Fast Conical Hull Algorithms for Near-separable Non-negative Matrix Factorization

Abhishek Kumar ABHISHEK@CS.UMD.EDU
Dept. of Computer Science, University of Maryland, College Park, MD 20742, USA

Vikas Sindhwani VSINDHW@US.IBM.COM
IBM T.J. Watson Research Center, Yorktown Heights, NY 10598 USA

Prabhanjan Kambadur PKAMBADU@US.IBM.COM
IBM T.J. Watson Research Center,Yorktown Heights, NY 10598 USA

Abstract

The separability assumption (Donoho & Stodden, 2003; Arora et al., 2012a) turns non-negative matrix factorization (NMF) into a tractable problem. Recently, a new class of provably-correct NMF algorithms have emerged under this assumption. In this paper, we reformulate the separable NMF problem as that of finding the extreme rays of the conical hull of a finite set of vectors. From this geometric perspective, we derive new separable NMF algorithms that are highly scalable and empirically noise robust, and have several other favorable properties in relation to existing methods. A parallel implementation of our algorithm demonstrates high scalability on shared- and distributed-memory machines.

1. Introduction

A data matrix $\mathbf{X}$ of size $m \times n$ is said to admit a Non-negative Matrix Factorization (NMF) with inner-dimension r, if $\mathbf{X}$ can be expressed as $\mathbf{X} = \mathbf{WH}$ where $\mathbf{W}, \mathbf{H}$ are two non-negative matrices of dimensions $m \times r$ and $r \times n$ respectively. In many applications, a compact (i.e., small r) approximate NMF tends to provide a natural and interpretable part-based decomposition of the data (Lee & Seung, 1999), which is appealing in many applications, e.g., modeling topics in text and hyperspectral image analysis (Cichocki et al., 2009).

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28. Copyright 2013 by the author(s).

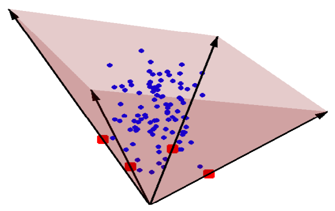

Figure 1. Geometry of the NMF Problem.

Figure 1 shows the geometry of the NMF problem. As a point cloud in $\mathbb{R}^m$, all the n columns of $\mathbf{X}$ are contained inside a cone that is generated by r non-negative vectors in $\mathbb{R}^m$ comprising the columns of $\mathbf{W}$. For any matrix $\mathbf{A}$, let *cone*$(\mathbf{A})$ denote the set obtained by taking linear combinations of the columns of $\mathbf{A}$ with non-negative coefficients. Then, the goal is to find a non-negative matrix $\mathbf{W}$, with just r columns, such that: $cone(\mathbf{X}) \subseteq cone(\mathbf{W}) \subseteq \mathbb{R}^m_+$, where $\mathbb{R}^m_+$ denotes the non-negative orthant in $\mathbb{R}^m$. Such polyhedral nesting problems studied in computational geometry are known to be NP-hard, which makes the exact and approximate NMF problem also NP-hard (Vavasis, 2009). Faced with such results, almost the entire algorithmic focus in the NMF literature, e.g., Cichocki et al. (2009); Lee & Seung (1999); Lin (2007); Hsieh & Dhillon (2011), has centered on treating the problem as an instance of general non-convex programming, leading to heuristic procedures that lack optimality guarantees beyond convergence to a stationary point of the objective function for approximate NMF. Very recently, in a series of elegant papers (Arora et al., 2012a;b; Bittorf et al., 2012; Gillis & Vavasis, 2012; Esser et al., 2012; Elhamifar et al., 2012), promising alternative approaches have been developed based on certain *separability* assumption on the data which enables the NMF problem to be solved exactly. Geometrically, the assumption states the following: *all columns of* $\mathbf{X}$ *reside in a cone generated by a small subset of* r *columns of* $\mathbf{X}$. In algebraic terms, $\mathbf{X} = \mathbf{WH} = \mathbf{X}_A\mathbf{H}$ so that the r columns of $\mathbf{W}$ are hidden among the columns of $\mathbf{X}$ (indexed by an unknown subset of indices A). Equivalently, a corresponding subset of r columns of $\mathbf{H}$ happen to constitute the $r \times r$ identity matrix. We refer to these columns as *anchors* (Arora et al., 2012a). Informally, in the context of topic modeling problems where $\mathbf{X}$ is a document-word matrix and $\mathbf{W}, \mathbf{H}$ are document-topic and topic-term associations respectively, the separability assumption equivalently posits the existence of special *anchor words* in the vocabulary, whose occurence uniquely identifies the presence of a topic, and whose usage across the corpus is collectively predictive of the usage of all the other words. The separability assumption was investigated earlier by Donoho & Stodden (2003) in the context of deriving uniqueness conditions for NMF. In order to place our contributions in the right context, we first briefly provide a flavor of recently proposed separable NMF algorithms.

Related Work: Assuming that the columns of $\mathbf{X}$ are normalized to have unit l_1-norm, the separable NMF problem reduces to that of finding the extreme points (that is, points inexpressible as convex combinations of other points) of the convex hull of the columns (Arora et al., 2012a). A Linear Program (LP) can be setup to attempt to express a given column as a convex combination of the other columns. If this LP declares infeasibility, an extreme point is identified. This approach (Arora et al., 2012a, Section5) requires solving n feasibility LP's each involving $n-1$ variables which is not scalable for many problems of interest. A noise-robust version of the procedure further requires knowledge of parameters that are hard to estimate apriori. Bittorf et al. (2012) formulate a single LP whose solution resolves the exactly separable NMF problem. An extension is also developed for noise-robustness. Instead of invoking a general LP solver, a specialized algorithm is derived based on an incremental stochastic gradient descent procedure, and its parallel (multithreaded) implementation is benchmarked on large datasets. On the other hand, this algorithm requires estimates of primal and dual step sizes, converges only asymptotically, and does not explicitly exploit the sparsity of the final solution. Gillis & Vavasis (2012) develop a highly scalable approach closely related to rank-revealing QR factorizations for column subset selection. A perturbation analysis of this algorithm under noise is also presented. In Esser et al. (2012), column subset selection is cast essentially as a form of multivariate regression with row-sparsity inducing norms, e.g., see Bien et al. (2010). Algorithms derived in this framework are asymptotically convergent, and sensitive to near-duplicate columns, making it necessary to perform certain adhoc preprocessing steps.

