Nonnegative Matrix Factorization through Cone Collapse

Manh Nguyen
Department of Statistics
Wisconsin Institute of Discovery
University of Wisconsin-Madison
mdnguyen4@wisc.edu

Daniel Pimentel-Alarcón
Department of Biostatistics and Medical Informatics
Wisconsin Institute of Discovery
University of Wisconsin-Madison
pimentelalar@wisc.edu

Abstract

Nonnegative matrix factorization (NMF) is a widely used tool for learning parts-based, low-dimensional representations of nonnegative data, with applications in vision, text, and bioinformatics [1, 2]. In clustering applications, orthogonal NMF (ONMF) variants further impose (approximate) orthogonality on the representation matrix so that its rows behave like soft cluster indicators [3, 4]. Existing algorithms, however, are typically derived from optimization viewpoints and do not explicitly exploit the conic geometry induced by NMF: data points lie in a convex cone whose extreme rays encode fundamental directions or "topics". In this work we revisit NMF from this geometric perspective and propose *Cone Collapse*, an algorithm that starts from the full nonnegative orthant and iteratively shrinks it toward the minimal cone generated by the data. We prove that, under mild assumptions on the data, Cone Collapse terminates in finitely many steps and recovers the minimal generating cone of $\mathbf{X}^\top$. Building on this basis, we then derive a cone-aware orthogonal NMF model (CC–NMF) by applying uni-orthogonal NMF to the recovered extreme rays [4, 5]. Across 16 benchmark gene-expression, text, and image datasets, CC–NMF consistently matches or outperforms strong NMF baselines—including multiplicative updates, ANLS, projective NMF, ONMF, and sparse NMF—in terms of clustering purity. These results demonstrate that explicitly recovering the data cone can yield both theoretically grounded and empirically strong NMF-based clustering methods. The implementation for our method is provided in github.com/manhbeo/cone-collapse.

1 Introduction

Low-rank matrix factorization methods are central tools for discovering low-dimensional structure in high-dimensional data. Classical techniques such as principal component analysis (PCA) and singular value decomposition (SVD) provide optimal rank-r approximations in the least-squares sense, but their components typically contain both positive and negative entries, which complicates interpretation when the data are inherently nonnegative (e.g., pixel intensities, word counts, gene-expression levels). Nonnegative matrix factorization (NMF) [1, 2] addresses this limitation by constraining both factors to be nonnegative. The resulting additive, parts-based decompositions have been successfully used for document clustering [6], topic modeling, and molecular pattern discovery, and have been linked to probabilistic latent semantic indexing and related latent-variable models [3].

Beyond representation learning, NMF has a long-standing connection to clustering. By constraining one factor to be close to an indicator matrix, NMF can be shown to approximate k-means and spectral clustering objectives [3]. Orthogonal NMF (ONMF) formulations make this connection

Preprint. Under review.

explicit by enforcing an orthogonality constraint on either the basis or the coefficient matrix, e.g., $\mathbf{H}\mathbf{H}^\top = \mathbf{I}$ in a factorization $\mathbf{X} \approx \mathbf{W}\mathbf{H}$. In such models, each row of $\mathbf{H}$ behaves like a soft indicator vector for a cluster, and assigning each data point to the row with maximal activation yields a clustering [4, 7, 5, 8]. These ONMF variants have been particularly successful on document and image clustering benchmarks.

Most NMF and ONMF algorithms are derived from an optimization viewpoint, using multiplicative updates [2], projected gradient methods, or alternating nonnegative least squares (ANLS) with advanced solvers such as block principal pivoting [9]. While effective in practice, these approaches rarely exploit the explicit *conic* geometry underlying NMF: the columns of a data matrix $\mathbf{X}$ lie in the convex cone $\mathrm{cone}(\mathbf{W})$ generated by the basis vectors, and the extreme rays of this cone play a role analogous to "topics" or "anchors". In parallel, a line of work on separable NMF and topic modeling has developed algorithms that directly recover extreme rays (or "anchor words") under structural assumptions, with provable guarantees [10, 3]. However, these methods typically operate in simplex or probability-simplex settings and are not designed to integrate with ONMF-style clustering objectives.

Our approach. In this paper we propose *Cone Collapse*, a new algorithm that explicitly recovers the minimal generating cone of the data and then uses it as the basis for an ONMF-style clustering model. Given a nonnegative data matrix $\mathbf{X} \in \mathbb{R}_+^{m\times n}$, we view its transpose $\mathbf{X}^\top$ as a set of points in $\mathbb{R}_+^n$ and seek a matrix $\mathbf{U}^\star = [\mathbf{u}_1, \ldots, \mathbf{u}_c]$ whose columns correspond to extreme rays of the data cone $\mathrm{cone}(\mathbf{X}^\top)$. Cone Collapse starts from the full nonnegative orthant (via the identity matrix) and iteratively *shrinks* this cone by tilting free rays toward the mean direction of the data while ensuring that all points remain inside the cone. When a data point falls outside, it is added as a new ray; when a ray becomes representable as a nonnegative combination of others, it is pruned. These steps are implemented via NNLS subproblems solved efficiently by block principal pivoting [9]. Intuitively, the algorithm contracts an initial, overly large cone until only the essential extreme rays remain.

Once $\mathbf{U}^\star$ has been recovered, we fit another orthogonal cone $\mathrm{cone}(\mathbf{A})$ of r rays to $\mathrm{cone}(\mathbf{U}^\star)$ by solving a uni-orthogonal NMF problem $\mathbf{U}^\star \approx \mathbf{A}\mathbf{S}$ with $\mathbf{A}^\top\mathbf{A} = \mathbf{I}$ using multiplicative ONMF updates [4, 5]. This yields an orthogonal factorization $\mathbf{X} \approx \mathbf{W}\mathbf{H}$, where $\mathbf{H} = \mathbf{A}^\top$ has orthonormal rows and can be used directly for clustering. We refer to the resulting model as *CC–NMF* (Cone Collapse NMF). Figure 1 provides a schematic illustration of this two-stage pipeline.

Contributions. Our main contributions are:

- We introduce **Cone Collapse**, a new algorithm for recovering a minimal generating cone of a nonnegative data matrix. The method combines mean-tilting, outside-point detection, and redundancy pruning, and is built on efficient NNLS routines [9].
- We provide a **theoretical justification** for Cone Collapse: under mild assumptions (clean data and nondegenerate extreme rays), we prove that the algorithm terminates after finitely many iterations and returns a basis whose cone coincides with the data cone, i.e., $\mathrm{cone}(\mathbf{U}^{(T)}) = \mathrm{cone}(\mathbf{X}^\top)$.
- We show how to integrate Cone Collapse with **orthogonal NMF**, yielding CC–NMF, a cone-aware ONMF model whose latent factors are explicitly tied to extreme rays of the data cone, in contrast to existing ONMF formulations [4, 7, 5, 8].
- We conduct a **comprehensive empirical evaluation** on 16 benchmark datasets spanning gene expression, text, and images, and demonstrate that CC–NMF consistently matches or outperforms strong NMF baselines—including MU, ANLS, PNMF, ONMF, and sparse NMF—in clustering purity.

Organization. Section 2 reviews related work on NMF and ONMF. Section 3 introduces the Cone Collapse algorithm and its geometric interpretation. Section 4 establishes the finite-termination and exact-recovery guarantees. Section 5 describes our experimental setup and clustering results, and Section 6 concludes with a discussion of limitations and future directions.

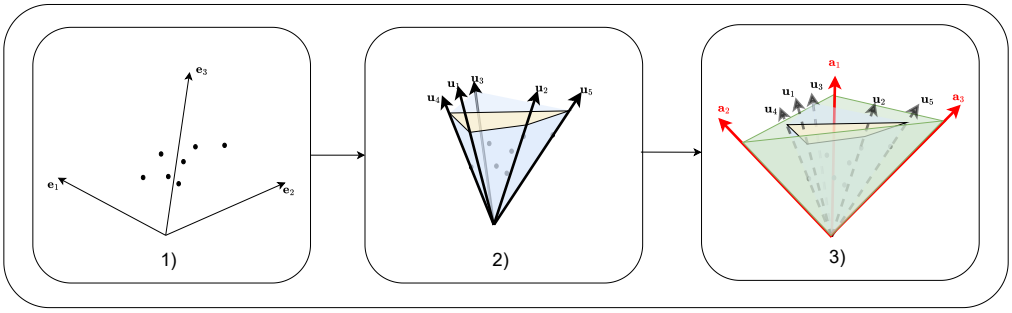

Figure 1: An illustration of our method: (1) initialize with basis vectors e_1, e_2, e_3; (2) recover the data cone $\mathbf{U}^\star$ that contains all columns of $\mathbf{X}^\top$ via Cone Collapse; and (3) fit an orthogonal cone $\mathbf{A}$ to $\mathbf{U}^\star$.

