On Rank Selection for Nonnegative Matrix Factorization

Srinivas Eswar* , Koby Hayashi†, Benjamin Cobb‡,
Ramakrishnan Kannan§, Grey Ballard¶, Richard Vuduc‡, Haesun Park‡
*Argonne National Laboratory, † Pacific Northwest National Laboratory, §Oak Ridge National Laboratory
¶Wake Forest University, ‡Georgia Institute of Technology

2024 IEEE International Conference on Big Data (BigData) | 979-8-3503-6248-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/BigData62323.2024.10825324

***Abstract*—Rank selection, i.e. the choice of factorization rank, is the first step in constructing Nonnegative Matrix Factorization (NMF) models. It is a long-standing problem which is not unique to NMF, but arises in most models which attempt to decompose data into its underlying components. Since these models are often used in the unsupervised setting, the rank selection problem is further complicated by the lack of ground truth labels. In this paper, we review and empirically evaluate the most commonly used schemes for NMF rank selection.**

***Index Terms*—Rank selection, Nonnegative Matrix Factorization**

I. Introduction

In data analytics and machine learning, one often encounters large datasets organised into a matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$. Many useful methods begin by approximating this matrix as the product of low-rank factors $\mathbf{X} \approx \mathbf{W}\mathbf{H}^\intercal$ with $\mathbf{W} \in \mathbb{R}^{m\times k}$, $\mathbf{H} \in \mathbb{R}^{n\times k}$, and $k \ll \min(m, n)$. These linear models are used for simpler representations, interpretability, or simply to enable computations on massive data and are a workhorse of modern data analytics [1]. Nonnegative Matrix Factorization (NMF) is a linear model which enforces element-wise nonnegativity constraints on both the factor matrices, i.e. $\{\mathbf{W}, \mathbf{H}\} \geq 0$. The extraordinary effectiveness of NMF in decomposing nonnegative inputs into its constituent parts has lead to its use in many different domains including analytical chemistry, earth sciences, text mining, amongst others [1].

Another appealing feature of linear models is the relatively low number of hyperparameters associated with the model. Typically, only the factorization rank is needed to fit a linear model, with the exception of some regularization hyperparameters. This is appealing in practice since there are fewer knobs to turn during exploratory data analysis. Finally, limiting the number of hyperparameters makes it more tractable to ensure reproducibility in linear models [2].

This manuscript has been authored by UT-Battelle, LLC under Contract No. DEAC05-00OR22725 with the U.S. Department of Energy. This material was based upon work supported by the U.S. Department of Energy, Office of Science under Contract DE-AC02-06CH11357 at Argonne National Laboratory. This work is supported by the National Science Foundation under Grant Nos. OAC-2106920 and OAC-2106738. The publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

In this paper we study the choice of the factorization rank k for NMF. There are subtle differences between the rank, nonnegative rank, and the number of components in the linear models we are trying to fit. Unlike numerical rank, nonnegative rank is NP-hard to compute [3], and the number of components is often derived from expert insight [1]. In this paper, we are concerned with the third category of the choice of k, where we try to select the number of components in a data matrix without overfitting to the noise present in the data.

We consider multiple metric-based approaches, which involve computing NMF for various ranks, computing a metric for each of those factorizations, and choosing the rank that optimises the metric. These methods include using residual error, Bayesian Information Criteria, and the Core Consistency Diagnostic. We also consider methods designed for the unconstrained case that use the singular values of the input matrix, and Cross Validation (CV) tailored for NMF. We provide a detailed description of the implementations along with their assumptions and computational complexity in Section III. Extensive numerical experiments are performed and presented in Section IV to evaluate the efficacy of the various methods. We believe that such a study is necessary to develop reliable, reproducible, and robust data analytics pipelines.

Based on the analyses of this paper, we conclude that Singular Value Thresholding (SVT) based methods and CV are the most promising techniques for use in practice. While the singular value based methods are not designed for nonnegativity constraints, they are relatively cheap to employ and work surprisingly well. Cross validation techniques have more theoretical justification for NMF and tend to work more robustly than singular value based methods, but they are very computationally expensive. We discuss these tradeoffs and future directions of inquiry in Sections V and VI.

II. Preliminaries

A. Notation

We use regular fonts for scalars (e.g., k), bold lowercase for vectors (e.g., $\mathbf{x}$) and bold uppercase for matrices (e.g., $\mathbf{A}$). For $\mathbf{A}$, its ith row is $\hat{\mathbf{a}}_i$ and jth column is $\mathbf{a}_j$. $\mathbf{I}_n$ represents the $n\times n$ identity matrix and $\mathbf{1}$ is a matrix of all ones of appropriate size. The $\mathrm{diag}(\cdot)$ operator extracts the diagonal elements of a matrix or forms a diagonal matrix from a vector depending on the context. $\mathbf{A} * \mathbf{B}$ represents the element-wise product between two compatible matrices. The comparison $\mathbf{X} \geq 0$

is performed in an element-wise manner. $\mathrm{nnz}(\cdot)$ counts the number of nonzero entries in an vector or matrix.

B. NMF Preliminaries

Formally, the NMF optimization problem is

$$\min_{\mathbf{W}\geq 0,\mathbf{H}\geq 0} \left\|\mathbf{X}-\mathbf{W}\mathbf{H}^{\mathsf{T}}\right\|_F^2 \tag{1}$$

with a nonnegative input matrix $\mathbf{X} \in \mathbb{R}_+^{m\times n}$ and low-rank matrices $\mathbf{W} \in \mathbb{R}_+^{m\times k}$ and $\mathbf{H} \in \mathbb{R}_+^{n\times k}$ with $k \ll \min(m,n)$. We assume that $\mathbf{X}$ is organized in a manner in which its columns correspond to different data samples and the rows to the features. The matrix $\mathbf{W}$ represents the "basis" matrix, that is the atomic parts used to generate the columns of $\mathbf{X}$. Rows of $\mathbf{H}$ describe how to combine the columns of $\mathbf{W}$ and are the k-dimensional embeddings of the data samples.

From the above discussion, it is clear that we assume that our data is linearly generated as $\mathbf{X}_{\text{true}} = \mathbf{W}\mathbf{H}^{\mathsf{T}}$ and then perturbed via some noise term $\mathbf{N}$. If we have additive Gaussian noise, then the optimization via Eq. (1) is appropriate to recover the factors. Other formulations for the objective are present for different noise models, see the references in [1, Chapter 5.1], but we limit our study to the most common Gaussian case. Conditions when these factors are unique have been studied but are beyond the scope of this work.

Finally, we want to stress that NMF is typically used in an unsupervised context. That is no labels are associated with the different columns of $\mathbf{X}$. Therefore, adapting hyperparameter optimization strategies from the supervised learning context is not direct and need to be appropriately modified.