Contributions: We present a new family of highly scalable and empirically noise-robust algorithms for separable NMFs, with several favorable properties:

- The algorithms produce a correct solution for the separable case after exactly r iterations. They require no additional parameters. They are closely related to convex and conical hull finding procedures proposed in the computational geometry literature (Clarkson, 1994; Dula et al., 1998). Computationally, the algorithms bear some resemblence to simultaneous Orthogonal Matching Pursuit (Buhlmann & Geer, 2010; Tropp et al., 2006) for sparse greedy reconstruction of multiple target variables from the same subset of input variables. We also derive a variant based on this connection that performs quite well under noise.
- Under controlled noise conditions in synthetic datasets and on real-world topic modeling problems, our algorithms consistently outperform other separable NMF techniques with respect to multiple performance metrics. Note, however, that a formal noise analysis in support of these empirical findings is not provided in the current paper. Our methods are also highly competitive with existing non-convex NMF algorithms, but are free of sub-optimal local minima and associated initialization issues.
- The solution for $(r-1)$ target anchors is contained in the solution for r target anchors (unlike non-convex NMF methods), which makes it easier to do model selection on real-world datasets by keeping track of performance on a validation set.
- The algorithms are highly scalable and have small memory footprint. The sparsity of the data, the intermediate variables and the final solution is carefully exploited in a high-performance parallel and distributed implementation which scales excellently on both shared- and distributed-memory machines. For example, a twitter corpus with 125-thousand tweets can be factorized for $r = 100$ in less than 10 seconds on a commodity 8-core machine.
- Column normalization, as suggested in prior work, is not needed in our approach. Such normalization tends to interfere with the TFIDF weightings routinely used in text modeling applications, leading to performance loss.
- Unlike Esser et al. (2012), the algorithms do not require any special preprocessing to eliminate duplicate or near-duplicate columns.

2. Fast Conical Hull Algorithms

An informal description: Figure 2 provides some geometric intuition underlying the proposed approach. The algorithm executes r iterations. In each iteration a new anchor column is identified. This corresponds to expanding a cone one extreme ray at a time, until the entire dataset is eventually contained in the cone defined by the full set of anchors. Figure 2 illustrates one step of the algorithm where there is an existing cone defined by three extreme rays (marked 1 to 3). To identify the next extreme ray, the algorithm picks a point outside the current cone (a green point) and projects it to the current cone to compute a residual vector (we call this the *projection step*). This residual vector separates the current cone from at least one non-selected extreme ray that can be found by maximizing a specific selection criteria (we call this the *detection step*). Intuitively, the algorithm picks a face of the current cone (spanned by rays 1 and 3 in Figure 2) that "sees" exterior points and rotates this face towards the exterior until it hits the "last" point. In the example shown in Figure 2, ray 4 is identified as a new extreme ray. These geometric intuitions are inspired

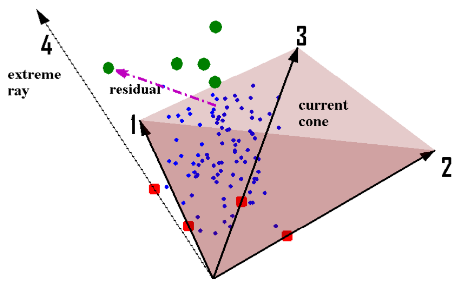

Figure 2. Geometry of the Conical Hull algorithms

by Clarkson (1994); Dula et al. (1998) who present LP-based algorithms for general convex and conical hull problems. Their algorithms are also directly applicable in our NMF setting, provided the data satisfies the separability assumption exactly. In this case, the residual of *any* single exterior point can be used to correctly expand the cone as described above. However, anchor detection criteria derived from *multiple* residuals demonstrates radically superior noise robustness, as we report in the experimental section. The emphasis on scalability and noise-robustness thus leads us to a new family of algorithms whose implementation (and associated proof of correctness) is distinct from prior work.

Cones, Extreme Rays and Projection: Here, we provide a short background on relevant geometric concepts and set some notation. Recall that a *cone* $\mathcal{C}$ is a non-empty convex set that is closed with respect to taking conic combinations (i.e., linear combinations with non-negative coefficients) of its elements. A *ray* in $\mathcal{C}$ generated by a vector $\mathbf{x} \neq 0 \in \mathcal{C}$ is the set of all vectors $\{t\mathbf{x} : t \geq 0\}$. A ray R is an *extreme ray* if its generators cannot be expressed by taking conic combinations of elements in $\mathcal{C}$ that do not themselves belong to R. A cone is called *finitely generated* if its elements are conic combinations of a finite set of vectors, and *pointed* if it does not contain both a vector $\mathbf{x}$ as well as its negation $-\mathbf{x}$. A fundamental result (e.g., see (Nemirovski, 2010)) states: *a pointed, finitely generated cone $\mathcal{C}$ possesses a finite and unique set of extreme rays, and $\mathcal{C}$ is the conical hull of the generators of these extreme rays.* Furthermore, the generators of these extreme rays are a subset of the finite set of vectors used to originally express the cone. In the NMF context, note that any cone contained in $\mathbb{R}^m_+$ is pointed. This implies that $cone(\mathbf{X})$ can also be described by a minimally compact set of generators, i.e., $cone(\mathbf{X}) = cone(\mathbf{X}_A)$ where A uniquely indexes the extreme rays (anchors). Thus, a non-negative matrix $\mathbf{X}$ admits a separable NMF with inner-dimension r if the number of extreme rays of $cone(\mathbf{X})$, i.e. size of A, coincides with r. A face of a cone is the intersection between the cone and a supporting hyperplane. The projection of a point $\mathbf{x}$ onto the cone generated by columns of a matrix $\mathbf{W}$, i.e. computing $\boldsymbol{z}^\star = \arg\min_{\boldsymbol{z}\in cone(\mathbf{W})} \|\mathbf{x} - \boldsymbol{z}\|_2^2$, can be obtained by solving a non-negative least squares problem, i.e., computing $\boldsymbol{h}^\star = \arg\min_{\boldsymbol{h}\geq 0} \|\mathbf{x} - \mathbf{W}\boldsymbol{h}\|_2^2$ and setting $\boldsymbol{z}^\star = \mathbf{W}\boldsymbol{h}^\star$. All columns of $\mathbf{X}$ can be simultaneously projected by solving $\mathbf{H}^\star = \arg\min_{\mathbf{H}\geq 0} \|\mathbf{X} - \mathbf{W}\mathbf{H}\|_2^2$. We will use the notation $\mathbf{R}$ to denote the residual matrix after a projection operation, i.e., $\mathbf{R} = \mathbf{X} - \mathbf{W}\mathbf{H}^\star$. We will use the notation $\mathbf{X}_i, \mathbf{R}_i$ to denote the i^{th} column of $\mathbf{X}$ and its corresponding residual. The notation $\mathbf{q}_+$ will denote the vector obtained by setting all negative entries of the vector $\mathbf{q}$ to 0.