2 Related work

Matrix factorization and parts-based representations. Matrix factorization methods such as singular value decomposition (SVD) and principal component analysis (PCA) have long been used to obtain low-dimensional representations of high-dimensional data. However, these factorizations typically produce components with both positive and negative entries, which hinders interpretability when the data themselves are nonnegative (e.g., images, word counts, or term–document matrices). Nonnegative matrix factorization (NMF) addresses this issue by constraining both factors to be nonnegative, leading to parts-based or additive representations that have proven useful in computer vision, text analysis, and bioinformatics [1, 2]. Subsequent work has revealed close connections between NMF and clustering objectives, including equivalences to k-means and spectral clustering under appropriate constraints [3].

Nonnegative matrix factorization. Nonnegative matrix factorization (NMF) seeks to approximate a nonnegative data matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}_+^{m \times n}$ by a low–rank product

$$\mathbf{X} \;\approx\; \mathbf{WH}, \qquad \mathbf{W} \in \mathbb{R}_+^{m \times r},\ \mathbf{H} \in \mathbb{R}_+^{r \times n}.$$

We assume $\mathbf{X}$ is clean, i.e. there is no full zero column or row in $\mathbf{X}$. Here, n is the number of examples and m is the number of features. Each column $\mathbf{x}_i$ is then represented as a nonnegative combination of the basis vectors $\mathbf{w}_1, \ldots, \mathbf{w}_r$ (the columns of $\mathbf{W}$), i.e.

$$\mathbf{x}_i \;\approx\; \mathbf{W}\mathbf{h}_i \;=\; \sum_{k=1}^{r} h_{ki}\, \mathbf{w}_k, \qquad h_{ki} \geq 0.$$

Geometrically, this means that all data points $\mathbf{x}_i$ lie (approximately) inside the convex cone generated by the columns of $\mathbf{W}$,

$$\mathrm{cone}(\mathbf{W}) \;:=\; \Big\{ \sum_{k=1}^{r} \alpha_k \mathbf{w}_k : \alpha_k \geq 0 \Big\}.$$

Thus, NMF can be interpreted as the problem of finding a low–dimensional cone that captures the data cloud $\mathbf{X}$ [2].

Orthogonal nonnegative matrix factorization for clustering. Orthogonal nonnegative matrix factorization (ONMF) augments NMF with an orthogonality constraint on one of the factors, typically

$$\mathbf{X} \;\approx\; \mathbf{WH}, \qquad \mathbf{W} \in \mathbb{R}_+^{m \times r},\ \mathbf{H} \in \mathbb{R}_+^{r \times n}, \quad \mathbf{H}\mathbf{H}^\top = \mathbf{I}_r,$$

or analogously with orthogonality imposed on the columns of $\mathbf{W}$. The nonnegativity of $\mathbf{H}$ encourages each data point $\mathbf{x}_i$ to be represented by a small number of latent components, while the orthogonality constraint forces the rows of $\mathbf{H}$ to behave like (soft) cluster-indicator vectors. A common

clustering interpretation is to assign each data point $\mathbf{x}_i$ to the cluster

$$\arg\max_{k\in\{1,\dots,r\}} h_{ki},$$

so that ONMF plays the role of a relaxed combinatorial clustering formulation. Under suitable conditions, these orthogonality constraints make NMF and its symmetric variants equivalent to k-means or spectral clustering objectives [3, 4], thereby providing a principled link between matrix factorization and graph-based clustering. While our approach is broadly applicable to NMF problems, in this work we focus on combining Cone Collapse with orthogonal NMF and evaluating it on clustering tasks.

3 Method

3.1 Cone Collapse Algorithm

Our goal in the first step is to recover a convex cone $\mathbf{U}^\star = [\mathbf{u}_1, \dots, \mathbf{u}_c] \in \mathbb{R}^{m\times c}$, whose generating rays capture the geometry of the data matrix $\mathbf{X}^\top$. Geometrically, the columns of $\mathbf{U}^\star$ correspond to (a subset or superset of) the *extreme rays* of the data cone

$$\mathrm{cone}(\mathbf{X}^\top) \;:=\; \{\mathbf{X}^\top\alpha : \alpha \geq 0\}.$$

A ray $\mathbb{R}_+\mathbf{u} \subset \mathrm{cone}(\mathbf{X}^\top)$ is called *extreme* if it cannot be written as a nontrivial conic combination of other rays in the cone: whenever $\mathbf{u} = \mathbf{v} + \mathbf{w}$ with $\mathbf{v}, \mathbf{w} \in \mathrm{cone}(\mathbf{X}^\top)$, we must have $\mathbf{v} = a\mathbf{u}$ and $\mathbf{w} = b\mathbf{u}$ for some $a, b \geq 0$. In other words, extreme rays play the role of "corners" of the cone. An illustration for $\mathbf{U}^\star$ is provided in Figure 2. If $c = 1$, the cone reduces to a single ray and the problem is trivial: any nonzero rows of $\mathbf{X}$, after normalization, provides the unique direction. Therefore, we focus on the nontrivial regime $c \geq 2$.

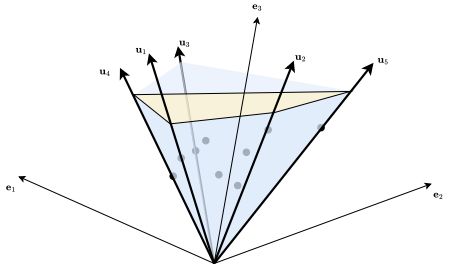

Figure 2: A 3D illustration for $\mathbf{U}^\star$ with $c = 6$. Each column of $\mathbf{U}^\star$ is represented as a ray, with the cone $\mathrm{cone}(\mathbf{U}^\star)$ contains all datapoints of $\mathbf{X}^\top$ (which are rows of $\mathbf{X}$, represented as points).

Intuition. The guiding idea behind our method is simple: we begin with a cone so large that it already contains all data points, and then we continuously *shrink* this cone toward a compact shape that reveals the true extreme rays of the data. Specifically, we initialize with $U^{(0)} = I_n$, which guarantees that every data point x_i lies inside $\mathrm{cone}(U^{(0)})$. From this starting point, we iteratively "tilt" each ray $\mathbf{u}_k^{(t)}$ toward the mean direction μ of the dataset. During this contraction process some data points may fall outside the current cone, which is then added to the cone. Finally, we prune any rays that have become redundant—those that can be expressed as a nonnegative combination of others—ensuring that the cone remains minimal.

For nonzero vectors $\mathbf{a}$ and $\mathbf{b}$, denote $\widehat{\mathbf{a}} := \mathbf{a}/\|\mathbf{a}\|$ and $\cos(\mathbf{a}, \mathbf{b}) := \langle\widehat{\mathbf{a}}, \widehat{\mathbf{b}}\rangle$; the Cone Collapse algorithm is describe in Algorithm 1.

Discussion:

- The *Mean tilt* step leaves the columns of $\widetilde{\mathbf{U}}^{(t)}$ that are co-linear with a point in $\mathbf{X}^\top$ unchanged and tilts the columns that are not co-linear with any point in $\mathbf{X}$ by

  $$\mathbf{u}_k^{(t)} \;\mapsto\; \widetilde{\mathbf{u}}_k^{(t)} \propto (1-\eta)\,\mathbf{u}_k^{(t)} + \eta\,\widehat{\boldsymbol{\mu}}.$$

  All columns always remain in $\mathbb{S}_+^{n-1} = \{\mathbf{v} \in \mathbb{R}_+^n : \|\mathbf{v}\|_2 = 1\}$.
- After each iteration t of the algorithm, we have $\mathbf{X}^\top \subseteq \mathrm{cone}\left(\mathbf{U}^{(t+1)}\right)$. Indeed, at the *Add outside point* step every $\mathbf{x}_i$ with residual $\|\mathbf{r}_i\|_2 > \epsilon\|\mathbf{x}_i\|_2$ is appended as $\widehat{\mathbf{x}}_i$, so $\mathbf{X}^\top \subseteq \mathrm{cone}(\widetilde{\mathbf{U}}^{(t)})$ holds immediately after that step. The *Remove redundant rays* step only removes columns $\widetilde{\mathbf{u}}_k$ that satisfy $\widetilde{\mathbf{u}}_k \in \mathrm{cone}(\widetilde{\mathbf{U}}_{-k}^{(t)})$ (up to ϵ), so deleting them does not change the cone.