C. NMF Algorithms

An example of a typical software for NMF is Parallel Low-rank Approximation with Nonnegativity Constraints (PLANC) [4], [5]. PLANC is designed to solve the optimization problem Eq. (1) for dense or sparse, nonnegative input matrices $\mathbf{X}$. It contains various algorithms for computing NMF, including both Block Coordinate Descent (BCD) and direct optimization variants. In BCD-style algorithms, the matrices $\mathbf{W}$ and $\mathbf{H}$ are divided into blocks and updated in sequence till some stopping criteria is satisfied. For example, in the popular two-block BCD variant Alternating Nonnegative Least Squares (ANLS), $\mathbf{W}$ is kept fixed and $\mathbf{H}$ is updated followed by keeping the updated $\mathbf{H}$ fixed and updating $\mathbf{W}$ [6]. In the direct optimization methods, both $\mathbf{W}$ and $\mathbf{H}$ are updated simultaneously using information like the current gradient, Hessian, and a various other factors [7]. Matrix multiplication is the computational bottleneck of NMF with most algorithms needing to compute $\mathbf{W}^{\mathsf{T}}\mathbf{W}$, $\mathbf{H}^{\mathsf{T}}\mathbf{H}$, $\mathbf{W}^{\mathsf{T}}\mathbf{X}$, and $\mathbf{H}^{\mathsf{T}}\mathbf{X}^{\mathsf{T}}$.

III. Rank Selection Methods

We describe rank selection methods in Constrained Low Rank Approximation (CLRA) useful for the NMF case.

A. Metric Based

The most common rank-selection method is the "scree test" [8]. The residual or fit is plotted versus the number of components in order to identify the "elbow" in the graph. For CLRA methods, this graph should be monotonically decreasing if a good approximation algorithm is used. Unfortunately, this test is rather subjective since scree plots can have multiple "elbows" making it difficult to decide the correct rank.

A prototypical scree test is shown in Algorithm 1. For the standard scree test, the metric computed will be the fit or residual $\mathbf{m}(k) = \left\|\mathbf{X}-\mathbf{W}_k\mathbf{H}_k^{\mathsf{T}}\right\|_F$. Let SELECTBEST be the procedure by which we select the "elbow" from $\mathbf{m}$. However, this procedure is not well defined for the standard scree test.

We estimate the cost of running a scree test with regular residuals as the metric used. The time is broken down into three categories; 1) computing the approximations, 2) computing the metrics, and 3) selecting the best k from the metrics.

$$T_{\text{scree}} = T_{\text{approx}} + T_{\text{metric}} + T_{\text{select}}.$$

Let T_{NMF} be the time taken to compute an NMF approximation for a given value of k. In general, this time is dependent on the size of the input matrix and the rank chosen but for the purposes of rank selection we can approximate it by $k_{\max}$ for all $k \in [k_{\min}, k_{\max}]$. This is reasonable as the dependence is a polynomial in k and our approximate cost can only differ by at most a constant. Computing the NMF is an iterative process which costs approximately

$$T_{\text{NMF}} = O\left(n_{\text{iters}} \cdot \left(4mnk_{\max} + (m+n)\,k_{\max}^2 + T_{\text{upd}}\right)\right),$$

for n_{iters} of updating the factor matrices $\mathbf{W}$ and $\mathbf{H}$. T_{upd} is the time taken to update both factor matrices which is algorithm dependent. The rest of the costs arise from computing $\mathbf{W}^{\mathsf{T}}\mathbf{X}, \mathbf{H}^{\mathsf{T}}\mathbf{X}^{\mathsf{T}}, \mathbf{W}^{\mathsf{T}}\mathbf{W}$, and $\mathbf{H}^{\mathsf{T}}\mathbf{H}$ (see Section II-C).

Another assumption is that the residuals are returned along with the factor matrices on a call to NMF. Therefore, for a regular scree test there is no overhead of computing the metric. The time taken for selecting the "elbow" is unclear but it is typically done by scanning over $\mathbf{m} \in \mathbb{R}^{k_{\max}-k_{\min}}$. We shall assume it is approximately linear in the number of candidate ranks $n_{\text{ranks}} = k_{\max} - k_{\min}$. Putting all this together we get the cost of performing a scree test as,

$$T_{\text{scree}} = n_{\text{ranks}} n_{\text{seeds}} T_{\text{NMF}} + O(n_{\text{ranks}}).$$

Similar methods exist with different metrics being used in place of the fit. One family of methods is based on the Bayesian Information Criteria (BIC) [9]. As noted above, the residual for CLRA models are typically monotonically decreasing which may result in overfitting when rank is increased. In order to resolve this monotonic decrease, a penalty term for the number of parameters in the model is introduced. For large ranks, this penalty term will dominate the residual and make it easier to select the rank with the minimum BIC criteria. This removes the subjectivity of "elbow" selection present in the scree test as well. A commonly used criteria is $\text{BIC}_1(k) = \log\left(\|\mathbf{X}-\mathbf{X}_k\|_F^2\right) + k\left(\frac{m+n}{mn}\right)\log\left(\frac{mn}{m+n}\right)$. Computing this criteria can be done in constant time given

Algorithm 1 A general scree test framework

procedure SCREETEST($\mathbf{X}, k_{\min}, k_{\max}, n_{\text{seeds}}$)
Input: $\mathbf{X} \in \mathbb{R}^{m \times n}$ the input data matrix, $[k_{\min}, k_{\max}]$ is the range of ranks to check, and n_{seeds} is the number of restarts per k for NMF.
Output: k_{opt} is the rank to use for the approximation.
▷ Fit an NMF model for each k.
  for $k \in [k_{\min}, k_{\max}]$ **do**
    for $s \in 1, \ldots, n_{\text{seeds}}$ **do**
      $[\mathbf{W}_s, \mathbf{H}_s] = \text{NMF}(\mathbf{X}, k)$.
    end for
    ▷ Select the NMF model with least error for k.
    $[\mathbf{W}_k, \mathbf{H}_k] = \text{BESTMODEL}(\{(\mathbf{W}_s, \mathbf{H}_s)\})$.
  end for
  ▷ Compute metrics for each of the different models.
  $\mathbf{m} = \text{COMPUTEMETRIC}(\{(\mathbf{W}_k, \mathbf{H}_k)\})$.
  $k = \text{SELECTBEST}(\mathbf{m})$.
  Return: k.
end procedure

the residual from the NMF approximations. The running time for a BIC based scree test is the same as a standard scree test.

$$T_{\text{BIC}} = n_{\text{ranks}} n_{\text{seeds}} T_{\text{NMF}} + O(n_{\text{ranks}}).$$

The Core Consistency Diagnostic (CORCONDIA) is a method for determining the number of components to select in a CANDECOMP/PARAFAC (CP) model for tensor data which can be extended to the nonnegative case [10]. The intuition behind this approach is to determine how well a CP decomposition fits the data when compared to a Tucker decomposition with the same factors. The Tucker core is computed using the factor matrices returned from CP and then compared to a tensor with 1 on its superdiagonal. CORCONDIA was shown to discern the appropriate number of CP components on multiple real world data sets [10]. This method can be specialised to the matrix case where we measure how independent or non-overlapping the different components are that are discovered for each rank. The CORCONDIA measure will be calculated for a number of different rank choices and the rank with the highest CORCONDIA score will be selected.