Numerical Description: Algorithm 1 details the steps of the proposed family of algorithms which we call XRAY . Each iteration consists of two steps: (i) a *detection step* that finds a column(s) of $\mathbf{X}$ to be added as an anchor, and (ii) a *projection step* where all data points are projected onto the current cone to get the residuals. Projection is done by solving simultaneous nonnegative least squares problem using Algorithm 2. Every residual vector $\mathbf{R}_i$ obtained after the projection step is normal to one of the faces of the current cone. In the selection step, we pick a face of the current cone (identified by its normal $\mathbf{R}_i$), normalize all the data points to lie on the hyperplane $\mathbf{p}^T\mathbf{x} = 1$ $\left(\mathbf{Y}_j = \frac{\mathbf{X}_j}{\mathbf{p}^T\mathbf{X}_j}\right)$ for a strictly positive vector $\mathbf{p}$, and expand the current cone by selecting an extreme ray that maximizes the

```
Algorithm 1 XRAY : Algorithms for Separable NMF
  Input: $\mathbf{X} \in \mathbb{R}_+^{m \times n}$, inner dimension $r$
  Output: $\mathbf{W} \in \mathbb{R}_{m \times r}, \mathbf{H} \in \mathbb{R}_{r \times n}$, $r$ indices in $A$
          such that: $\mathbf{X} = \mathbf{WH}$, $\mathbf{W} = \mathbf{X}_A$
  Initialize: $\mathbf{R} \leftarrow \mathbf{X}$, $A \leftarrow \{\}$
  while $|A| < r$ do
    1. Detection Step: Find an extreme ray.
    Below, $\mathbf{p}$ is a strictly positive vector (not collinear with $\mathbf{R}_i$):

      $j^\star = \arg\max_j \frac{\mathbf{R}_i^T \mathbf{X}_j}{\mathbf{p}^T \mathbf{X}_j}$ for any $i : \|\mathbf{R}_i\|_2 > 0$   (1)

    Some exterior point selection criteria:

      rand :  any random $i : \|\mathbf{R}_i\|_2 > 0$   (2)
      max :   $i = \arg\max_k \|\mathbf{R}_k\|_2$   (3)
      dist :  $i = \arg\max_k \| \left(\mathbf{R}_k^T \mathbf{X}\right)_+ \|_2$   (4)

    A greedy variant:  $j^* = \arg\max_j \frac{\|(\mathbf{R}^T \mathbf{X}_j)_+\|_2^2}{\|\mathbf{X}_j\|_2^2}$   (5)

    2. Update: $A \leftarrow A \cup \{j^*\}$ (see Remarks)
    3. Projection Step: Project onto current cone.

      $\mathbf{H} = \arg\min_{\mathbf{B} \geq 0} \|\mathbf{X} - \mathbf{X}_A \mathbf{B}\|_2^2$ (Algorithm 2)   (6)

    4. Update Residuals (not explicitly): $\mathbf{R} = \mathbf{X} - \mathbf{X}_A \mathbf{H}$
  end while
```

inner product $\mathbf{R}_i^T \mathbf{Y}_j$. The selection step can be implemented in various ways - some options are listed in Algorithm 1.

To show that XRAY correctly identifies all the extreme rays, we need the following lemmas.

Lemma 2.1. *The residual matrix* $\mathbf{R}$, *obtained after projection of columns of* $\mathbf{X}$ *onto the current cone satisfies* $\mathbf{R}^T \mathbf{X}_A \leq 0$, *where* $\mathbf{X}_A$ *are the extreme rays of the current cone.*

Proof. Residuals are given by $\mathbf{R} = \mathbf{X} - \mathbf{X}_A \mathbf{H}$, where $\mathbf{H} = \arg\min_{\mathbf{B} \geq 0} \|\mathbf{X} - \mathbf{X}_A \mathbf{B}\|_F^2$.
Forming the Lagrangian for Eq. 6, we get $L(\mathbf{B}, \mathbf{\Lambda}) = \|\mathbf{X} - \mathbf{X}_A \mathbf{B}\|_F^2 - tr(\mathbf{\Lambda}^T \mathbf{B})$, where the matrix $\mathbf{\Lambda}$ contains the nonnegative Lagrange multipliers. Differentiating w.r.t. $\mathbf{B}$ and evaluating at the optimum $\mathbf{B} = \mathbf{H}$, we have the following from the KKT conditions: $2\mathbf{X}_A^T(\mathbf{X}_A \mathbf{H} - \mathbf{X}) - \mathbf{\Lambda} = 0$
$\Rightarrow -2\mathbf{X}_A^T \mathbf{R} = \mathbf{\Lambda} \geq 0 \quad \Rightarrow \mathbf{R}^T \mathbf{X}_A \leq 0$ □

Lemma 2.2. *For any point* $\mathbf{X}_i$ *exterior to the current cone, we have* $\mathbf{R}_i^T \mathbf{X}_i > 0$, *where* $\mathbf{R}_i$ *is the residual of* $\mathbf{X}_i$ *obtained by projecting it onto the current cone.*