Algorithm 1 Cone–Collapse Algorithm

Require: Data $\mathbf{X}^\top = [\mathbf{x}_1, \ldots, \mathbf{x}_m] \in \mathbb{R}_+^{n \times m}$, learning rate $\eta \in (0, 1)$, tolerance ϵ.
Ensure: Final basis $\mathbf{U}^{(T)}$

init $\boldsymbol{\mu} \leftarrow \dfrac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i, \quad \mathbf{U}^{(0)} \leftarrow \mathbf{I}_m, \quad t \leftarrow 0$
repeat ▷ Iteration t
  for each column $\mathbf{u}_k^{(t)}$ of $\mathbf{U}^{(t)}$ **do** ▷ *Mean tilt*

$$\widetilde{\mathbf{u}}_k \leftarrow \begin{cases} \mathbf{u}_k^{(t)}, & \text{If } \exists\, i \in [n] : \cos(\mathbf{u}_k^{(t)}, \mathbf{x}_i) = 1, \\ \dfrac{(1-\eta)\,\mathbf{u}_k^{(t)} + \eta\,\widehat{\boldsymbol{\mu}}}{\|(1-\eta)\,\mathbf{u}_k^{(t)} + \eta\,\widehat{\boldsymbol{\mu}}\|_2}, & \text{otherwise.} \end{cases}$$

  end for
  $\widetilde{\mathbf{U}}^{(t)} \leftarrow [\widetilde{\mathbf{u}}_1, \ldots, \widetilde{\mathbf{u}}_{c_t}]$ ▷ c_t: number of columns of $\mathbf{U}^{(t)}$
  $\mathbf{H}^\star \leftarrow \arg\min_{\mathbf{H} \geq 0} \left\|\mathbf{X}^\top - \widetilde{\mathbf{U}}^{(t)}\mathbf{H}\right\|_F^2$
  $\mathbf{R} := [\mathbf{r}_1, \ldots, \mathbf{r}_n] \leftarrow \mathbf{X}^\top - \widetilde{\mathbf{U}}^{(t)}\mathbf{H}^\star$
  for $i = 1, \ldots, n$ **do**
    if $\|\mathbf{r}_i\|_2 > \epsilon\, \|\mathbf{x}_i\|_2$ **then**
      $\widetilde{\mathbf{U}}^{(t)} \leftarrow \left[\, \widetilde{\mathbf{U}}^{(t)} \;\; \widehat{\mathbf{x}}_i \,\right]$ ▷ *Add outside point*
      for each column index k of $\widetilde{\mathbf{U}}^{(t)}$ **do**
        $\widetilde{\mathbf{U}}_{-k}^{(t)} \leftarrow (\widetilde{\mathbf{U}}^{(t)}$ with column k removed$)$
        $\mathbf{w}_k^\star \leftarrow \arg\min_{\mathbf{w} \geq 0} \left\|\widetilde{\mathbf{u}}_k - \widetilde{\mathbf{U}}_{-k}^{(t)}\mathbf{w}\right\|_2^2, \quad \rho_k \leftarrow \left\|\widetilde{\mathbf{u}}_k - \widetilde{\mathbf{U}}_{-k}^{(t)}\mathbf{w}_k^\star\right\|_2$
        if $\rho_k \leq \epsilon\, \|\widetilde{\mathbf{u}}_k\|_2$ **then**
          $\widetilde{\mathbf{U}}^{(t)} \leftarrow \widetilde{\mathbf{U}}_{-k}^{(t)}$ ▷ *Remove redundant rays*
        end if
      end for
    end if
  end for
  $\mathbf{U}^{(t+1)} \leftarrow \widetilde{\mathbf{U}}^{(t)}$
  $t \leftarrow t + 1$
until $\forall\, k\ \exists\, i \in [n]$ such that $\cos(\widetilde{\mathbf{u}}_k, \mathbf{x}_i) = 1$
return $\mathbf{U}^{(T)} \leftarrow \mathbf{U}^{(t)}$

- $\arg\min_{\mathbf{w} \geq 0} \left\|\widetilde{\mathbf{u}}_k - \widetilde{\mathbf{U}}_{-k}^{(t)}\mathbf{w}\right\|_2^2$ and $\mathbf{H}^\star \leftarrow \arg\min_{\mathbf{H} \geq 0} \left\|\mathbf{X}^\top - \widetilde{\mathbf{U}}^{(t)}\mathbf{H}\right\|_F^2$ are solved by Algorithm 2 and Algorithm 3 provided in Appendix D. One may consider adding all the outside points before removing redundant rays; however, in practice with large m and n, this can increase the solving time and lead to unnecessary NNLS calls.
- The learning rate η does not affect the results of the algorithm, but it affect the time required to converge. A smaller η typically requires more iterations but leads to a smaller number of points falling outside $\mathrm{cone}(\mathbf{U}^{(t)})$ and vise versa.

3.2 Orthogonal NMF using Cone Collapse algorithm

Recall that our goal is to obtain an orthogonal NMF of the form

$$\mathbf{X} \;\approx\; \mathbf{W}\mathbf{H}, \qquad \mathbf{W} \in \mathbb{R}_+^{m \times r},\ \mathbf{H} \in \mathbb{R}_+^{r \times n},\ \mathbf{H}\mathbf{H}^\top = \mathbf{I}_r,$$

so that the rows of $\mathbf{H}$ act as (approximately) orthogonal cluster indicators for the columns of $\mathbf{X}$, in the spirit of orthogonal NMF and its connection to k-means clustering [4].

After obtaining the extreme–ray basis $\mathbf{U}^\star \in \mathbb{R}_+^{n \times c}$ from Algorithm 1, we first solve a nonnegative least–squares problem

$$\mathbf{V}^\star \;=\; \arg\min_{\mathbf{V} \geq 0} \left\|\mathbf{X}^\top - \mathbf{U}^\star\mathbf{V}\right\|_F^2,$$

so that $\mathbf{X}^\top \approx \mathbf{U}^\star\mathbf{V}^\star$ and hence $\mathbf{X} \approx (\mathbf{V}^\star)^\top(\mathbf{U}^\star)^\top$.

We then compress the cone basis $\mathbf{U}^\star$ by solving a uni-orthogonal ONMF problem

$$\min_{\mathbf{A}\geq 0,\ \mathbf{S}\geq 0} \left\|\mathbf{U}^\star - \mathbf{A}\mathbf{S}\right\|_F^2 \quad \text{s.t.} \quad \mathbf{A}^\top\mathbf{A} = \mathbf{I}_r,$$

where $\mathbf{A} \in \mathbb{R}_+^{n\times r}$ has orthonormal columns and $\mathbf{S} \in \mathbb{R}_+^{r\times c}$ contains nonnegative loadings. Note that all columns of $\mathbf{U}^\star$ are in $\mathrm{cone}(\mathbf{A})$, so geometrically we are fitting another cone with a smaller number of extreme rays $r \leq c$ to the existing $\mathrm{cone}(\mathbf{U}^\star)$.

Following the uni-orthogonal NMF updates of Ding et al. [4], we adopt the multiplicative rules

$$\mathbf{S} \leftarrow \mathbf{S} \ \odot\ \frac{\mathbf{A}^\top\mathbf{U}^\star}{(\mathbf{A}^\top\mathbf{A})\,\mathbf{S}}, \tag{1}$$

$$\mathbf{A} \leftarrow \mathbf{A} \ \odot\ \frac{\mathbf{U}^\star\mathbf{S}^\top}{\mathbf{A}\mathbf{A}^\top\mathbf{U}^\star\mathbf{S}^\top}, \tag{2}$$

where all products/divisions are taken elementwise. In practice we periodically re-normalize the columns of $\mathbf{A}$ to keep $\mathbf{A}^\top\mathbf{A} \approx \mathbf{I}_r$, as is standard in ONMF algorithms based on multiplicative updates on the Stiefel manifold [5, 7].