The matrix version of CORCONDIA works as follows. Given factor matrices $\mathbf{W}_k \in \mathbb{R}^{m \times k}$ and $\mathbf{H}_k \in \mathbb{R}^{n \times k}$, we form the "core matrix" $\mathbf{G}_k \in \mathbb{R}^{k \times k}$ as $\mathbf{G}_k = \mathbf{W}_k^{\dagger} \mathbf{X} \left(\mathbf{H}_k^{\dagger}\right)^{\mathsf{T}}$. This matrix is compared to the "ideal" core matrix $\mathbf{I}_k$, which represents an exact factorization of the data matrix. Then the metric for the matrix case, is derived from CORCONDIA as $\mathbf{m}(k) = \|\mathbf{G}_k - \mathbf{I}_k\|_F^2 / \|\mathbf{I}_k\|_F^2$. We can also modify the core matrix calculation above to enforce nonnegativity of $\mathbf{G}_k$ via the following least squares problem $\min_{\mathbf{G}_k \geq 0} \left\|\mathbf{X} - \mathbf{W}_k \mathbf{G}_k \mathbf{H}_k^{\mathsf{T}}\right\|_F^2$, but we work with the unconstrained version here.

Computing the CORCONDIA for a particular k involves two different least squares solves costing $O\left((m+n)\,k^2\right)$ followed by a norm calculation costing $O(k^2)$. Typically, we shall compute the CORCONDIA scores only for the approximation with the smallest residual among the different seeds for a given rank. Finally, we select the rank that maximizes this score. Thus using a scree test with CORCONDIA will cost approximately

$$T_{\text{C}} = n_{\text{ranks}} n_{\text{seeds}} T_{\text{NMF}} + O\left(n_{\text{ranks}} \left((m+n)\,k_{\max}^2\right)\right) + O(n_{\text{ranks}}).$$

It should be stressed that even with all the different metrics, the majority of the time is spent in performing the $n_{\text{ranks}} n_{\text{seeds}}$ different factorizations.

Algorithm 2 A typical SVT procedure

procedure SVT($\mathbf{X}, p$)
Input: $\mathbf{X} \in \mathbb{R}^{m \times n}$ the input data matrix, and p are the optional parameters.
Output: k_{opt} is the rank to use for the approximation.
▷ Compute the Singular Value Decomposition (SVD).
  $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \text{SVD}(\mathbf{X})$.
  ▷ Compute threshold (may have optional parameters).
  $\theta = \text{FINDTHRESHOLD}(\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}, p)$.
  ▷ Compute the best rank.
  $k = \sum_i \mathbf{1}\{\sigma_i > \theta\}$.
  Return: k.
end procedure

B. Perturbation Analysis

Another class of rank-selection methods is based upon perturbation analysis like the popular NMFk [11]–[13]. NMFk makes multiple perturbations of the input $\mathbf{X}$. Then for every rank, NMFk factorizes each of the perturbed matrices and reorders the columns of each of the resulting $\mathbf{W}$s, using a variation of k-medians, to match them. The rows of $\mathbf{H}$ are ordered analogously to keep $\mathbf{W}\mathbf{H}^{\mathsf{T}}$ invariant and the elementwise median of the factors are computed. The cluster stability of the median factors via the Silhouette index is calculated. The rank with the largest separation between the relative error and silhoutte index is selected. Computing the median factors and Silhouette index for a particular k costs $O\left(n_{\text{per}} n_{\text{clus}} \left(k^3 + mk^2 + (m+n)k\right)\right)$, where n_{per} is the number of perturbations and n_{clus} is the number of iterations used in clustering [13]. The cost for NMFk is

$$T_{\text{NMFk}} = n_{\text{ranks}} n_{\text{per}} T_{\text{NMF}} + O\left(n_{\text{per}} n_{\text{clus}} \left(k_{\max}^3 + mk_{\max}^2 + (m+n)k_{\max}\right)\right) + O(n_{\text{ranks}}).$$

C. SVD Thresholding Methods

The SVD has also been used to "denoise" the data matrix and estimate the signal rank. While this does not correspond directly to notion of nonnegative rank or components used in NMF, these methods are shown to work well in practice. These methods are known as SVT methods and are outlined in Algorithm 2. The heart of these approaches consist of the asymptotic analysis of a sequence of matrix recovery problems $\mathbf{X}_{m,n} = \mathbf{L}_{m,n} + \mathbf{N}_{m,n}$ as $m, n \to \infty$ proportionately, i.e. $\frac{m}{n}$ always remains approximately a fixed ratio. It is assumed that the $\mathbf{L}_{m,n}$ has a fixed rank k and the noise matrix comes from a particular distribution. Notice that our data $\mathbf{X}$ is a specific instance in this chain of matrices.

Gavish and Donoho study the white noise case where the noise matrix consists of uncorrelated mean-zero Gaussian entries of unknown variance [14]. Their threshold formula becomes $\theta_{\text{opt}} \approx \frac{4}{\sqrt{3}} \cdot \frac{\sigma_{\text{median}}}{\sqrt{n \cdot 0.6528}}$. The Parallel Analysis (PA) method of rank selection also falls in this category [15]. In PA, multiple copies of the input are generated by randomly permuting entries within each column of the input matrix. The rationale is that this permutation will destroy the low-rank signal but not affect the noise. An empirical distribution of the noise singular values from these copies is computed and only those singular values from the original data matrix are retained when they exceed a certain percentile of the noise singular values. Dobriban and Owen analyse this procedure and develop theory for PA [16]. They "derandomize" the method to compute only a single SVD of the original data matrix. Finally, Donoho et al. study the case of correlated noise [17]. They also show *finite-sample* optimality at reasonable problem sizes. While the last two methods' cost for computing the threshold is not as simple as Gavish and Donoho's, they are still constant for every singular value considered.