Proof. Let $\mathbf{R} = \mathbf{X} - \mathbf{X}_A \mathbf{H}$, where $\mathbf{H} = \arg\min_{\mathbf{B} \geq 0} \|\mathbf{X} - \mathbf{X}_A \mathbf{B}\|_F^2$ and $\mathbf{X}_A$ are the extreme rays of the current cone. From the KKT conditions (used in the proof of Lemma 2.1) we have $2\mathbf{R}^T \mathbf{X}_A = -\mathbf{\Lambda}^T$, where $\mathbf{\Lambda}$ are the Lagrange multipliers. Hence, $2\mathbf{R}_i^T \mathbf{X}_A = -\mathbf{\Lambda}_i^T$ (ith row of both left and right side matrices). From the complementary slackness property, we have $\mathbf{\Lambda}_{ji} \mathbf{H}_{ji} = 0 \; \forall j, i$. Hence, $2\mathbf{R}_i^T \mathbf{X}_A \mathbf{H}_i = -\mathbf{\Lambda}_i^T \mathbf{H}_i = 0$.
Hence we have $\mathbf{R}_i^T \mathbf{X}_i = \mathbf{R}_i^T(\mathbf{R}_i + \mathbf{X}_A \mathbf{H}_i) = \|\mathbf{R}_i\|_2^2 + \mathbf{R}_i^T \mathbf{X}_A \mathbf{H}_i = \|\mathbf{R}_i\|_2^2 > 0$ since $\mathbf{R}_i \neq 0$. □

```
Algorithm 2 Solver for: $\arg\min_{\mathbf{B} \geq 0} \|\mathbf{X} - \mathbf{X}_A \mathbf{B}\|_2^2$
  Input: $\mathbf{X} \in \mathbb{R}^{m \times n}$, Index set $A$ with $r$ indices, Initial value for warm-starts: $\mathbf{B}_{init} \in \mathbb{R}^{r \times n}$
  Convergence paramaters $tol$, $maxcycles$
  Initialize: $\mathbf{B} = \mathbf{B}_{init}$
  Set $\mathbf{C} = \mathbf{X}^T \mathbf{X} \in \mathbb{R}^{n \times n}$, $\mathbf{S} = \mathbf{C}_{A,A} \in \mathbb{R}^{r \times r}$, $\boldsymbol{s} = diag(\mathbf{S})$,
  $\mathbf{U} = \mathbf{B}^T \mathbf{S} \in \mathbb{R}^{n \times r}$
  while true do
    for $i = 1 \ldots r$ (//cyclic coordinate descent) do
      $\mathbf{b} = (\mathbf{B}^T)_i$
      $\mathbf{u} = \mathbf{U}_i - s_i \mathbf{b}$
      $(\mathbf{B}^T)_i = s_i^{-1}(\mathbf{C}_i - \mathbf{u})_+$
      $\mathbf{U} = \mathbf{U} + \left((\mathbf{B}^T)_i - \mathbf{b}\right) \mathbf{S}_i^T$   // sparse rank-1 update
    end for
    $objective = \|\mathbf{X}\|^2 + \sum_{i=1}^r (\mathbf{U}_i + \mathbf{C}_i)^T (\mathbf{B}^T)_i$
    Exit: $\Delta objective < tol$, or #iters $> maxcycles$.
  end while
```

Using the above two lemmas, we prove the following theorem regarding the correctness of Algorithm 1.

Theorem 2.1. *The data point* $\mathbf{X}_{j^*}$ *added at each iteration in the Detection step of Algorithm 1, if the maximizer in Eqn. 1 is unique, is an extreme ray of* $\mathcal{C}$ *that has not been selected in previous iterations.*

Proof. Let the index set A identify all the extreme rays of $\mathcal{C}$. Under the separability assumption, we have $\mathbf{X} = \mathbf{X}_A \mathbf{H}$. Let the index set A^t identify the extreme rays of the current cone $\mathcal{C}^t$.

Let $\mathbf{Y}_j = \frac{\mathbf{X}_j}{\mathbf{p}^T \mathbf{X}_j}$ and $\mathbf{Y}_A = \mathbf{X}_A [diag(\mathbf{p}^T \mathbf{X}_A)]^{-1}$ (since $\mathbf{p}$ is strictly positive, the inverse exists). Hence $\mathbf{Y}_j = \mathbf{Y}_A \frac{[diag(\mathbf{p}^T \mathbf{X}_A)] \mathbf{H}_j}{\mathbf{p}^T \mathbf{X}_j}$. Let $\mathbf{C}_j = \frac{[diag(\mathbf{p}^T \mathbf{X}_A)] \mathbf{H}_j}{\mathbf{p}^T \mathbf{X}_j}$. We also have $\mathbf{p}^T \mathbf{Y}_j = 1$ and $\mathbf{p}^T \mathbf{Y}_A = \mathbf{1}^T$. Hence, we have $1 = \mathbf{p}^T \mathbf{Y}_j = \mathbf{p}^T \mathbf{Y}_A \mathbf{C}_j = \mathbf{1}^T \mathbf{C}_j$.

Using Lemma 2.1, Lemma 2.2 and the fact that $\mathbf{p}$ is strictly positive, we have $\max_{1 \leq j \leq n} \mathbf{R}_i^T \mathbf{Y}_j = \max_{j \notin A^t} \mathbf{R}_i^T \mathbf{Y}_j$. Indeed, for all $j \in A^t$ we have $\mathbf{R}_i^t \mathbf{Y}_j \leq 0$ using Lemma 2.1 and there is at least one $j = i \notin A^t$ for which $\mathbf{R}_i^t \mathbf{Y}_j > 0$ using Lemma 2.2. Hence the maximum lies in the set $\{j : j \notin A^t\}$.

Further, we have $\max_{j \notin A^t} \mathbf{R}_i^T \mathbf{Y}_j = \max_{j \notin A^t} \mathbf{R}_i^T \mathbf{Y}_A \mathbf{C}_j \leq \max_{j \in (A \setminus A^t)} \mathbf{R}_i' \mathbf{Y}_j$. The second inequality is the result of the fact that $\|\mathbf{C}_j\|_1 = 1$ and $\mathbf{C}_j \geq 0$. This implies that if there is a unique maximum at a $j^* = \arg\max_{j \notin A^t} \mathbf{R}_i^T \mathbf{Y}_j$, then $\mathbf{X}_{j^*}$ is generator of an extreme ray of the cone $\mathcal{C}$. □

Remarks: (1) If the maximum occurs at two points j_1^* and j_2^*, both these points $\mathbf{X}_{j_1^*}$ and $\mathbf{X}_{j_2^*}$ generate the extreme rays of the cone $\mathcal{C}$. Hence both are added to anchor set A. If the maximum occurs at more than two points, some of these are the generators of the extreme rays of $\mathcal{C}$ and others are conic combinations of these generators. We can identify the extreme rays of this subset of points by calling Algorithm 1 recursively and add them to anchor set A. (2) Note that the algorithm is not influenced by presence of repeated anchors. (3) In the Algorithm, the vector $\mathbf{p}$ simply needs to satisfy $\mathbf{p}^T\mathbf{x}_i > 0, i = 1\dots n$. In our implementation, we simply used $\mathbf{p} = [1, \dots 1] \in \mathbb{R}^m$, i.e., $\mathbf{p}^T\mathbf{x}_i = \|\mathbf{x}_i\|_1$. (4) Note that unlike Gillis & Vavasis (2012), we do not need $\mathbf{X}_A$ to be full-rank.