Combining the two stages, we obtain the overall approximation

$$\mathbf{X} \ \approx\ (\mathbf{V}^\star)^\top\mathbf{S}^\top\mathbf{A}^\top,$$

so that a valid ONMF of $\mathbf{X}$ is given by

$$\mathbf{W} := (\mathbf{V}^\star)^\top\mathbf{S}^\top \in \mathbb{R}_+^{m\times r}, \qquad \mathbf{H} := \mathbf{A}^\top \in \mathbb{R}_+^{r\times n},$$

and the orthogonality constraint holds as

$$\mathbf{H}\mathbf{H}^\top = \mathbf{A}^\top\mathbf{A} = \mathbf{I}_r.$$

Thus, Cone Collapse provides a geometrically motivated extreme-ray basis $\mathbf{U}^\star$, while the ONMF step refines it into an orthogonal low-rank factorization of $\mathbf{X}$ via the multiplicative updates (1)–(2).

4 Theoretical justification

In this part, we introduce a theorem (and prove it) to demonstrate why the Cone Collapse algorithm will recover the minimal generating cone of $\mathbf{X}^\top$ and terminate in finitely many iterations. Formally, let $\mathbf{U}^{(t)} = \left[\mathbf{u}_1^{(t)}\ \ldots\ \mathbf{u}_{c_t}^{(t)}\right]$ and define the "frozen" and "free" sets:

$$\mathcal{F}^{(t)} := \left\{k : \exists\, i,\ \cos(\mathbf{u}_k^{(t)}, \mathbf{x}_i) = 1\right\}, \qquad \mathcal{B}^{(t)} := \{1, \ldots, c_t\} \setminus \mathcal{F}^{(t)},$$

that is, $\mathcal{F}^{(t)}$ is the set of ray indices in $\mathbf{U}^{(t)}$ that are co-linear with some data points in $\mathbf{X}^\top$, and $\mathcal{B}^{(t)}$ is the set of ray indices that are *not* co-linear with any data point in $\mathbf{X}^\top$.

We first introduce several Lemmas to help proving the theorem:

Lemma 4.1 (mean direction is not an extreme ray)**.** *Let $\mathbf{U}^\star = [\mathbf{u}_1, \ldots, \mathbf{u}_c] \in \mathbb{R}^{n\times c}$ be the convex cone built from the extreme rays of $\mathbf{X}^\top$, and $\boldsymbol{\mu} = \frac{1}{m}\sum_{i=1}^m \mathbf{x}_i$ be the mean of the data in $\mathbf{X}^\top$. If $c \geq 2$, $\forall k \in \{1,\ \ldots,\ c\}$ there does not exist τ such that $\boldsymbol{\mu} = \tau\ \mathbf{u}_k$.*

In other words, if there are more than 1 extreme ray, then no extreme ray is colinear with the mean.

Lemma 4.2 (contraction of free columns towards $\widehat{\boldsymbol{\mu}}$)**.** *For any $\mathbf{u}_k^{(t)} \in \mathcal{B}^{(t)}$ and $\eta \in (0,1)$, if $\mathbf{u}_k^{(t)} \neq \widehat{\boldsymbol{\mu}}$*

$$\cos(\widetilde{\mathbf{u}}_k^{(t)}, \widehat{\boldsymbol{\mu}}) > \cos(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}}),$$

that is, if $\mathbf{u}_k^{(t)} \neq \widehat{\boldsymbol{\mu}}$, the cosine of the angle between $\mathbf{u}_k^{(t)}$ and the mean $\widehat{\boldsymbol{\mu}}$ strictly increases after the Mean tilt step.

Lemma 4.3 (spherical cap stability under conic combinations)**.** *For $\alpha \in (0,1)$, define the cap*

$$\mathcal{C}_\alpha := \{\mathbf{v} \in \mathbb{S}_+^{n-1} : \cos(\mathbf{v}, \widehat{\boldsymbol{\mu}}) \geq \alpha\}.$$

For any $\mathbf{b}_\ell \in \mathcal{C}_\alpha$ and $\mathbf{w} \neq 0$, if $\mathbf{w} = \sum_\ell \lambda_\ell \mathbf{b}_\ell$ with $\lambda_\ell \geq 0$ then $\widehat{\mathbf{w}} \in \mathcal{C}_\alpha$.

The proofs for the Lemmas are provided in the Appendix. Taken together, these lemmas formalize the geometric intuition behind Cone Collapse. Lemma 4.1 guarantees that the mean direction $\widehat{\boldsymbol{\mu}}$ is never itself an extreme ray, so tilting rays toward $\widehat{\boldsymbol{\mu}}$ does not accidentally "snap" an extreme ray into the mean. Lemma 4.2 shows that every free ray $\mathbf{u}_k^{(t)} \in \mathcal{B}^{(t)}$ is progressively contracted toward $\widehat{\boldsymbol{\mu}}$, increasing its cosine with the mean at each iteration. Lemma 4.3 then implies that, after sufficiently many iterations, all free rays (and any conic combination thereof) lie inside a narrow spherical cap around $\widehat{\boldsymbol{\mu}}$, whereas each true extreme ray can be placed outside this cap by an appropriate choice of its aperture. As a consequence, the algorithm is forced to add any missing extreme rays as new columns whenever they are detected as "outside" points, and later removes all non-extreme columns since they remain representable as conic combinations of the extremes. We are thus led to the following finite-termination and exact-recovery guarantee for Cone Collapse.

Theorem 4.4. *Algorithm 1 halts after finitely many iteration T, with $\mathbf{U}^{(T)}$ consists of exactly c columns $\{\widehat{\mathbf{u}}_1, \ldots, \widehat{\mathbf{u}}_c\}$ (in some order), and*

$$\operatorname{cone}(\mathbf{U}^{(T)}) = \operatorname{cone}(\mathbf{X}) = \operatorname{cone}(\mathbf{U}^\star).$$

Proof. To prove the theorem, we show that: (i) All extreme rays are added to $\mathbf{U}^{(t)}$ and (ii) All other columns are removed from $\mathbf{U}^{(t)}$ after finitely many iterations.

All extreme rays appear after finitely many iterations: For any extreme rays u_j ($j \in \{1, \ldots, c\}$), suppose $\widehat{\mathbf{u}}_j$ is not added to $\mathbf{U}^{(t)}$ at the start of iteration t. By Lemma 4.1, we can choose $\alpha_j \in \big(\cos(\widehat{\mathbf{u}}_j, \widehat{\boldsymbol{\mu}}), 1\big)$. By Lemma 4.2, there exists T_j such that for all $t \geq T_j$, If $\mathbf{u}_k^{(t)} \in \mathcal{B}^{(t)}$ then $\cos(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}}) \geq \alpha_j$, which means that the $\operatorname{cone}\big(\{\mathbf{u}_k^{(t)} : k \in \mathcal{B}^{(t)}\}\big)$ is contained in $\mathcal{C}_{\alpha_j}$ by Lemma 4.3 (see Figure 3 for an illustration of $\mathcal{C}_{\alpha_i}$ contains $\operatorname{cone}\big(\{\mathbf{u}_k^{(t)} : k \in \mathcal{B}^{(t)}\}\big)$). Since $\alpha_j > \cos(\widehat{\mathbf{u}}_j, \widehat{\boldsymbol{\mu}})$, $\widehat{\mathbf{u}}_j \notin \mathcal{C}_{\alpha_j}$, and hence $\widehat{\mathbf{u}}_j \notin \operatorname{cone}\big(\{\mathbf{u}_k^{(t)} : k \in \mathcal{B}^{(t)}\}\big)$.

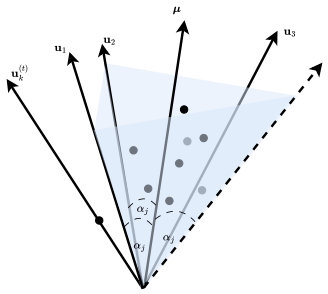

Figure 3: An illustration for $\mathcal{C}_{\alpha_i}$ contains $\operatorname{cone}\big(\{\mathbf{u}_k^{(t)} : k \in \mathcal{B}^{(t)}\}\big)$, with $\mathcal{C}_{\alpha_i}$ shaded. Here, we assume that $\mathcal{B}^{(t)} = \{1, 2, 3\}$.