The SVT procedure starts by computing the SVD of the input data matrix. Using the empirical singular values and singular vectors, a threshold θ is computed. Only the components with singular values greater than this threshold are retained and thus the rank is determined. SVT-style methods cost approximately

$$T_{\text{SVT}} = T_{\text{SVD}} + O(n_{\text{ranks}}).$$

Only a single SVD of the input is needed, which costs $O(mn^2)$ flops for $m \geqslant n$.

D. Cross Validation

CV is a fundamental paradigm in data analysis to aid in model hyperparameter selection and to counter overfitting. It eliminates the need to rely on user inputs (for e.g., locating elbows) and reduces this part of the analysis to a numerical criterion. In a supervised setting, the basic idea is to partition labeled data into training and test sets, fit the model to the training, or "held-in", data and test the model error on the test, or "held-out", data. The idea of ℓ-fold cross validation is to partition the data into ℓ sets, and treat each of the sets in turn as the hold-out set.

The difficulty in holding out an entire sample of data for NMF, say column j of $\mathbf{X}$, is that it we can not fit all the model parameters. In this case row j of $\mathbf{H}$ will be left out. Similarly, leaving out a row i of $\mathbf{X}$ will exclude fitting row i of $\mathbf{W}$. The solutions for cross-validating NMF are inspired by similar methods for SVD [18], [19]. Wold cross-validates the rank of an SVD model by leaving out a set of matrix elements [18]. This holdout pattern ensures that no column or row of $\mathbf{X}$ is completely missing. Another approach for the SVD by Gabriel is to hold out a single entry of $\mathbf{X}$ at a time [19].

These methods can be applied directly to NMF by working with a masking matrix $\mathbf{M} \in \{0,1\}^{m \times n}$ with masking either corresponding to a single held out entry, as in the case of Gabriel hold outs, or multiple entries for Wold hold outs. The held-in entries correspond to 1 in the masking matrix and held out to 0 [20]. An ANLS-style NMF algorithm would proceed to solve the following "censored" least-squares problems,

$$\begin{aligned} \mathbf{H} &= \underset{\mathbf{H} \geq 0}{\operatorname{argmin}} \left\| \mathbf{M} * \left(\mathbf{X} - \mathbf{W}\mathbf{H}^{\mathsf{T}} \right) \right\|_F \\ \mathbf{W} &= \underset{\mathbf{W} \geq 0}{\operatorname{argmin}} \left\| \mathbf{M} * \left(\mathbf{X} - \mathbf{W}\mathbf{H}^{\mathsf{T}} \right) \right\|_F , \end{aligned} \tag{2}$$

where $*$ denotes the element-wise multiplication. Finally, the fit is calculated on the held out entries to evaluate the model $\left\| (\mathbf{1} - \mathbf{M}) * \left(\mathbf{X} - \mathbf{W}\mathbf{H}^{\mathsf{T}} \right) \right\|_F$. Owen and Perry [21] generalize the Gabriel method to hold out $r \times s$ elements of $\mathbf{X}$ at a time for both SVD and NMF in a method known as Bi-Cross-Validation (BiCV). Kanagal and Sindhwani designed structured masks to alleviate some of the computational bottlenecks associated with BiCV for NMF [22].

We start with analysing the cost of computing a single approximation of the factor matrices in the presence of held-out data. For the BCD approach to NMF we need to repeatedly perform the "censored" Nonnegative Least Squares (NLS) shown in Eq. (2). Let us consider the update for $\mathbf{H}$. Using the fact that $\mathbf{M}$ is Boolean, this problem reduces to using the following normal equations form [20]:

$$\mathbf{W}^{\mathsf{T}} \left(\mathbf{M} * \mathbf{W}\mathbf{H}^{\mathsf{T}} \right) = \mathbf{W}^{\mathsf{T}} \left(\mathbf{M} * \mathbf{X} \right). \tag{3}$$

Let us denote $\mathbf{B} = \mathbf{M} * \mathbf{X}$ to handle the RHS of Eq. (3). This can be precomputed, costing $O(mn)$ flops, and cached for all iterations. Let us look at Eq. (3) column by column. Unlike the regular NMF case we now have different LHS for each row j of $\mathbf{H}$ depending on the bitmask defined by column $\mathbf{m}_j$, $\left(\mathbf{W}^{\mathsf{T}} \mathsf{diag}\left(\mathbf{m}_j \right) \mathbf{W} \right) \hat{\mathbf{h}}_j = \mathbf{W}^{\mathsf{T}} \mathbf{b}_j$. Computing the RHS is a simple matrix multiplication, $\mathbf{W}^{\mathsf{T}}\mathbf{B}$, which costs $2mnk$ flops. Computing the n different LHS terms, each costing $2mk^2$ flops, results in a total of $2mnk^2$ operations. Similarly accounting for updating $\mathbf{W}$, we can see that each inner iteration of masked NMF takes approximately $4mnk^2 + 4mnk + T_{\text{upd}}$ operations. Notice the extra $O(mnk^2)$ term when compared to a regular NMF algorithm's inner iteration costs.

The alternating iteration continues until the objective function value is sufficiently small or the change in value is sufficiently small. The objective function satisfies the following:

$$\begin{aligned} &\left\| \mathbf{M} * \left(\mathbf{X} - \mathbf{W}\mathbf{H}^{\mathsf{T}} \right) \right\|_F^2 = \\ &\|\mathbf{M} * \mathbf{X}\|_F^2 - 2\mathsf{Tr}\left(\mathbf{W}^{\mathsf{T}} \left(\mathbf{M} * \mathbf{X} \right) \mathbf{H} \right) + \left\| \mathbf{M} * \mathbf{W}\mathbf{H}^{\mathsf{T}} \right\|_F^2 \end{aligned} \tag{4}$$

which can make computation more efficient compared to direct evaluation because the first term is constant with respect to $\mathbf{W}$ and $\mathbf{H}$ and can be precomputed and the second term involves the RHS matrix of Eq. (3) which can be re-used. The final term, $\mathbf{M} * \mathbf{W}\mathbf{H}^{\mathsf{T}}$ becomes the bottleneck computation. For every nonzero in $\mathbf{M}$, we need to compute the inner product of two k-length vectors costing $2\mathsf{nnz}\left(\mathbf{M} \right) k$ flops. For ℓ-fold CV, we can assume $\mathbf{M}$ has $\left(1 - 1/\ell \right) mn$ nonzeros.

Another approach for individual NMF in CV is via direct optimization. We need to compute the cost of forming the gradient, which is the computational bottleneck for most first

TABLE I: Comparison of the different rank-selection methods in terms of operations costs. The approximation and metric calculations are for a single low-rank approximation (see Section III).