Exterior Point Selection: It can be noted that residual of any point exterior to the current cone (i.e., any $\mathbf{R}_i \neq 0$) can be used in the selection step of Algorithm 1. This gives us multiple ways of expanding the current cone depending on which i is chosen - all of which solve the separable problem but may behave very differently in the presence of noise. Some natural options are listed in Algorithm 1: choosing a random exterior point (Eqn. 2), one with maximum residual norm (Eqn. 3) or one which defines a normal to a supporting hyperplane of the current cone which "sees" maximum "mass" of points in its positive halfspace, as measured by Eqn. 4. In the experiments, we will refer to these variants as Xray (rand), Xray (max) and Xray (dist) respectively.

A Greedy variation for noisy data: In high dimensional noisy data almost all the points may masquerade as anchors. A natural choice is to expand the current cone greedily by selecting a point that best minimizes the current residual, i.e., $j^* = \arg\min_j \min_{\mathbf{b}>0} \|\mathbf{R} - \mathbf{X}_j\mathbf{b}^T\|_F^2$. This selection criterion simplifies to Eqn.5 in Algorithm 1 (referred as Xray (greedy) henceforth). One may view this approach as implementing a non-negative variant of simultaneous orthogonal matching pursuit (Tropp et al., 2006), which is a greedy approach to the problem of sparse regression of multiple response variables on the same subset of explanatory variables, i.e., for solving $\min_{\mathbf{B}\geq 0}\|\mathbf{X} - \mathbf{XB}\|_F^2$ s.t. $\|\mathbf{B}\|_{0,1} = r$ where $\|\mathbf{B}\|_{0,1}$ pseudo-norm counts the number of non-zero rows in $\mathbf{B}$. In the context of separable NMF, both response variables and explanatory variables are the columns data matrix $\mathbf{X}$. A relaxed version of this problem is solved in Esser et al. (2012) ($\min_{\mathbf{B}\geq 0}\|\mathbf{X} - \mathbf{XB}\|_F^2 + \lambda\|\mathbf{B}\|_{1,\infty}$). It is also possible to have $\|\mathbf{B}\|_{1,2}$ penalized variant (Tropp, 2006; Bien et al., 2010) which is natural for sparse multivariate regression problems. Note that the greedy approach is not guaranteed to solve the separable NMF problem, but may perform well in the noisy settings as we observe in our experiments. Intuitively, this variant is concerned with greedily optimizing all residuals on average at every iteration, instead of making a decision based on the residual of a single, albeit well-chosen, exterior point.

Solution Refinement and Model Selection: In practice, the separable solution $(\mathbf{W}, \mathbf{H})$ as obtained from Algorithm 1 may be further refined with a few steps of alternating optimization with respect to a divergence measure of interest (e.g., Frobenius reconstruction $\|\mathbf{X} - \mathbf{WH}\|_F^2$). Also, in real-world datasets, the value of r is typically unknown. Since our algorithms build the solution one anchor at a time, r can be set based on a performance measure evaluated on held-out data. Alternatively, Algorithm 1 can exit if the amount of improvement from introducing a new anchor falls below a prespecified threshold.

Scalability and Parallelization

Here we describe various implementation details that allow us to gracefully scale to large sparse datasets (e.g., document-term matrices). The detection step can be parallelized by scoring the candidate anchors simultaneously. Likewise, the projection step involves solving Eqn. 6, which is separable in the columns of $\mathbf{B}$ and hence can be optimized in parallel.

Detection Step: We avoid materializing the dense residual matrix $\mathbf{R}$ in the evaluation of the anchor selection criteria. Instead, we score candidate anchors on-the-fly as we compute (but not explicitly materialize) a matrix $\mathbf{Q} = \left(\mathbf{R}^T\mathbf{X}\right)_+ = \left(\mathbf{C} - (\mathbf{C}_A\mathbf{H})^T\right)_+$ where $\mathbf{C} = \mathbf{X}^T\mathbf{X}$ denotes a covariance matrix (word-by-word for topic modeling applications). Here, the potential sparsity, symmetry of the covariance matrix $\mathbf{C}$ as well as the non-negativity of $\mathbf{H}$ can be further exploited. For example, if $\mathbf{C}_{ij} = 0$, the corresponding entry in the product $(\mathbf{C}_A\mathbf{H})$ need not be computed, since the resulting negative value is anyway reset to zero by the $(\cdot)_+$ thresholding operator. On a P core machine, the selection criteria may be evaluated in $O(\frac{nnz(\mathbf{C})r}{P})$ time where $nnz(\mathbf{C})$ is the number of non-zeros in $\mathbf{C}$. If $\mathbf{C}$ is dense, we compute $\mathbf{Q}$ using parallel dense BLAS-3 operations. The one time computation of $\mathbf{C}$ is done via a parallel aggregation of rank-one outer-product terms defined by the rows of $\mathbf{X}$.

Projection Step: Algorithm 2 gives the steps of a cyclic block coordinate descent algorithm organized around very light-weight incremental sparsity-exploiting updates for solving Eqn. 6 (derivation omitted for brevity). The algorithm can be invoked in parallel on columns of $\mathbf{X}$ to compute the corresponding columns of $\mathbf{B}$. The previous value of $\mathbf{B}$ is used to warm start the optimization and typically a very small number of iterations is needed for convergence.

3. Empirical Observations

Here, we report extensive comparisons on synthetic and medium-scale topic modeling problems, and benchmark our parallel implementation on large text datasets on multicore machines and distributed systems. We compare with the methods proposed in Bittorf et al. (2012) (abbrv. as *Hottopixx*) and Gillis & Vavasis (2012) (abbrv. as GV), as well as traditional NMFs based on alternating optimization(Cichocki et al., 2009). The source codes for *Hottopixx* and GV were taken from the respective authors' websites. In comparisons with Esser et al. (2012), it was observed that it tends to select near-duplicate anchors, as also mentioned in Esser et al. (2012). This characteristic causes it to consistently perform less favorably compared to other methods unless the data is preprocessed in an adhoc fashion to remove similar columns of $\mathbf{X}$; hence we do not include it in our list of baselines. We also do not compare with Arora et al. (2012a) since *Hottopixx* reportedly performs better (Bittorf et al., 2012) and the algorithm requires parameters which are hard to guess apriori.