For contradiction, we now assume that $\widehat{\mathbf{u}}_j \in \operatorname{cone}\big(\widetilde{\mathbf{U}}_k^{(t)}\big)$, which means there exists coefficients vector $\boldsymbol{\alpha} \geq 0$ such that

$$\widehat{\mathbf{u}}_j = \sum_{i \in \mathcal{B}^{(t)}} \alpha_i \mathbf{u}_i^{(t)} + \sum_{h \in \mathcal{F}^{(t)}} \alpha_h \mathbf{u}_h^{(t)} = \sum_k \alpha_k \mathbf{u}_k^{(t)},$$

However, by definition of extreme rays, we have $\widehat{\mathbf{u}}_j \notin \operatorname{cone}\big(\{\mathbf{u}_k^{(t)} : k \in \mathcal{F}^{(t)}\}\big)$. Hence, $\sum_{i \in \mathcal{B}^{(t)}} \alpha_i \mathbf{u}_i^{(t)} \neq 0$, which means $\widehat{\mathbf{u}}_j$ is then a conic combination of other rays (whether $\sum_{h \in \mathcal{F}^{(t)}} \alpha_h \mathbf{u}_h^{(t)} = 0$ or not), contradicting $\widehat{\mathbf{u}}_j$ is an extreme ray. Therefore, $\widehat{\mathbf{u}}_j$ is outside the $\operatorname{cone}\big(\widetilde{\mathbf{U}}_k^{(t)}\big)$.

The algorithm will then detect some $\mathbf{x}_i$ on ray $\widehat{\mathbf{u}}_j$ as a point outside ($\|\mathbf{r}_i\|_2 > \epsilon \|\mathbf{x}_i\|$) and appends $\widehat{\mathbf{x}}_i = \widehat{\mathbf{u}}_j$ to $\mathbf{U}^{(t)}$. Therefore, at step $T := \max_{1 \leq j \leq c} T_j$ the algorithm will append every extreme rays $\widehat{\mathbf{u}}_j$ that was previously missing. Since those extreme rays are not tilted toward the mean, there exists a finite iteration index T such that all extreme rays are added to $\mathbf{U}^{(T)}$.

Pruning to the minimal generating set and stopping criteria. Once all extreme rays are added to $\mathbf{U}^{(T)}$, $\operatorname{cone}\big(\mathbf{U}^{(T)}\big) = \operatorname{cone}(\mathbf{U}^\star) = \operatorname{cone}(\mathbf{X})$. Any other column $\mathbf{u}_k^{(t)}$ satisfies $\mathbf{u}_k^{(t)} \in \operatorname{cone}(\mathbf{U}^\star)$, hence, when tested in the *Remove redundant ray* step, $\mathbf{u}_k^{(t)} \in \operatorname{cone}\big(\mathbf{U}_{-k}^{(T)}\big)$ (up to tolerance ϵ) and is removed. Because only finitely many non-extreme columns exist (at most $m + n$ have ever been present) and each iteration of the *Removing* step in (b) removes at least one such column, after finitely many iterations of step (b) we achieve $\mathbf{U}^{(T)}$ consists of $\{\widehat{\mathbf{u}}_1, \ldots, \widehat{\mathbf{u}}_c\}$ in some order, and $\operatorname{cone}\big(\mathbf{U}^{(T)}\big) = \operatorname{cone}(\mathbf{U}^\star) = \operatorname{cone}(\mathbf{X})$. □

Dataset	Domain	#samples n	#classes C	#features m	Shape $\mathbf{X} \in \mathbb{R}^{m \times n}$
AMLALL	gene	38	3	5 000	$5\,000 \times 38$
DUKE	medical	44	2	7 129	$7\,129 \times 44$
KHAN	gene	83	4	2 318	$2\,318 \times 83$
CANCER	medical	198	14	16 063	$16\,063 \times 198$
ROSETTA	gene	300	5	12 634	$12\,634 \times 300$
MED	text	1 033	31	5 831	$5\,831 \times 1\,033$
CITESEER	text	3 312	6	3 703	$3\,703 \times 3\,312$
WEBKB4	text	4 196	4	10 000	$10\,000 \times 4\,196$
7SECTORS	text	4 556	7	10 000	$10\,000 \times 4\,556$
REUTERS	text	8 293	65	18 933	$18\,933 \times 8\,293$
RCV1	text	9 625	4	29 992	$29\,992 \times 9\,625$
ORL	image	400	40	10 304	$10\,304 \times 400$
UMIST	image	575	20	10 304	$10\,304 \times 575$
YALEB	image	1 292	38	32 256	$32\,256 \times 1\,292$
COIL-20	image	1 440	20	16 384	$16\,384 \times 1\,440$
CURETGREY	image	5 612	61	10 000	$10\,000 \times 5\,612$

Table 1: Dataset statistics for selected benchmarks (gene expression, text, and image) used in our NMF experiments. Here m is the original feature dimension (genes, words, or pixels) and n is the number of samples.

5 Experiment

Datasets. We evaluate Cone Collapse combined with orthogonal NMF (CC–NMF) on a diverse collection of 16 benchmark datasets covering gene expression, text, and image domains (Table 1). The gene expression sets (AMLALL, DUKE, KHAN, CANCER, ROSETTA) come from standard microarray studies for cancer subtyping and prognosis [11, 12, 13, 14, 15]. Text datasets (MED, CITESEER, WEBKB4, 7SECTORS, REUTERS, RCV1) are represented as term–document matrices using TF–IDF weighting, and follow common preprocessing pipelines used in information retrieval and text categorization [16, 17, 18, 19, 20]. Image datasets (ORL, UMIST, YALEB, COIL-20, CURETGREY) consist of vectorized grayscale faces or objects and are widely used in clustering and representation learning [21, 22, 23, 24, 25]. For all datasets we use the full feature dimensionality m and the full set of samples n as summarized in Table 1.

Baselines and experimental protocol. We compare CC–NMF against several representative NMF variants:

- **MU** – the classical NMF with multiplicative updates introduced by Lee and Seung [1, 2].
- **ANLS** – alternating nonnegative least squares with block principal pivoting, a fast and robust solver for constrained least squares [9].
- **PNMF** – projective NMF, which constrains the factorization to take the form $\mathbf{X} \approx \mathbf{X}\mathbf{G}\mathbf{G}^\top$ and is closely related to spectral clustering [3].
- **ONMF** – orthogonal NMF baselines that impose (approximate) orthogonality on one factor, following [4, 7, 5, 8].
- **Sparse NMF** – an ℓ_1-regularized NMF model that promotes sparse encodings in $\mathbf{H}$, implemented on top of ANLS [9].
- **CC–NMF** (ours) – the proposed two-stage approach that first extracts an extreme-ray basis $\mathbf{U}^\star$ using the Cone Collapse algorithm and then performs an orthogonal NMF refinement on $\mathbf{U}^\star$ to obtain an ONMF factorization of $\mathbf{X}$.

For all methods we fix the factorization rank to the number of ground-truth classes, $r = C$, and use the same preprocessed nonnegative matrix $\mathbf{X} \in \mathbb{R}_+^{m \times n}$. We run each algorithm from random nonnegative initializations and stop when the relative decrease of the objective falls below a preset threshold or a maximum number of iterations is reached. For MU, ANLS, PNMF, ONMF, and Sparse NMF we use standard update rules and hyperparameters as suggested in the original papers

Dataset	MU	ANLS	PNMF	ONMF	Sparse NMF	CC-NMF
AMLALL	0.91	0.91	0.92	0.92	0.90	**0.98**
DUKE	0.52	0.48	0.52	0.52	**0.54**	0.53
KHAN	0.57	0.58	0.60	0.60	0.59	**0.63**
CANCER	0.52	0.50	0.54	0.53	0.53	**0.56**
ROSETTA	0.68	0.76	0.77	0.77	0.78	**0.82**
MED	0.45	0.50	0.54	0.54	0.55	**0.59**
CITESEER	0.28	0.25	0.31	0.31	0.32	**0.35**
WEBKB4	0.31	0.37	0.39	0.39	0.41	**0.43**
7SECTORS	0.18	0.23	0.27	0.25	**0.28**	**0.28**
REUTERS	0.65	0.71	**0.74**	0.72	0.72	0.72
RCV1	0.23	0.29	**0.35**	0.31	0.28	0.33
ORL	0.78	0.81	0.82	0.82	0.84	**0.88**
UMIST	0.64	0.63	0.64	0.66	0.63	**0.68**
YALEB	0.39	0.38	0.42	**0.45**	**0.45**	**0.45**
COIL-20	0.55	0.64	0.71	0.65	0.62	**0.74**
CURETGREY	0.19	0.15	0.22	0.21	0.23	**0.31**

Table 2: Clustering purity of different NMF variants on various datasets. Boldface entries indicate the best value in each row.