Algorithm	No. of Approximations	Approximation Costs	Metric Costs
SVT	1	$O(mn^2)$	$O(1)$
Scree	$n_{\text{ranks}} \cdot n_{\text{seeds}}$	$O\left(n_{\text{iters}} \cdot \left(4mnk + (m+n)\,k^2 + T_{\text{upd}}\right)\right)$	$O(1)$
BIC	$n_{\text{ranks}} \cdot n_{\text{seeds}}$	$O\left(n_{\text{iters}} \cdot \left(4mnk + (m+n)\,k^2 + T_{\text{upd}}\right)\right)$	$O(1)$
CORCONDIA	$n_{\text{ranks}} \cdot n_{\text{seeds}}$	$O\left(n_{\text{iters}} \cdot \left(4mnk + (m+n)\,k^2 + T_{\text{upd}}\right)\right)$	$O\left((m+n)\,k^2\right)$
NMFk	$n_{\text{ranks}} \cdot n_{\text{per}}$	$O\left(n_{\text{iters}} \cdot \left(4mnk + (m+n)\,k^2 + T_{\text{upd}}\right)\right)$	$O\left(\left(k^3 + mk^2 + (m+n)\,k\right) n_{\text{clus}} n_{\text{per}}\right)$
CV (BCD)	$n_{\text{ranks}} \cdot n_{\text{seeds}} \cdot n_{\text{folds}}$	$O\left(n_{\text{iters}} \cdot \left(4mnk^2 + 4mnk + 2\left(1 - 1/n_{\text{folds}}\right)mnk + T_{\text{upd}}\right)\right)$	$O\left(mnk/n_{\text{folds}}\right)$
CV (Direct)	$n_{\text{ranks}} \cdot n_{\text{seeds}} \cdot n_{\text{folds}}$	$O\left(n_{\text{iters}} \cdot \left(8mnk + 4\left(1 - 1/n_{\text{folds}}\right)mnk + T_{\text{upd}}\right)\right)$	$O\left(mnk/n_{\text{folds}}\right)$

and second order direct optimization methods. Considering $f(\mathbf{W},\mathbf{H}) = \left\|\mathbf{M} * \left(\mathbf{X} - \mathbf{W}\mathbf{H}^{\mathsf{T}}\right)\right\|_F^2$, the gradient for each factor is given by,

$$\frac{\partial f}{\partial \mathbf{H}} = 2\left(\mathbf{W}^{\mathsf{T}}\left(\mathbf{M} * \mathbf{X}\right) - \mathbf{W}^{\mathsf{T}}\left(\mathbf{M} * \mathbf{W}\mathbf{H}^{\mathsf{T}}\right)\right) \tag{5}$$

$$\frac{\partial f}{\partial \mathbf{W}} = 2\left(\mathbf{H}^{\mathsf{T}}\left(\mathbf{M} * \mathbf{X}\right)^{\mathsf{T}} - \mathbf{H}^{\mathsf{T}}\left(\mathbf{M} * \mathbf{W}\mathbf{H}^{\mathsf{T}}\right)^{\mathsf{T}}\right) \tag{6}$$

The first term of each gradient costs the same as applying the data matrix individually to each of the factors costing $4mnk$ flops. The second terms costs $2\mathsf{nnz}(\mathbf{M})\,k + 4mnk$ flops. Thus, the cost of each inner iteration is approximately $8mnk + 2\mathsf{nnz}(\mathbf{M})\,k + T_{\text{upd}}$ operations.

Assuming the number of CV folds is n_{folds}, we shall perform $n_{\text{ranks}} n_{\text{seeds}} n_{\text{folds}}$ different masked NMF approximations. Unlike the non-masked version, computing the "held-out" residual is no longer an output of the approximation and will require $O\left(mnk/n_{\text{folds}}\right)$ flops. The best rank is selected by finding the lowest aggregated held-out residual for every k in $O\left(n_{\text{ranks}}\right)$ operations. Including $O\left(\left(1 - \frac{1}{n_{\text{folds}}}\right)mnk\right)$ cost for computing the objective, we get

$$\begin{aligned} T_{\text{CV}} &= n_{\text{ranks}} n_{\text{seeds}} n_{\text{folds}} T_{\text{MNMF}} \\ &\quad + O\left(n_{\text{ranks}} n_{\text{seeds}} mnk\right) + O(n_{\text{ranks}}). \end{aligned}$$

The different costs for the various methods discussed in this section are summarised in Table I.

IV. Numerical Experiments

The numerical experiments were performed on a server with two Intel® Xeon® E5-2680 v3 CPUs and 377 GB of DDR4-2,133 MHz DRAM using Matlab version R2021a[1].

A. Datasets

We conducted experiments on the following datasets.

1) **Synmats**: These matrices are formed as $\mathbf{X} = \mathbf{W}\mathbf{H}^{\mathsf{T}} + \sigma\mathbf{N}$. $\mathbf{W}$ and $\mathbf{H}$ are chosen with i.i.d. $\mathcal{U}[0,1]$ entries. The columns of $\mathbf{W}$ and rows of $\mathbf{H}$ sum to one. Here $\mathbf{N}$ is a noise matrix which could arise from $\mathcal{N}(0,1)$, $\mathcal{U}[0,1]$, or $\mathcal{U}[-1,1]$. The parameter σ is used to scale as some percentage of the maximum entry in $\mathbf{W}\mathbf{H}^{\mathsf{T}}$. Finally, to ensure nonnegativity of $\mathbf{X}$ we clip any values which become negative to 0. The dimensions for $\mathbf{X}$ were 100×600 with the number of true components $k = 10$. We vary σ from 1% to 37% in increments of 4%.

[1]https://gitlab.com/seswar3/cv-nmf

2) **Fashion MNIST** [23]: This dataset comprises 70,000 28×28 grayscale images of clothing articles divided into 10 categories (e.g., trouser, coat, shirt, etc.). We randomly sample 1,000 images, 100 from each class, and flatten them into vectors of length 784. These vectors form the columns of $\mathbf{X}$. Some examples from the dataset are shown in Fig. 1.

B. Methods Compared

We compare the following methods in our experiments.