3.1. Synthetic experiments

We perform a synthetic experiment that injects controlled amount of noise to corrupt the separable structure. Each entry of the matrix $\mathbf{W} \in \mathbb{R}_+^{200\times 20}$ is generated i.i.d. according to a uniform distribution between 0 and 1. The matrix $\mathbf{H} \in \mathbb{R}_+^{20\times 210}$ is taken to be $[I_{20\times 20}\ \mathbf{H}']$ where each column of $\mathbf{H}' \in \mathbb{R}_+^{20\times 190}$ is generated according to a Dirichlet distribution whose parameters are chosen uniformly in $[0, 1]$. The data matrix $\mathbf{X}$ is set to $\mathbf{WH}+\mathbf{N}$ where each entry of noise matrix $\mathbf{N}$ is generated i.i.d. according to a Gaussian distribution with zero mean and std. dev. δ. Fig. 3 plots the fraction of correctly recovered anchors (averaged over 10 runs for each value of δ) against the noise level δ ranging from 0 to 1.5. The proposed XRAY (max) shows the best noise-robustness in terms of anchor recovery, followed by XRAY (dist) and GV. Although XRAY (greedy) does not perfectly resolve the separable NMF problem ($\delta = 0$), it performs better than *Hottopixx* and is competitive with GV for near-separable case ($\delta > 0$). As described below, on real datasets it turns out to be highly competitive. XRAY (rand), although solves the separable problem ($\delta = 0$), degrades significantly under noise, which shows that proper selection of an exterior point to expand the current cone is crucial for noise-robustness.

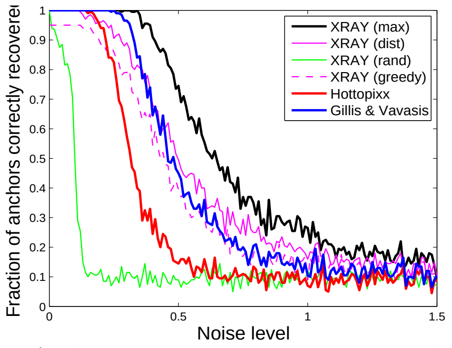

Figure 3. Anchor recovery rate versus noise level (best viewed in color)

3.2. Medium-scale Topic modeling problems

We evaluate the proposed methods on three human-labeled text datasets that are commonly used in topic modeling literature: TDT-2 (TDT2) ($m = 9394, n = 19528, r = 30$), BBC (Greene & Cunningham, 2006) ($m = 2225, n = 9635, r = 5$) and Reuters (Reuters) ($m = 7285, n = 18221, r = 10$). We used standard tf-idf representation with document frequency thresholding in constructing the data matrix $\mathbf{X}$. As required in *Hottopixx* and GV, we use ℓ_1-normalized columns of $\mathbf{X}$ (referred as matrix $\mathbf{X}^{(\ell_1)}$ henceforth) to identify the anchor column indices $A^{(\ell_1)}$, and use the unnormalized data $\mathbf{X}$ (and the corresponding anchor columns $\mathbf{X}_{A^{(\ell_1)}}$) for classification and clustering tasks. The use of $\mathbf{X}_A^{(\ell_1)}$ in clustering and classification (for any index set A) resulted in significantly worse performance uniformly for all methods so these results are not reported. For the sake of clarity in the figures, we do not show the results for XRAY (max) which performed almost similar to XRAY (dist) in these experiments. For an illustration of topics and anchor words recovered from these text datasets, the reader is referred to (Kumar et al., 2012).

Classification experiments: Figure 4 shows the classification accuracy results obtained with the features (columns of the document-term matrix restricted to anchor words) selected by different methods on the three datasets. Black dotted line is the classification accuracy with full features (all the words). We use 5% of the documents for training and the rest 95% for testing to emulate a semi-supervised learning scenario where we view various methods as inducing a topical representation based on all (unlabeled) data. We use multiclass SVM classifier as implemented in LIBLINEAR (Fan et al., 2008) and use four-fold cross validation to select the parameter C. Among separable NMF techniques, the proposed XRAY (greedy) and XRAY (dist) (with exception on Reuters) outperform *Hottopixx* and GV on all the three datasets, more so on TDT. On average, traditional NMFs with local

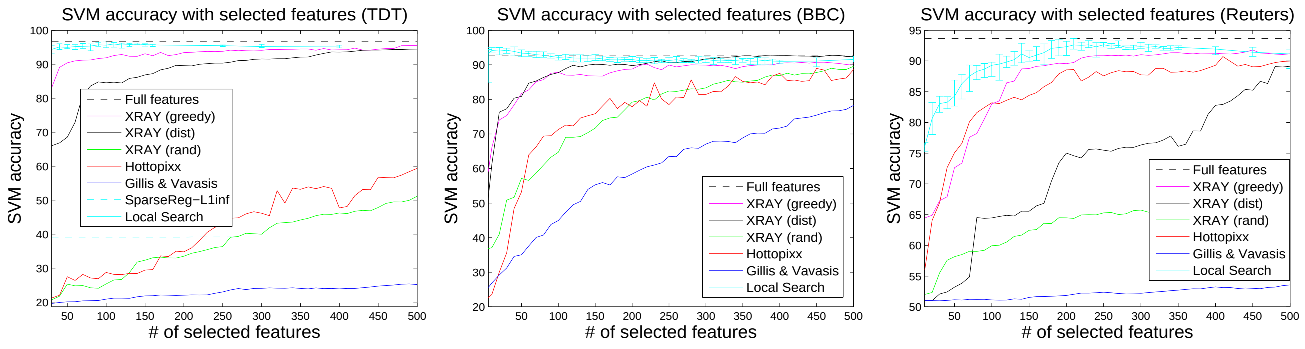

Figure 4. Classification Accuracy using selected features on TDT, BBC and Reuters datasets (best viewed in color)

