that our algorithm compares favorably with the state of the art, providing more accurate solutions than the MU from [4], having better clustering accuracy than ONMF and K-means, and being more expressive than ONMF and symNMF for a large range of parameter numbers.

6. REFERENCES

[1] W.-S. Chen, K. Xie, R. Liu, and B. Pan, "Symmetric nonnegative matrix factorization: A systematic review," *Neurocomputing*, p. 126721, 2023.

[2] A. Berman and N. Shaked-Monderer, *Completely positive matrices*, World Scientific, 2003.

[3] X. Fu, K. Huang, N. D. Sidiropoulos, and W.-K. Ma, "Nonnegative matrix factorization for signal and data analytics: Identifiability, algorithms, and applications.," *IEEE Signal Process. Mag.*, vol. 36, pp. 59–80, 2019.

[4] C. Ding, T. Li, W. Peng, and H. Park, "Orthogonal nonnegative matrix t-factorizations for clustering," in *ACM SIGKDD*, 2006, vol. 2006, pp. 126–135.

[5] N. Gillis, *Nonnegative Matrix Factorization*, SIAM, Philadelphia, 2020.

[6] S. Hoseinipour, M. Aminghafari, and A. Mohammadpour, "Orthogonal parametric non-negative matrix tri-factorization with α-divergence for co-clustering," *Expert Syst. Appl.*, vol. 231, pp. 120680, 2023.

[7] S. Arora, R. Ge, R. Kannan, and A. Moitra, "Computing a nonnegative matrix factorization–provably," in *ACM Symposium on Theory of Computing*, 2012.

[8] F. Pompili, N. Gillis, P.-A. Absil, and F. Glineur, "Two algorithms for orthogonal nonnegative matrix factorization with application to clustering," *Neurocomputing*, vol. 141, pp. 15–25, 2014.

[9] M. Araújo et al., "The successive projections algorithm for variable selection in spectroscopic multicomponent analysis," *Chemom. Intell. Lab. Syst. 57:65–73*, 2001.

[10] N. Nadisic, N. Gillis, and C. Kervazo, "Smoothed separable nonnegative matrix factorization," *Linear Algebra and its Applications*, vol. 676, pp. 174–204, 2023.

[11] JuliaStats, "Clustering.jl documentation," 2022.

[12] A. Vandaele, N. Gillis, Q. Lei, K. Zhong, and I. Dhillon, "Efficient and non-convex coordinate descent for symmetric nonnegative matrix factorization," *IEEE Transactions on Signal Processing*, vol. 64, pp. 1–1, 11 2016.