1) SVHT - This is the optimally tuned singular value hard thresholding (SVHT) algorithm from Gavish and Donoho [14] (see Section III-C).
2) We evaluate the Relative Error, BIC, and the matrix version of CORCONDIA metric-based scree tests (see Section III-A). We compute NMF approximations for each rank from 5 different initializations using the ANLS method. The NLS solver used is the Block Principal Pivoting (BPP) method [24]. The NMF algorithms were limited to 100 iterations. The median value across the different random initializations are reported. The rank reported is the $\min$ (or $\max$ for CORCONDIA) value of the metric among the ranks tested.
3) Cross-validation - We run 5-fold CV holding out roughly 20% entries of $\mathbf{X}$ in each fold. We again utilize a ANLS method with BPP used as the "censored" NLS solver. Outer iterations were limited to a maximum of 100 iterations with 5 random initializations. The median of the different CV errors, across initializations and folds, are reported.
   a) Heldin: The elementwise held-in error is computed as $\left\|\mathbf{M} * \left(\mathbf{X} - \mathbf{W}\mathbf{H}^{\mathsf{T}}\right)\right\|_F^2 / \mathsf{nnz}(\mathbf{M})$. These entries are seen by the masked NMF while fitting for $\mathbf{W}$ and $\mathbf{H}$.
   b) Heldout: The elementwise held-out error is computed as $\left\|(\mathbf{1} - \mathbf{M}) * \left(\mathbf{X} - \mathbf{W}\mathbf{H}^{\mathsf{T}}\right)\right\|_F^2 / \left(mn - \mathsf{nnz}(\mathbf{M})\right)$.
   c) Generalization: Average of the above quantities.

   The rank reported is the $\min$ elementwise error among the ranks tested.

For the metric and CV ranks, we explicitly force SELECTBEST to be simple, like the $\min$, to improve reproducibility in such methods. We would like to insure against any form of human interpretation of the plots in our study. NMFk is not included in these experiments since it involves the tuning of other hyperparameters. We observed that the rank selected by NMFk to vary substantially based on the values of the silhouette threshold and perturbation noise variance parameters.

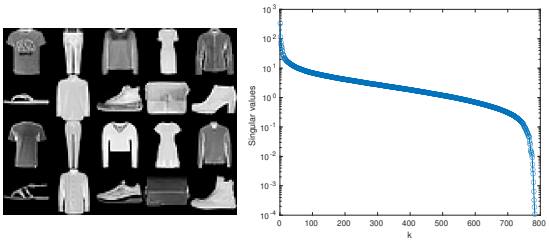


Fig. 1: Sample images and spectra of Fashion MNIST.

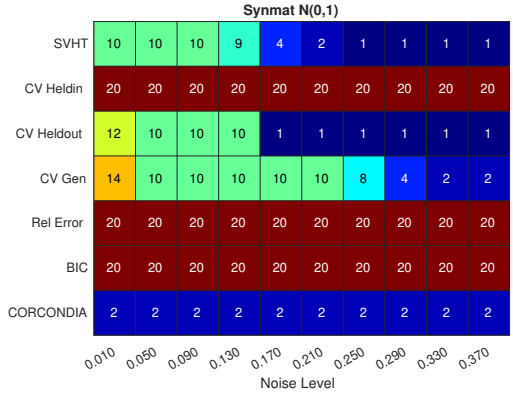


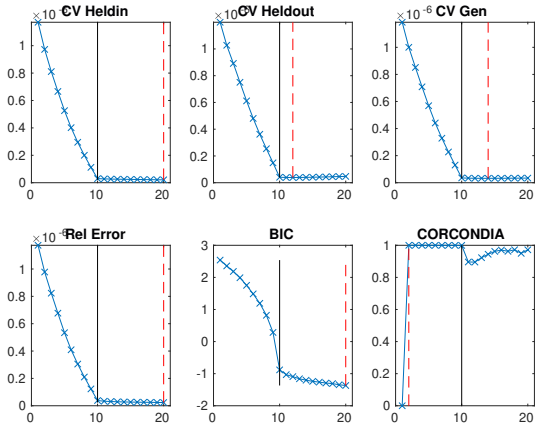
Fig. 2: Ranks selected for synthetic matrices corrupted with varying amounts of Gaussian noise. The true rank is 10.

C. Synthetic Matrices

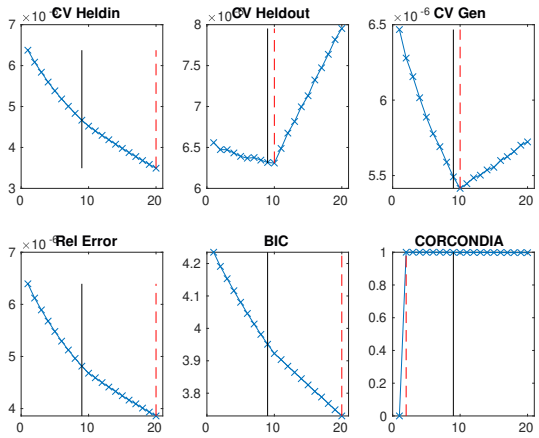
The performance of the different methods for the synthetic matrices perturbed by Gaussian noise is shown in Fig. 2. Recall that the true number of components is 10. We tested all values of $k \in [1, 20]$. Only SVHT and CV are able to recover this value for some noise levels whereas all the other methods fail to detect the true k even for small noise values. SVHT behaves in an expected manner by correctly discovering the number of true components when the noise levels are low and then progressively becoming worse as the noise level rises. After around 13% noise it fails to recover k. CV initially overestimates the number of components but is able to ascertain the true k till around 25% noise.

Let us investigate the scree plots for the low, medium, and high noise cases (see Fig. 3). Figure 3a shows the lowest noise level of 1%. While the "elbow" in the scree plots are evident on inspection, most of the metrics show monotonic decrease with increasing rank and thus fail to detect the true k. For the medium noise regime, Fig. 3b, we observe the classic CV curves with an initial decrease followed by a sharp increase for the heldout and generalization plots. The BIC and relative error plots all show a monotonic decrease making it hard even to detect k by more complicated means than simply taking the minimum. Finally, in the high noise regime (Fig. 3c) all the methods fail. Most curves are monotonic with no discernible elbow with the exception of CV generalization. However, detecting that $k = 10$ in that curve is not immediately evident. One final observation is that the CORCONDIA metric behaves similarly irrespective of the noise level.

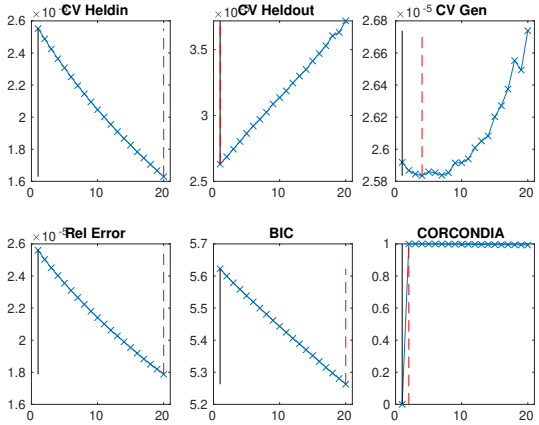
We next see the cases with uniform noise. As before, only CV and SVHT recover the the true number of components