[2, 9, 4, 5]. For Cone Collapse we set the learning rate η and tolerance ϵ to fixed values of 0.25 and 10^{-8} across datasets; empirical results indicate that CC–NMF is not overly sensitive to moderate changes of these hyperparameters.

Clustering is performed in the low-dimensional representation induced by the NMF factors. In ONMF-type methods (ONMF and CC–NMF), each column $\mathbf{x}_i$ is assigned to cluster

$$\widehat{c}(i) \;=\; \arg\max_{k\in\{1,\dots,r\}} h_{ki},$$

where $\mathbf{h}_i$ is the i-th column of $\mathbf{H}$. For MU, ANLS, PNMF, and Sparse NMF we apply the same rule to the corresponding $\mathbf{H}$ matrices, which is equivalent to using the learned components as soft cluster indicators. Clustering performance is evaluated using *purity*:

$$\text{purity} \;=\; \frac{1}{n}\sum_{k=1}^{r}\max_{1\le\ell\le C} n_k^{\ell},$$

where n_k^{ℓ} denotes the number of samples in cluster k whose true label is ℓ. A larger purity indicates better agreement between clusters and ground-truth classes.

Results. Table 2 reports clustering purity for all methods and datasets. Boldface entries indicate the best value in each row. Overall, CC–NMF consistently matches or outperforms the competing NMF variants across most benchmarks. It achieves the highest or tied-best purity on 13 out of 16 datasets, spanning all three domains (gene expression, text, and images). The gains are particularly pronounced on several high-dimensional problems such as ROSETTA, MED, CITESEER, WEBKB4, COIL-20, and CURETGREY, where CC–NMF improves purity by 3–8 points over the strongest baseline.

On a few datasets (e.g., DUKE and RCV1) traditional NMF variants remain competitive, suggesting that the benefit of explicitly recovering the extreme rays of the data cone is most significant when the underlying clusters are well aligned with conic structure. Nevertheless, even on these datasets CC–NMF is never dramatically worse than the best baseline, and it often provides a robust trade-off across all tasks. These results support our interpretation of Cone Collapse as a geometrically grounded orthogonal NMF method that yields high-quality clusterings across heterogeneous data types.

6 Conclusion

We proposed Cone Collapse, a new algorithm that explicitly leverages the conic geometry underlying nonnegative matrix factorization. Rather than optimizing a factorization objective directly

in the space of $\mathbf{W}$ and $\mathbf{H}$, Cone Collapse starts from the full nonnegative orthant and iteratively shrinks it toward the minimal cone that contains all data points. By combining mean-tilting toward the data mean, the addition of outside points, and the removal of redundant rays via NNLS tests, the algorithm converges to a compact set of extreme rays that summarize the data. Our theoretical analysis shows that, under mild assumptions, Cone Collapse terminates in finitely many iterations and recovers the minimal generating cone of $\mathbf{X}^\top$.

Building on this geometric foundation, we constructed CC–NMF, a cone-based orthogonal NMF model obtained by applying uni-orthogonal NMF to the recovered extreme-ray basis. This yields an ONMF factorization in which the orthogonal cluster-indicator matrix $\mathbf{H}$ is explicitly tied to the extreme rays of the data cone, providing a clear geometric interpretation that complements existing ONMF approaches [4, 7, 5, 8]. Empirically, CC–NMF achieves competitive or superior clustering purity to strong NMF baselines (MU, ANLS, PNMF, ONMF, sparse NMF) across a diverse suite of gene-expression, text, and image datasets, suggesting that explicitly modeling the data cone is beneficial in practice.

There are several avenues for future work. First, our analysis focuses on an idealized noiseless setting; extending the guarantees to noisy or approximately separable data, in the spirit of provable topic modeling and separable NMF [10], is an interesting direction. Second, Cone Collapse currently relies on repeated NNLS solves; it would be valuable to investigate more scalable approximation schemes or stochastic variants for very large-scale matrices. Third, while we have focused on clustering, the cone-based viewpoint may also prove useful for other tasks such as outlier detection, semi-supervised learning, and interpretable representation learning. We hope that this work stimulates further exploration of explicit conic geometry in NMF and related factorization models.

References

[1] Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[2] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems 13 (NIPS 2000)*, pages 556–562. MIT Press, 2001.

[3] Chris Ding, Xiaofeng He, and Horst D. Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. In *Proceedings of the 2005 SIAM International Conference on Data Mining (SDM)*, pages 606–610. SIAM, 2005.

[4] Chris Ding, Tao Li, Wei Peng, and Haesun Park. Orthogonal nonnegative matrix tri-factorizations for clustering. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 126–135, Philadelphia, PA, USA, 2006. ACM.

[5] Jiho Yoo and Seungjin Choi. Orthogonal nonnegative matrix factorization: Multiplicative updates on stiefel manifolds. In *Proceedings of the 9th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL 2008)*, volume 5326 of *Lecture Notes in Computer Science*, pages 140–147. Springer, 2008.

[6] Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003)*, pages 267–273, Toronto, Canada, 2003. ACM.

[7] Seungjin Choi. Algorithms for orthogonal nonnegative matrix factorization. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, Hong Kong, China, 2008. IEEE.

[8] Francesco Pompili, Nicolas Gillis, Przemyslaw R. Grzywacz, and Andrés L. Gómez. ONP-MF: An orthogonal nonnegative matrix factorization algorithm. In *Proceedings of the 21st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, pages 145–150, 2013.

[9] Jingu Kim and Haesun Park. Toward faster nonnegative matrix factorization: A new algorithm and comparisons. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, pages 353–362, Washington, DC, USA, 2008. IEEE Computer Society.

[10] Sanjeev Arora, Rong Ge, Yonatan Halpern, David Mimno, Ankur Moitra, David Sontag, Yichen Wu, and Michael Zhu. A practical algorithm for topic modeling with provable guarantees. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, volume 28 of *Proceedings of Machine Learning Research*, pages 280–288, Atlanta, Georgia, USA, 2013. PMLR.

[11] T. R. Golub et al. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, 1999.

[12] M. West, C. Blanchette, H. Dressman, et al. Predicting the clinical status of human breast cancer by using gene expression profiles. *Proceedings of the National Academy of Sciences of the USA*, 98(20):11462–11467, 2001.

[13] J. Khan, J. S. Wei, M. Ringnér, et al. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature Medicine*, 7(6):673–679, 2001.

[14] L. J. van 't Veer, H. Dai, M. J. van de Vijver, et al. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415(6871):530–536, 2002.

[15] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer, 2 edition, 2009.

[16] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[17] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.

[18] CMU Text Learning Group. Webkb4 and 7sectors text datasets. `http://www.cs.cmu.edu/~TextLearning/datasets.html`. Accessed 2025.

[19] David D. Lewis. Reuters-21578 text categorization test collection, distribution 1.0. `https://www.daviddlewis.com/resources/testcollections/reuters21578/`, 1997. Newswire documents from Reuters, 1987.

[20] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

[21] Ferdinando S. Samaria and Andy C. Harter. Parameterisation of a stochastic model for human face identification. In *Proceedings of the 2nd IEEE Workshop on Applications of Computer Vision (WACV)*, pages 138–142, Sarasota, FL, 1994.

[22] Daniel Graham and Nigel Allinson. The sheffield (formerly umist) face database. `https://www.sheffield.ac.uk/eee/research/iel/research/face`, 1998. Multi-pose face database with 564 images of 20 individuals.

[23] Athinodoros S. Georghiades, Peter N. Belhumeur, and David J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):643–660, 2001.

[24] Samer A. Nene, Shree K. Nayar, and Hiroshi Murase. Columbia object image library (coil-20). Technical Report CUCS-005-96, Department of Computer Science, Columbia University, 1996.

[25] Kristin J. Dana, Bram van Ginneken, Shree K. Nayar, and Jan J. Koenderink. Reflectance and texture of real-world surfaces. *ACM Transactions on Graphics*, 18(1), 1999.