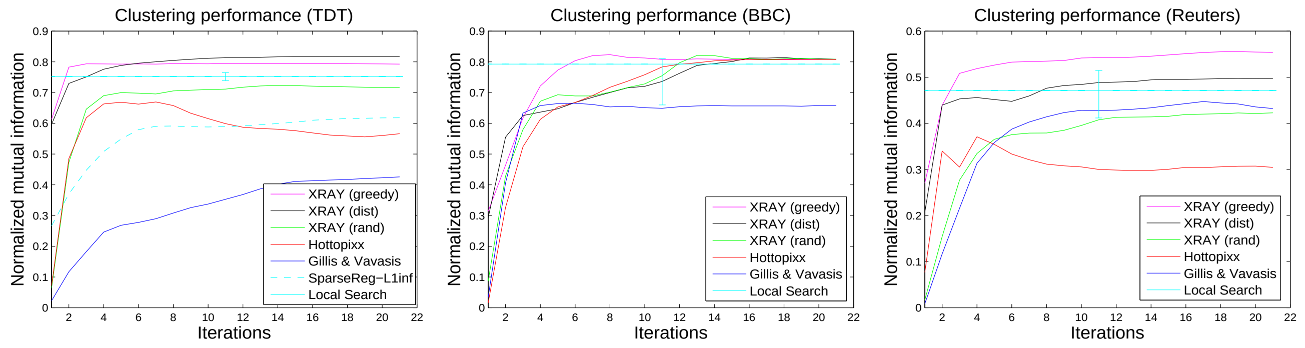

Figure 5. Clustering Performance on TDT, BBC and Reuters datasets (best viewed in color)

optimization perform quite well on these datasets especially when r is small, but can show significant performance variance (shown as error bars) with respect to initialization. As the number of topics increases, the performance gap between the proposed methods and the local optimization method rapidly diminishes. In this regime our techniques are a viable alternative to local optimization methods, and have the advantage of being local-minima-free, i.e., eliminating uncertainty with respect to initialization and therefore not requiring multiple runs.

Clustering experiments: We also evaluated clustering performance by assigning a cluster label to each document based on the maximum element in the corresponding row of $\mathbf{W}$. We refine the solution with a few iterations of alternating optimization. Figure 5 shows the clustering performance in terms of Normalized Mutual Information (NMI) as these iterations proceed. We also show the NMI obtained with local search method after it has converged to a local optimum (averaged NMI from ten runs with different random initializations is shown; error-bar indicates the variation around the average). Again, the proposed XRAY methods are among the best performing methods in terms of clustering performance and do not require multiple runs as traditional NMFs do.

Effect of column normalization: It can be seen from Table 1 that anchors selected using the unnormalized $\mathbf{X}$ (i.e., pure tf-idf features) achieve significantly better classification accuracy than anchors selected using ℓ_1-normalized version of $\mathbf{X}$. These empirical results suggest that discarding the norm information by explicit ℓ_1 normalization can adversely affect the predictive quality of the selected anchors. Such normalization is not required by our approach. We advocate that previous algorithms, which ignore this issue, should be carefully modified accordingly.

Table 1. Effect of word normalization (r=100).

	XRAY (dist)			XRAY (greedy)		
	ℓ_1	ℓ_2	None	ℓ_1	ℓ_2	None
TDT	21.06	83.04	84.52	31.69	90.05	91.87
BBC	58.59	84.36	87.73	75.41	82.18	87.82
REUT	51.88	76.88	64.46	55.65	81.28	83.02

3.3. Large-scale Experiments

We implemented a shared- and distributed-memory parallel version of XRAY in C++. That is, our implementation can exploit parallelism when running on multi-core machines, or on clusters of multi-core machines. For shared-memory parallelism, we use PFunc (Kambadur et al., 2009), a lightweight and portable library that provides C and C++ APIs to express task parallelism. For distributed-memory parallelism, we use MPI[1], a popular library specification for message-passing that is used extensively in high-

[1]http://www.mpi-forum.org/

Table 2. Datasets used for large-scale experiments. Times are for $r = 100$ on eight cores on `daniel`.

Name	#documents	#words	nnz($\mathbf{X}$)	nnz($\mathbf{X}^T\mathbf{X}$)	Sparsity($\mathbf{X}^T\mathbf{X}$)	Time($r = 100$)	Memory
RCV1	781265	43001	59.16e06	172.3e06	5%	409 secs	3.6 GB
PPL2	351849	44739	19.43e06	1.99e09	99.8%	1147 secs	30.2 GB
IBMT	124708	25998	1.03e06	1.77e06	0.2%	9.8 secs	1 GB

Table 3. Running times of XRAY versus *Hottopixx* on 8 threads for r topics and E epochs.

Dataset	XRAY (secs)			*Hottopixx* (secs)					
				E=5			E=10		
	r=25	r=50	r=100	r=25	r=50	r=100	r=25	r=50	r=100
IBMT	0.38	1.78	9.8	338.6	337.2	327.7	642.1	668.5	636.9
RCV1	15.4	67.2	409	2026.8	1938.3	1883.6	3769	3774.7	3888.9
PPL2	196	443.8	1147	1818.1	1935.5	1892.8	3725.2	3895.5	3913.7

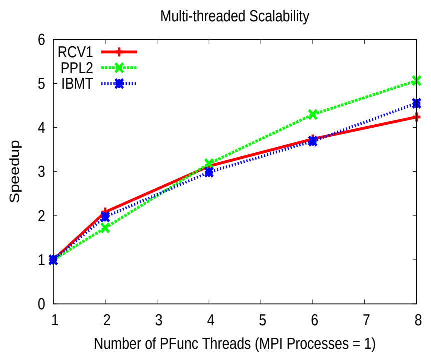

Figure 6. Multi-core speedup of RCV1, IBMT, and PPL2 datasets when running on `daniel`. All times are for R=100.

performance computing.

To test the shared-memory performance and scalability of XRAY , we ran experiments on `daniel`, a dual-socket, quad-core Intel® Xeon™ X5570 machine with 64GB of RAM running Linux Kernel 2.6.35-24 (total 8 cores). For compilation, we used GCC v4.4.5 with: "`-O3 -fomit-frame-pointer -funroll-loops`" in addition to PFunc 1.02, OpenMPI 1.4.5 and untuned ATLAS BLAS. We ran large-scale experiments on three datasets: RCV1 (Lewis et al., 2004), co-occurence matrix of people and places from ClueWeb09 dataset (Lemur), and IBM Twitter (IBMT) dataset. The statistics relating to these three large datasets are presented in Table 2.We report scalability results for XRAY (greedy) - other variants are computationally very similar.