(a) 1% noise



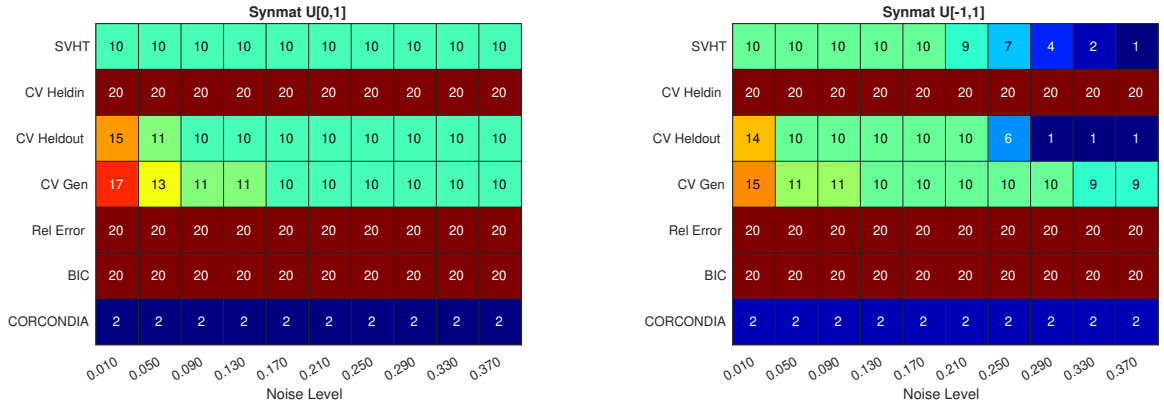
(b) 13% noise



(c) 29% noise

Fig. 3: Scree plots for different levels of Gaussian noise. The red line shows the rank selected by the method while the black line shows that of SVHT.

for different noise levels. The $\mathcal{U}[0, 1]$ case seemed easier to handle in our experiments, with SVHT recovering the true number of components for all noise levels and CV also performing well. CV again initially overestimates the number of components but is more tolerant to the higher noise regions for $\mathcal{U}[-1, 1]$ noise than SVHT. Another point of note is that in our synthetic experiments SVHT either detects the true rank



(a) $\mathcal{U}[0,1]$ noise (b) $\mathcal{U}[-1,1]$ noise

Fig. 4: Ranks selected for synthetic matrices perturbed with uniform noise. The true number of components is 10

TABLE II: Run times on Gaussian synthetic matrix.

Algorithm	SVHT	Scree	CV (BCD)
Time Taken (s)	0.0044 s	11.3065 s	823.2508 s

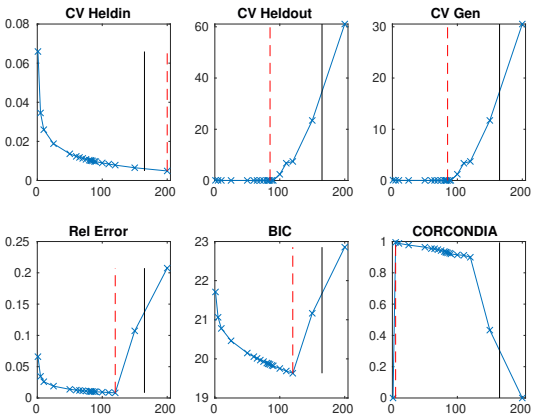


Fig. 5: Fashion MNIST scree plots. The outputs of SVHT is 165, CV Heldin is 200, CV Heldout and Generalization is 85, relative error is 120, BIC is 120, and CORCONDIA is 5.

or underapproximates that value as noise increases. CV can also overestimate the rank in the low noise settings.

Table II shows the run time needed to perform the one run of rank selection for a synthetic cases (Gaussian noise of 13%). This is to quantify the computational complexity described in Table I. The advantages of not sweeping through the different ranks is evident with screeing taking $\sim 2{,}500\times$ than SVT and CV taking a further $72\times$ longer.

D. Fashion MNIST Case Study

The spectrum of the sampled Fashion MNIST dataset is shown in Fig. 1. The numerical rank for this matrix is 783, which is nearly full rank. Approximately 90%, 95%, and 99% of the Frobenius norm is captured at rank 15, 54, and 226, respectively. The number of different classes of clothing articles is only 10. It is reasonable to assume that the number of different "parts" used to construct these would lie somewhere closer to 10 than the numerical rank.

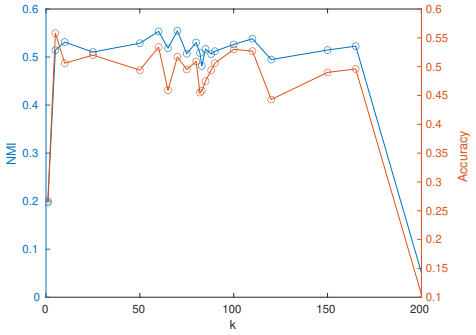


Fig. 6: K-Means clustering results on the embeddings generated for different ranks. Both measures remain rather agnostic to the k value chosen with the quality remaining flat.

SVHT predicts the rank for this dataset to be 165. In order to select the different ranks to test in the scree and CV methods, we adopt a binary search like procedure between rank 1 and 200. Apart from the midpoints obtained via the binary search, we also evaluate a few neighbouring ranks[2] as shown in Fig. 5. The CV plots display expected behaviour with the heldin error monotically decreasing with the minimum at 200, and the heldout and generalization curves showing a inflection point at around 85. CORCONDIA is again similar to the synthetic examples, with rank 5 having the highest value. The scree plots for the relative error and BIC show different behaviour than earlier. Instead of monotonically decreasing, we see an inflection point at rank 120. This is surprising, since relative error is expected to not increase as we increase rank. On closer inspection, we observe that at large ranks the NLS solver breaks down for multiple random initializations and return all zeros for the factors. Only a few of the restarts were able to recover good solutions. On the other hand, CV was able to show the inflection points without any numerical errors. Next we test the embeddings, rows of $\mathbf{H}$, returned by NMF for the different ranks considered. For each rank, we pick the best approximation from the 5 different seeds in terms of relative

[2]The ranks tested were 1, 5, 10, 25, 50, 60, 65, 70, 75, 80, 82, 83, 85, 88, 90, 100, 110, 120, 150, and 200.

errors. Then we run K-means on the embeddings to obtain 10 clusters. We measure the normalized mutual information between the cluster labels and true categories in Fig. 6. We also report the the supervised accuracy from these labels by first matching each cluster membership vector to the true categorical labels followed by counting the correctly identified samples. We see both NMI and accuracy jump to reasonable values at $k = 5$ and remain relatively flat till $k = 200$. The highest NMI is reached at $k = 70$ and highest accuracy at $k = 5$. This observation along with the relative plateauing of these clustering metrics indicate $k \leq 100$ is a reasonable size for this dataset. This has an added benefit of remaining in the range where the NLS solvers remain stable.