A Proof for Lemma 4.1

Suppose, For contradiction, that $\mu = \tau\, u_k$ For some $k \in \{1, \ldots, c\}$. Let $\mathcal{U}$ be the set of points that is colinear with u_k. Note that we can write

$$\boldsymbol{\mu} = \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i = \frac{1}{m}(\sum_{i\notin\mathcal{U}}\mathbf{x}_i + \sum_{i\in\mathcal{U}}\mathbf{x}_i) = \frac{1}{m}(\sum_{i\notin\mathcal{U}}\mathbf{x}_i + \omega\ \mathbf{u}_k),$$

with some $\omega > 0$. We then have

$$\begin{aligned}
&\boldsymbol{\mu} = \tau\ \mathbf{u}_k \\
\Rightarrow\ &\frac{1}{m}(\sum_{i\notin\mathcal{U}}\mathbf{x}_i + \omega\ \mathbf{u}_k) = \tau\ \mathbf{u}_k \\
\Rightarrow\ &\frac{1}{m}\sum_{i\notin\mathcal{U}}\mathbf{x}_i = (\tau - \frac{\omega}{m})\ \mathbf{u}_k.
\end{aligned}$$

Consider three cases:

- If $\tau > \dfrac{\omega}{m}$, then $\mathbf{u}_k = \dfrac{1}{m\tau - \omega}\sum_{i\notin\mathcal{U}}\mathbf{x}_i$, or $\mathbf{u}_k$ is a conic combination of the remaining extreme rays, which contradicts the fact that $\mathbf{u}_k$ is an extreme ray.
- If $\tau = \dfrac{\omega}{m}$, since all datapoints lie in the positive region, $\mathbf{x}_i = 0\ \forall i \notin \mathcal{U}$. However, this contradicts the assumption that no column of $\mathbf{X}$ is full zero.

- If $\tau < \frac{\omega}{m}$, $\frac{1}{m}\sum_{i\notin\mathcal{U}} \mathbf{x}_i \notin \mathbb{R}^n_+$ or $\mathbf{u}_k \notin \mathbb{R}^n_+$, which contradicts the fact that all data points are in the positive region.

Therefore, if there are more than 1 extreme ray, then no extreme ray is colinear with the mean.

B Proof for Lemma 4.2

Since $\mathbf{u}_k^{(t)}$ and $\widehat{\boldsymbol{\mu}}$ are unit vectors,

$$\begin{aligned}
\cos(\widetilde{\mathbf{u}}_k^{(t)}, \widehat{\boldsymbol{\mu}}) = \widetilde{\mathbf{u}}_k^{(t)}\widehat{\boldsymbol{\mu}} &= \left(\frac{(1-\eta)\,\mathbf{u}_k^{(t)} + \eta\,\widehat{\boldsymbol{\mu}}}{\|(1-\eta)\,\mathbf{u}_k^{(t)} + \eta\,\widehat{\boldsymbol{\mu}}\|}\right)^\top \widehat{\boldsymbol{\mu}} \\
&= \frac{(1-\eta)\,\mathbf{u}_k^{(t)\top}\widehat{\boldsymbol{\mu}} + \eta\,\widehat{\boldsymbol{\mu}}^\top\widehat{\boldsymbol{\mu}}}{\sqrt{\left((1-\eta)\,\mathbf{u}_k^{(t)} + \eta\,\widehat{\boldsymbol{\mu}}\right)^\top\left((1-\eta)\,\mathbf{u}_k^{(t)} + \eta\,\widehat{\boldsymbol{\mu}}\right)}} \\
&= \frac{(1-\eta)\,\mathbf{u}_k^{(t)\top}\widehat{\boldsymbol{\mu}} + \eta}{\sqrt{(1-2\eta+\eta^2)\,\mathbf{u}_k^{(t)\top}\mathbf{u}_k^{(t)} + 2\eta(1-\eta)\,\mathbf{u}_k^{(t)\top}\widehat{\boldsymbol{\mu}} + \eta^2\,\widehat{\boldsymbol{\mu}}^\top\widehat{\boldsymbol{\mu}}}} \\
&= \frac{(1-\eta)\,\mathbf{u}_k^{(t)\top}\widehat{\boldsymbol{\mu}} + \eta}{\sqrt{1-2\eta+2\eta^2 + 2\eta(1-\eta)\,\mathbf{u}_k^{(t)\top}\widehat{\boldsymbol{\mu}}}}
\end{aligned}$$

We then have

$$\begin{aligned}
&\cos^2(\widetilde{\mathbf{u}}_k^{(t)}, \widehat{\boldsymbol{\mu}}) - \cos^2(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}}) \\
=&\left(\frac{(1-\eta)\,\mathbf{u}_k^{(t)\top}\widehat{\boldsymbol{\mu}} + \eta}{\sqrt{1-2\eta+2\eta^2 + 2\eta(1-\eta)\,\mathbf{u}_k^{(t)\top}\widehat{\boldsymbol{\mu}}}}\right)^2 - \cos^2(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}}) \\
=&\frac{\left((1-\eta)\cos(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}}) + \eta\right)^2 - \cos^2(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}})\,\left(1-2\eta+2\eta^2 + 2\eta(1-\eta)\cos(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}})\right)}{1-2\eta+2\eta^2 + 2\eta(1-\eta)\cos(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}})}
\end{aligned}$$

Since $\mathbf{u}_k^{(t)} \neq \widehat{\boldsymbol{\mu}}$, $1 - \cos^2(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}}) > 0$. Hence,

$$\begin{aligned}
&\left((1-\eta)\cos(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}}) + \eta\right)^2 - \cos^2(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}})\,\left(1-2\eta+2\eta^2 + 2\eta(1-\eta)\cos(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}})\right) \\
=&\,(1-2\eta+\eta^2)\cos^2(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}}) + 2\eta\,(1-\eta)\cos(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}}) + \eta^2 - (1-2\eta+2\eta^2)\cos^2(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}}) - \\
&\,2\eta(1-\eta)\cos^3(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}}) \\
=&\,\eta^2\,\left(1-\cos^2(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}})\right) + 2\eta\,(1-\eta)\cos(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}})\,\left(1-\cos^2(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}})\right) > 0 \quad \text{for } 0<\eta<1
\end{aligned}$$

We also know that $\left(1-2\eta+2\eta^2+2\eta(1-\eta)\cos(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}})\right) > 0$. Therefore, $\cos^2(\widetilde{\mathbf{u}}_k^{(t)}, \widehat{\boldsymbol{\mu}}) - \cos^2(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}}) > 0$, or $\cos^2(\mathbf{u}_k^{(t+1)}, \widehat{\boldsymbol{\mu}}) - \cos^2(\mathbf{u}_k^{(t)}, \widehat{\boldsymbol{\mu}}) > 0$ if $\mathbf{u}_k^{(t)}$ is not removed after step t.

C Proof for Lemma 4.3

Since $b_\ell \in \mathcal{C}_\alpha$,

$$\cos(\mathbf{w}, \widehat{\boldsymbol{\mu}}) = \frac{\mathbf{w}^\top\widehat{\boldsymbol{\mu}}}{\|\mathbf{w}\|} = \frac{\sum_\ell \lambda_\ell \mathbf{b}_\ell^\top\widehat{\boldsymbol{\mu}}}{\|\mathbf{w}\|} = \frac{\sum_\ell \lambda_\ell \cos(\mathbf{b}_\ell, \widehat{\boldsymbol{\mu}})\|\mathbf{b}_\ell\|}{\|\mathbf{w}\|} \geq \frac{\sum_\ell \lambda_\ell \alpha}{\|\mathbf{w}\|}$$

Moreover, by Triangle inequality

$$\|\mathbf{w}\| = \|\sum_\ell \lambda_\ell \mathbf{b}_\ell\| \leq \sum_\ell \lambda_\ell \|\mathbf{b}_\ell\| = \sum_\ell \lambda_\ell,$$

which means

$$\cos(\mathbf{w}, \widehat{\boldsymbol{\mu}}) \geq \frac{\sum_\ell \lambda_\ell \alpha}{\sum_\ell \lambda_\ell} = \alpha,$$

hence $\widehat{\mathbf{w}} \in \mathcal{C}_\alpha$.

D Block Principal Pivoting For Non-Negative Least Square (NNLS) problem

NMF is typically computed by alternating updates of two nonnegative factors, where each update step reduces to a set of NNLS problems. Consequently, the overall efficiency and scalability of NMF hinge on how quickly these NNLS subproblems can be solved. Among various NNLS algorithms, we adopt the principal block pivoting method [9] because it is a fast active set-like scheme that handles large numbers of variables and multiple right–hand sides very efficiently. In the following, we briefly review BPP for the single right–hand side case and its extension to multiple right–hand sides.