Figure 6 depicts the multi-threaded performance of our implementation on `daniel` while detecting 100 topics. Our implementation is able to factorize RCV1 in 409 seconds on 8 cores and achieve 4.2x speedup over 8 threads when compared to the sequential implementation. Similarly, for IBMT we achieve 4.5x speedup, while completing the factorization in 9.8 seconds on 8 cores. For the dense $\mathbf{X}^T\mathbf{X}$ case, we are able to factorize PPL2 in 1147 seconds with just 8 cores. We believe that further speedup improvements can be demonstrated on these problems by (a) optimizing the data layout of various sparse matrices to alleviate memory contention amongst threads, and (b) in dense problems such as PPL2, by using a version of BLAS tuned to our architecture and by reorganizing our implementation around more BLAS-3 operations that have better memory to compute ratio than BLAS-1 or 2 operations. Our implementation showed good scalability on distributed-memory machines as well (details omitted for brevity).

To compare our performance against the state-of-the-art *Hottopixx* algorithm (Bittorf et al., 2012), we ran their algorithm on `daniel` with the options "`--dual 0.01 --epochs 10 --splits 8 --hott <R> --normse 1 --primal 1e-6`" set in close consultation with the authors. A detailed comparison is shown in Table 3.2. Note that a head-to-head comparison is difficult because of the different performance characteristics of *Hottopixx* and XRAY . For example, *Hottopixx* 's *per-epoch* runtime is not dependent on r, the number of topics, but it's accuracy is dependent on E, the number of epochs, while our methods execute exactly r iterations, where each iteration has a superlinear dependence on r. Nonetheless, for all three datasets with $r = 25, 50, 100$, we see that XRAY performs better than *Hottopixx* even when *Hottopixx* is run only for 5 epochs. In particular, for the sparse datasets IBMT and RCV1, in comparison to *Hottopixx* , XRAY runs to completion in significantly shorter amount of time.

Acknowledgments: We thank Victor Bittorf and Ben Recht for help with *Hottopixx*. Haim Avron, Christos Boutsidis, Ken Clarkson, Nicolas Gillis, Rick Lawrence and Ankur Moitra provided helpful feedback. Research was sponsored by the U.S. Defense Advanced Research Projects Agency (DARPA) under the Social Media in Strategic Communication (SMISC) program, Agreement Number W911NF-12-C-0028. Abhishek Kumar was a SMISC summer intern at IBM Research when this work was done.

References

Arora, Sanjeev, Ge, Rong, Kannan, Ravi, and Moitra, Ankur. Computing a nonnegative matrix factorization – provably. In *STOC*, 2012a.

Arora, Sanjeev, Ge, Rong, and Moitra, Ankur. Learning topic models - going beyond svd. In *FOCS*, 2012b.

Bien, J., Xu, Y., and Mahoney, M. CUR from a sparse optimization viewpoint. In *NIPS*, 2010.

Bittorf, Victor, Recht, Benjamin, Re, Christopher, and Tropp, Joel A. Factoring nonnegative matrices with linear programs. In *NIPS*, 2012.

Buhlmann, Peter and Geer, Sara Van De. *Statistics for High Dimensional Data*. Springer, 2010.

Cichocki, A., Zdunek, R., Phan, A. H., and Amari, S. *Non-negative Matrix and Tensor Factorizations*. Wiley, 2009.

Clarkson, K. More output-sensitive geometric algorithms. In *FOCS*, 1994.

Donoho, D. and Stodden, V. When does non-negative matrix factorization give a correct decomposition into parts? In *NIPS*, 2003.

Dula, J. H., Hegalson, R. V., and Venugopal, N. An algorithm for identifying the frame of a pointed finite conical hull. *INFORMS Jour. on Comp.*, 10(3): 323–330, 1998.

Elhamifar, Ehsan, Sapiro, Guillermo, and Vidal, Rene. See all by looking at a few: Sparse modeling for finding representative objects. In *CVPR*, 2012.

Esser, Ernie, Mller, Michael, Osher, Stanley, Sapiro, Guillermo, and Xin, Jack. A convex model for non-negative matrix factorization and dimensionality reduction on physical space. *IEEE Transactions on Image Processing*, 21(10):3239 – 3252, 2012.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. Liblinear: A library for large linear classification. *JMLR*, 2008.

Gillis, Nicolas and Vavasis, Stephen A. Fast and robust recursive algorithms for separable nonnegative matrix factorization. *arXiv:1208.1237v2*, 2012.

Greene, Derek and Cunningham, Pádraig. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *ICML*, 2006.

Hsieh, C. J. and Dhillon, I. S. Fast coordinate descent methods with variable selection for non-negative matrix factorization. In *KDD*, 2011.

Kambadur, P., Gupta, A., Ghoting, A., Avron, H., and Lumsdaine, A. PFunc: Modern Task Parallelism For Modern High Performance Computing. In *ACM/IEEE conference on Supercomputing*, 2009.

Kumar, Abhishek, Sindhwani, Vikas, and Kambadur, Prabhanjan. Fast conical hull algorithms for near-separable non-negative matrix factorization. *arXiv:1210.1190*, 2012.

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

Lemur. `http://lemurproject.org/clueweb09/`.

Lewis, D, Yang, Y, Rose, T, and Li, F. RCV1: A new benchmark collection for text categorization research. *Journal Of Machine Learning Research*, 5: 361–397, 2004.

Lin, C.-J. Projected gradient methods for non-negative matrix factorization. *Neural Computation*, 2007.

Nemirovski, A. *Lecture Notes: Introduction to Linear Optimization*. 2010.

Reuters. `archive.ics.uci.edu/ml/datasets/Reuters-21578+Text+Categorization+Collection`.

TDT2. `http://www.itl.nist.gov/iad/mig/tests/tdt/1998/`.

Tropp, J. A. Algorithms for simultaneous sparse approximation. part ii: Convex relaxation. *Signal Processing*, 86:589–602, 2006.

Tropp, J. A., Gilbert, A. C., and Strauss, M. J. Algorithms for simultaneous sparse approximation. part i: Greedy pursuit. *Signal Processing*, 86:572–588, 2006.

Vavasis, S. On the complexity of non-negative matrix factorization. *SIAM Journal on Optimization*, 20 (3):1364–, 2009.