V. Discussion

From our numerical experiments it is clear that rank selection is a difficult task with no method successful in all scenarios. The SVT methods show the most promise with their relatively cheap computational needs and nice theoretical properties for unconstrained matrix denoising. In our synthetic experiments, SVHT never overestimated k and had very good results on all three types of noise. However, in the image dataset it seems to overestimate the number of components and returns the largest number among the methods tested.

CV is the other promising method in our experiments. It seems to be more tolerant to a wider range of noise in the synthetic experiments and returns reasonable k estimates even in the Fashion MNIST case study. Surprisingly, it does overestimate k at low noise levels which warrants further study. Unfortunately, CV is by far the most expensive of all the methods in our experiments, taking an order of magnitude more time than any of the metric based scree tests.

The metric based methods, simple scree plots, BIC, and CORCONDIA, all performed poorly. The relative error and BIC plots were mostly monotonic and returned the largest k tested. For low noise regimes, a more sophisticated method than simply choosing the $\min$ might perform well (see Fig. 3a). However this is still a subjective selector and for the sake of reproducibility a simple selection criteria should be the default option. CORCONDIA seemed to behave in the same manner for all inputs and performed poorly.

VI. Conclusions and Future Work

We surveyed and described some popular methods for selecting the rank for NMF. We compared these tools against synthetic and real-world datasets via numerical experimentation. While no method clearly outperforms the rest, we believe that spectral methods, like the SVHT, should be the first option to try when encountering a new dataset. However, in scenarios with lower signal-to-noise ratios, CV methods can provide more accurate estimates albeit at a higher computational cost.

Further theoretical analyses are needed to understand the roles of masks in the constrained least squares settings. This would help shed light on the failure of CV in the low-noise settings. Future avenues of research could also be in reducing the computational overhead of CV. Some immediate approaches could be via the use of more structured masks for partitioning the input matrix. Parallel algorithms for accelerating censored least squares is another viable option.

References

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

[2] T. Adali, F. Kantar, M. A. B. S. Akhonda, S. Strother, V. D. Calhoun, and E. Acar, "Reproducibility in matrix and tensor decompositions: Focus on model match, interpretability, and uniqueness," *IEEE Sig. Proc. Mag.*, vol. 39, no. 4, pp. 8–24, 2022.

[3] A. Moitra, "An almost optimal algorithm for computing nonnegative rank," *SIAM Journal on Computing*, vol. 45, no. 1, pp. 156–173, 2016.

[4] R. Kannan, G. Ballard, and H. Park, "MPI-FAUN: An MPI-based framework for alternating-updating nonnegative matrix factorization," *IEEE TKDE*, vol. 30, no. 3, pp. 544–558, Mar. 2018.

[5] S. Eswar, K. Hayashi, G. Ballard, R. Kannan, M. A. Matheson, and H. Park, "PLANC: Parallel low-rank approximation with nonnegativity constraints," *ACM TOMS*, vol. 47, no. 3, jun 2021.

[6] J. Kim, Y. He, and H. Park, "Algorithms for nonnegative matrix and tensor factorizations: A unified view based on block coordinate descent framework," *J. Global Optimization*, vol. 58, no. 2, pp. 285–319, 2014.

[7] K. Huang, N. D. Sidiropoulos, and A. P. Liavas, "A flexible and efficient algorithmic framework for constrained matrix and tensor factorization," *IEEE TSP*, vol. 64, no. 19, pp. 5052–5065, 2016.

[8] R. B. Cattell, "The scree test for the number of factors," *Multivariate behavioral research*, vol. 1, no. 2, pp. 245–276, 1966.

[9] G. Schwarz, "Estimating the dimension of a model," *The annals of statistics*, pp. 461–464, 1978.

[10] R. Bro and H. A. Kiers, "A new efficient method for determining the number of components in parafac models," *Journal of Chemometrics: A Journal of the Chemometrics Society*, vol. 17, no. 5, pp. 274–286, 2003.

[11] L. Alexandrov, S. Nik-Zainal, D. Wedge, P. Campbell, and M. Stratton, "Deciphering signatures of mutational processes operative in human cancer," *Cell reports*, vol. 3, pp. 246–259, 01 2013.

[12] B. Alexandrov and V. Vesselinov, "Blind source separation for groundwater pressure analysis based on nonnegative matrix factorization," *Water Resources Research*, vol. 50, p. 7332–7347, 09 2014.

[13] G. Chennupati, R. Vangara, E. Skau, H. Djidjev, and B. Alexandrov, "Distributed non-negative matrix factorization with determination of the number of latent features," *The Journal of Supercomputing*, vol. 76, no. 9, pp. 7458–7488, 2020.

[14] M. Gavish and D. L. Donoho, "The optimal hard threshold for singular values is $4/\sqrt{3}$," *IEEE Transactions on Information Theory*, vol. 60, no. 8, pp. 5040–5053, 2014.

[15] J. L. Horn, "A rationale and test for the number of factors in factor analysis," *Psychometrika*, vol. 30, pp. 179–185, 1965.

[16] E. Dobriban and A. B. Owen, "Deterministic parallel analysis: an improved method for selecting factors and principal components," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 81, no. 1, pp. 163–183, 2019.

[17] D. Donoho, M. Gavish, and E. Romanov, "Screenot: Exact mse-optimal singular value thresholding in correlated noise," *The Annals of Statistics*, vol. 51, no. 1, pp. 122–148, 2023.

[18] S. Wold, "Cross-validatory estimation of the number of components in factor and principal components models," *Technometrics*, vol. 20, no. 4, pp. 397–405, 1978.

[19] K. R. Gabriel, "Le biplot-outil d'exploration de données multidimensionnelles," *Journal de la société française de statistique*, vol. 143, no. 3-4, pp. 5–55, 2002.

[20] A. Williams, "Solving least-squares regression with missing data," http://alexhwilliams.info/itsneuronalblog/2018/02/26/censored-lstsq/, 2018.

[21] A. B. Owen and P. O. Perry, "Bi-cross-validation of the svd and the nonnegative matrix factorization," *The annals of applied statistics*, vol. 3, no. 2, pp. 564–594, 2009.

[22] B. Kanagal and V. Sindhwani, "Rank selection in low-rank matrix approximations: A study of cross-validation for NMFs," in *Proc Conf Adv Neural Inf Process*, vol. 1, 2010, pp. 10–15.

[23] H. Xiao, K. Rasul, and R. Vollgraf. (2017) Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.

[24] J. Kim and H. Park, "Fast nonnegative matrix factorization: An active-set-like method and comparisons," *SISC*, vol. 33, no. 6, pp. 3261–3281, 2011.