Single right–hand sides. We consider the NNLS problem

$$\min_{\mathbf{x} \geq 0} \ \|\mathbf{C}\mathbf{x} - \mathbf{b}\|_2^2, \tag{3}$$

where $\mathbf{C} \in \mathbb{R}^{m \times n}$ and $\mathbf{b} \in \mathbb{R}^m$. The KKT conditions for (3) are

$$\begin{aligned} \mathbf{y} &= \mathbf{C}^\top \mathbf{C}\mathbf{x} - \mathbf{C}^\top \mathbf{b}, \\ \mathbf{y} &\geq 0, \\ \mathbf{x} &\geq 0, \\ \mathbf{x}_i \, \mathbf{y}_i &= 0, \quad i = 1, \ldots, n. \end{aligned}$$

Block principal pivoting maintains a partition of the indices $\{1, \ldots, n\}$ into a *free set* F and an *active set* G with $F \cup G = \{1, \ldots, n\}$ and $F \cap G = \emptyset$. Given (F, G), we set $\mathbf{x}_G = 0$ and $\mathbf{y}_F = 0$ and compute

$$\mathbf{x}_F = \arg \min_{\mathbf{z} \in \mathbb{R}^{|F|}} \left\| \mathbf{C}_F \mathbf{z} - \mathbf{b} \right\|_2^2, \tag{4a}$$

$$\mathbf{y}_G = \mathbf{C}_G^\top \big(\mathbf{C}_F \mathbf{x}_F - \mathbf{b} \big), \tag{4b}$$

where $\mathbf{C}_F$ (respectively $\mathbf{C}_G$) contains columns of $\mathbf{C}$ indexed by F (respectively G). We then define the sets of infeasible indices

$$H_1 = \{\, i \in F : \mathbf{x}_i < 0 \,\}, \qquad H_2 = \{\, i \in G : \mathbf{y}_i < 0 \,\},$$

and exchange blocks $\widehat{H}_1 \subseteq H_1$, $\widehat{H}_2 \subseteq H_2$ between F and G. If $H_1 \cup H_2 = \emptyset$, all KKT conditions are satisfied and the algorithm terminates. The detailed procedure is given in Algorithm 2.

Multiple right–hand sides. We now consider the NNLS problem with multiple right–hand sides

$$\min_{\mathbf{X} \geq 0} \ \|\mathbf{C}\mathbf{X} - \mathbf{B}\|_F^2, \tag{5}$$

where $\mathbf{C} \in \mathbb{R}^{m \times n}$, $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_r] \in \mathbb{R}^{m \times r}$, and $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_r] \in \mathbb{R}^{n \times r}$. Each column $\mathbf{x}_j$ solves an NNLS problem of the form (3) with the same coefficient matrix $\mathbf{C}$. A naive approach is to run Algorithm 2 independently for $j = 1, \ldots, r$; however, this ignores the shared structure of $\mathbf{C}$.

The block principal pivoting method for multiple right–hand sides exploits this structure by precomputing $\mathbf{G} = \mathbf{C}^\top \mathbf{C}$ and $\mathbf{H} = \mathbf{C}^\top \mathbf{B}$, and by grouping columns that share a common free set. For each column j, we maintain free and active index sets F_j, G_j and corresponding primal/dual variables $\mathbf{x}_j, \mathbf{y}_j$ with $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_r]$. Given a group of columns $\mathcal{J}$ that share the same free set F, we solve the normal equations

$$\mathbf{G}_{FF} \, \mathbf{X}_{F,\mathcal{J}} = \mathbf{H}_{F,\mathcal{J}}, \tag{6a}$$

$$\mathbf{Y}_{G,\mathcal{J}} = \mathbf{G}_{GF} \, \mathbf{X}_{F,\mathcal{J}} - \mathbf{H}_{G,\mathcal{J}}, \tag{6b}$$

Algorithm 2 Block principal pivoting for NNLS with a single right–hand side

Input: $\mathbf{C} \in \mathbb{R}^{m \times n}$, $\mathbf{b} \in \mathbb{R}^m$.
$F \leftarrow \emptyset, \quad G \leftarrow \{1, \ldots, q\}, \quad \mathbf{x} \leftarrow 0, \quad \mathbf{y} \leftarrow -\mathbf{C}^\top \mathbf{b}, \quad p \leftarrow 3, \quad t \leftarrow q + 1.$
Compute $\mathbf{x}_F$ and $\mathbf{y}_G$ by (4).
repeat
  $H_1 \leftarrow \{i \in F : \mathbf{x}_i < 0\}$, $H_2 \leftarrow \{i \in G : \mathbf{y}_i < 0\}$.
  if $H_1 \cup H_2 = \emptyset$ **then**
    break
  end if
  if $|H_1 \cup H_2| < t$ **then**
    $t \leftarrow |H_1 \cup H_2|$, $p \leftarrow 3$;
    $\widehat{H}_1 \leftarrow H_1$, $\widehat{H}_2 \leftarrow H_2$.
  else if $|H_1 \cup H_2| \geq t \ \wedge \ p \geq 1$ **then**
    $p \leftarrow p - 1$;
    $\widehat{H}_1 \leftarrow H_1$, $\widehat{H}_2 \leftarrow H_2$.
  else
    Choose $i^\star$ as the largest index in $H_1 \cup H_2$.
    $\widehat{H}_1 \leftarrow \{i^\star\} \cap F$, $\widehat{H}_2 \leftarrow \{i^\star\} \cap G$.
  end if
  Update index sets

$$F \leftarrow (F \setminus \widehat{H}_1) \cup \widehat{H}_2, \qquad G \leftarrow (G \setminus \widehat{H}_2) \cup \widehat{H}_1.$$

  Recompute $\mathbf{x}_F$ and $\mathbf{y}_G$ by (4).
until all variables are feasible
Output: $\mathbf{x}$

Algorithm 3 Block principal pivoting for NNLS with multiple right–hand sides

Input: $\mathbf{C} \in \mathbb{R}^{m \times n}$, $\mathbf{B} \in \mathbb{R}^{m \times r}$.
Precompute $\mathbf{G} \leftarrow \mathbf{C}^\top \mathbf{C}$, $\mathbf{H} \leftarrow \mathbf{C}^\top \mathbf{B}$.
Initialize $\mathbf{X} \leftarrow 0$, $\mathbf{Y} \leftarrow -\mathbf{H}$; for all j, set $F_j \leftarrow \emptyset$, $G_j \leftarrow \{1, \ldots, n\}$, $P_j \leftarrow 3$, $T_j \leftarrow n + 1$.
repeat
  Reorder columns of $\mathbf{X}$ and $\mathbf{Y}$ to group those with a common free set.
  For each group $\mathcal{J}$ with free set F, update $\mathbf{X}_{F,\mathcal{J}}$ and $\mathbf{Y}_{G,\mathcal{J}}$ using (6).
  For each column j, form $H_1(j)$, $H_2(j)$, choose $\widehat{H}_1(j)$, $\widehat{H}_2(j)$ using T_j, P_j, and update F_j, G_j accordingly.
until $H_1(j) \cup H_2(j) = \emptyset$ for all j
Output: $\mathbf{X}$

where $\mathbf{X}_{F,\mathcal{J}}$ (resp. $\mathbf{Y}_{G,\mathcal{J}}$) collects the rows indexed by F (resp. G) and columns in $\mathcal{J}$, and $\mathbf{G}_{FF}, \mathbf{G}_{GF}$ are the corresponding submatrices of $\mathbf{G}$. As in the single right–hand side case, we define for each column j

$$H_1(j) = \{\, i \in F_j : \mathbf{x}_{ij} < 0 \,\}, \qquad H_2(j) = \{\, i \in G_j : \mathbf{y}_{ij} < 0 \,\},$$

and move blocks $\widehat{H}_1(j) \subseteq H_1(j)$, $\widehat{H}_2(j) \subseteq H_2(j)$ between F_j and G_j, using the same block/backup exchange strategy as in Algorithm 2. If $H_1(j) \cup H_2(j) = \emptyset$ for all j, all columns satisfy the KKT conditions and the algorithm terminates. Algorithm 3 summarizes the procedure.